SURT 2.0: Advances in Transducer-based Multi-talker Speech Recognition

Desh Raj, *Student Member, IEEE*, Daniel Povey, *Fellow, IEEE*, and Sanjeev Khudanpur, *Member, IEEE*

***Abstract*—The Streaming Unmixing and Recognition Transducer (SURT) model was proposed recently as an end-to-end approach for continuous, streaming, multi-talker speech recognition (ASR). Despite impressive results on multi-turn meetings, SURT has notable limitations: (i) it suffers from *leakage* and *omission* related errors; (ii) it is computationally expensive, due to which it has not seen adoption in academia; and (iii) it has only been evaluated on synthetic mixtures. In this work, we propose several modifications to the original SURT which are carefully designed to fix the above limitations. In particular, we (i) change the unmixing module to a mask estimator that uses dual-path modeling, (ii) use a streaming zipformer encoder and a stateless decoder for the transducer, (iii) perform mixture simulation using force-aligned subsegments, (iv) pre-train the transducer on single-speaker data, (v) use auxiliary objectives in the form of masking loss and encoder CTC loss, and (vi) perform domain adaptation for far-field recognition. We show that our modifications allow SURT 2.0 to outperform its predecessor in terms of multi-talker ASR results, while being efficient enough to train with academic resources. We conduct our evaluations on 3 publicly available meeting benchmarks — LibriCSS, AMI, and ICSI, where our best model achieves WERs of 16.9%, 44.6% and 32.2%, respectively, on far-field unsegmented recordings. We release training recipes and pre-trained models:** `https://sites.google.com/view/surt2`**.**

***Index Terms*—Multi-talker ASR, transducers, SURT.**

I. Introduction

The last decade has seen tremendous advancement in the transcription of single-speaker utterances [1], [2], resulting from the adoption of deep learning methods and large-scaled supervised training [3]. With conversational speech recognition (ASR) systems reaching super-human performance, researchers are now turning towards more challenging settings such as multi-talker conversations [4]–[6]. This setting is often characterized by overlapping speech, turn-taking, and far-field audio, and therefore requires special modeling techniques to tackle the problem [7]–[9]. Despite these challenges, multi-talker ASR, particularly in the context of meeting transcription, has a long history originating from the NIST Rich Transcription evaluation [10], [11], and its subsequent application in recognizing dinner-party conversations [4], [6], [12].

A conventional modeling approach for multi-talker ASR is through a cascade of separation and recognition systems. This approach leverages advancements in speech separation research to obtain single-speaker audio [13], which can then be used with a regular ASR component [14]; however, such a model may be sub-optimal since the components are independently optimized, and may also require greater engineering efforts for maintenance [15].

Due to these limitations with cascaded systems, researchers have proposed jointly optimized models that combine separation and ASR and directly solve for the task of multi-talker transcription, often using a permutation-invariant training (PIT) objective [16]. Such a paradigm has been explored in the context of hybrid HMM-DNN systems [17], and more recently for end-to-end ASR [18], with most research focusing on attention-based encoder-decoders (AEDs). For meeting transcription, a well-studied framework is *serialized output training* (SOT), wherein multiple references are serialized into a single prediction sequence based on special tokens [19]. A detailed review of related work is presented in Section II.

For the case of streaming multi-talker ASR, two methods have been proposed to extend neural transducers [20] for the task. These methods are beneficial since neural transducers (using RNNs or transformers) have become the standard modeling technique for on-device speech recognition [21]–[23]. The first method, exemplified by token-level SOT (t-SOT) [24], uses a conventional single-branch model similar to single-talker ASR, whereas the second method relies on a two-branch modeling strategy, as exemplified by *streaming unmixing and recognition transducer* (SURT) [25] and *multi-turn RNNT* (MT-RNNT) [26]. In this paper, we focus on the latter class of models, and refer to them as SURT, but the same ideas should also be applicable to MT-RNNT. The SURT model separates overlapping speech into multiple simultaneous *branches* (or channels), each of which is transcribed by a shared transducer. It has been extended to handle long-form multi-turn recordings [27], [28], and to jointly perform speaker identification [29], endpointing [30], and segmentation [31].

Although SURT is a promising framework for end-to-end multi-talker ASR, it suffers from several limitations. It was shown in [27] that SURT performance often degrades on multi-turn sessions due to *omission* and *leakage* related errors. Here, omission refers to the case when an utterance is missed by all output branches, whereas leakage happens when a non-overlapping segment is transcribed on multiple branches. Additionally, SURT requires training on long sessions with the transducer loss, which may be computationally prohibitive, or even infeasible using academic resources. Consequently, there exist no open-source recipes for training SURT-like models.

This work was partially funded by the National Science Foundation CCRI program via Grant No. 2120435, and a fellowship from Amazon via the JHU-Amazon Initiative for Interactive Artificial Intelligence (AI2AI).

Desh Raj is with the Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: draj@cs.jhu.edu).

Daniel Povey is with Xiaomi Corp., Beijing, China (e-mail: dpovey@gmail.com).

Sanjeev Khudanpur is with the Center for Language and Speech Processing, and Human Language Technology Center of Excellence, Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: khudanpur@jhu.edu).

Furthermore, all evaluations of SURT so far are limited to the setting of simulated meetings (such as LibriCSS), and it is not clear whether the models transfer well to real-world settings.

In this work, our objective is three-fold:

(i) to improve the training efficiency of SURT models so that they are feasible to train on an academic budget;
(ii) to analyze errors (such as those caused by omission and leakage), and develop methods to improve multi-talker ASR performance of SURT models; and
(iii) to demonstrate the effectiveness of the resulting model on real meeting benchmarks.

Towards these objectives, we propose systematic modifications of several aspects of SURT — relating to the model design, network architecture, training mixture simulation, loss functions, and training schemes. We conduct ablation studies to demonstrate the impact of each of these design choices. For (iii), we demonstrate the efficacy of the resulting SURT 2.0 on three public benchmarks: LibriCSS, AMI, and ICSI. Across all settings, our model outperforms the original SURT and MT-RNNT in terms of WER performance, while being smaller and more training efficient. We release all training recipes and model checkpoints through the open-source `icefall` toolkit: `https://desh2608.github.io/pages/surt2`.

II. Related Work

A. Joint optimization for multi-talker ASR

Early work on joint separation and ASR involved hybrid HMM-DNN models as the ASR backbone [17], [32]. These models were often trained with auxiliary speaker information [33] or using transfer learning from single-speaker acoustic models [34]. With the success and flexibility of end-to-end ASR systems [3], [20], [21], [23], [35]–[38], researchers quickly adapted these into jointly optimized multi-talker ASR pipelines [18], [39]. Similar training schemes — speaker embeddings, curriculum learning, or knowledge distillation — were used to improve these pipelines [40]–[42]. Multi-channel extensions of these models have also been proposed that seek to improve separation capabilities through neural beamforming techniques [43], [44]. For the single-channel case, time-domain modeling has been used to improve separation and consequently benefit downstream ASR performance [45]. However, most of these models were studied in the context of AEDs that do not perform streaming transcription[1], and evaluated on the simple setting of fully-overlapping two-speaker mixtures (such as the WSJ-Mix dataset [49]). On the other hand, real-world multi-talker settings usually contain an arbitrary number of speakers and sparse overlaps.

B. Serialized output training (SOT)

SOT was first proposed as a method to use existing AED architectures for multi-talker ASR [19]. It was extended to perform joint speaker counting and speaker identification using an auxiliary speaker inventory [50], [51]. A token-level variant of SOT, in conjunction with neural transducers, has shown strong performance on streaming multi-talker ASR [24], [52], and has also been combined with multi-channel front-ends [53] and large-scale pre-training [54]. An advantage of t-SOT is that it allows the same model and training scheme to be used for both single and multi-talker settings. The recently concluded M2MeT challenge used SOT as the baseline system [55], [56].

[1]There are concurrent efforts to perform streaming ASR with AEDs using monotonic attention mechanisms [46]–[48], but with limited adoption.

C. Continuous speech separation

Continuous speech separation (CSS) refers to the task of generating overlap-free speech signals from a continuous audio stream consisting of multiple utterances spoken by different people [57]. This task has its origins in early work on *unmixing transducers* [58], [59]. CSS research has seen increased activity in the last few years due to interest in sparsely-overlapped meeting separation. The original model (which used PIT-based supervised training of BLSTM encoders) has been improved by leveraging better architectures such as Conformer [60], two-stage training [15], and large-scale semi-supervised and self-supervised learning [61], [62]. While the original CSS used PIT-based training, other training methods such as recurrent selective attention network (RSAN) [63], [64] and Graph-PIT [65], [66] have also been investigated. Multi-channel extensions of CSS have been proposed using complex spatial features [67], low-latency beamforming [68], and direction-of-arrival (DOA) based source localization [69]. The SURT model may be regarded as a jointly optimized version of CSS with transducer-based ASR.

D. Other cascaded systems

Recent editions of the CHiME challenge [6], [70] saw the use of cascaded multi-talker ASR systems that leverage guided source separation (GSS) [71]. GSS is an unsupervised target-speaker extraction (TSE) method that relies on pre-computed speaker activities and blind source separation to perform front-end enhancement of overlapped speech signals. Since its proposal, block-online [72] and GPU-accelerated [73] variants have been proposed to speed up the method. GSS has been combined with strong back-end ASR to obtain state-of-the-art performance on the CHiME-5 dataset [74]; however, the separation performance depends heavily on estimated speaker activity, and ASR results may degrade heavily if the segmentation is poor [75], [76]. Recently, a spatial mixture model based meeting separation method was proposed which generalizes GSS and avoids some of its pitfalls [77]. Nevertheless, this approach is naturally offline and requires multiple microphone channels to perform well. Other cascades of separation and ASR for multi-talker transcription have explored methods such as neural TSE [78], mixture-invariant training [79], sequential neural beamforming [14], and cross-channel attention [80].

E. Multi-talker datasets

Research on multi-talker ASR has benefited heavily from the availability of public corpora. These include early benchmarks such as AMI [7], ICSI [81], and Sheffield Wargames [82], and recently released datasets such as CHiME-5 [70], DiPCo [12], LibriCSS [57], AISHELL-4 [83], AliMeeting [55], and RAMC [84]. In this work, we evaluate our models on the LibriCSS, AMI, and ICSI meeting data.

This article has been accepted for publication in IEEE/ACM Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2023.3318398

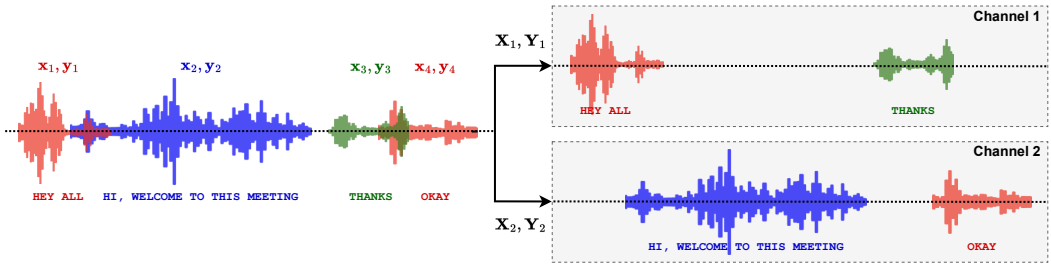


Fig. 1. Example of heuristic error assignment training (HEAT). Each color represent a different speaker. The utterances ($\mathbf{x}_u$,$\mathbf{y}_u$) are assigned, in order of start times, to the next available channel. Such an assignment avoids the exponential complexity associated with permutation invariant training (PIT).

III. Preliminary

A. Speech recognition with neural transducers

In conventional single-talker ASR, audio features for a segmented utterance $\mathbf{X} \in \mathbb{R}^{T \times F}$ (where T and F denote the number of time frames and the input feature dimension, respectively) are provided as input to the system, and we are required to predict the transcript $\mathbf{y} = (y_1, \ldots, y_U)$, where $y_u \in \mathcal{V}$ denotes output units such as graphemes or word-pieces, and U is the length of the label sequence. For the case of discriminative training, this requires computing the conditional likelihood $P(\mathbf{y}|\mathbf{X})$ (or its log for numerical stability). For inference, we search for $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{X})$, often in a constrained search space using greedy or beam search. Transducers achieve this by marginalizing over the set of all alignments $\mathbf{a} \in \bar{\mathcal{V}}^{T+U}$, where $\bar{\mathcal{V}} = \mathcal{V} \cup \{\phi\}$ and ϕ is called the blank label. Formally,

$$P(\mathbf{y}|\mathbf{X}) = \sum_{\mathbf{a} \in \mathcal{B}^{-1}(\mathbf{y})} P(\mathbf{a}|\mathbf{X}), \tag{1}$$

where $\mathcal{B}$ is a deterministic mapping from an alignment $\mathbf{a}$ to an output sequence $\mathbf{y}$. In the original transducer, all alignments $\mathbf{a}$ consist of $\mathbf{y}$ interspersed with T blank tokens, usually represented as a $T \times U$ lattice with ϕ on horizontal arcs and $\mathbf{y}$ on vertical arcs. Since there may be $\binom{T+U}{U}$ such paths on the lattice, some transducer variants (such as recurrent neural aligner [85] or monotonic RNN-T [86], [87]) restrict the number of non-blank tokens emitted per time step. External or internal alignments may also be used to further prune the lattice for marginalization [88], [89].

Transducers parameterize $P(\mathbf{a}|\mathbf{X})$ with an encoder, a prediction network, and a joiner (see "recognition" component in Fig. 2a). The encoder maps $\mathbf{x}$ into hidden representations $\mathbf{f}_1^T$, while the prediction network maps $\mathbf{y}$ into $\mathbf{g}_1^U$. The joiner combines the outputs from the encoder and the prediction network to compute logits $\mathbf{z}_{t,u}$ which are fed to a softmax function to produce a posterior distribution over $\bar{\mathcal{V}}$. Under assumptions of full context encoder, we can expand (1) as

$$P(\mathbf{y}|\mathbf{X}) = \sum_{\mathbf{a} \in \mathcal{B}^{-1}(\mathbf{y})} \prod_{t=1}^{T+U} P(\mathbf{a}_t|\mathbf{f}_1^T, \mathbf{g}_1^{u(t)-1}) \tag{2}$$

$$= \sum_{\mathbf{a} \in \mathcal{B}^{-1}(\mathbf{y})} \prod_{t=1}^{T+U} \mathrm{Softmax}(\mathbf{z}_{t,u(t)}), \tag{3}$$

where $u(t) \in \{1, \ldots, U\}$ denotes the index in the label sequence at time t. We will denote the log of this expression as $\mathcal{L}_{\text{rnnt}}(\mathbf{y}, \mathbf{z})$ (or simply $\mathcal{L}_{\text{rnnt}}$) for the remainder of this paper.

B. Multi-talker ASR with SURT

In multi-talker ASR, the input $\mathbf{X} \in \mathbb{R}^{T \times F}$ is an unsegmented mixture containing N utterances from K speakers, i.e., $\mathbf{X} = \sum_{n=1}^{N} \mathbf{x}_n$, where $\mathbf{x}_n$ is the n-th utterance ordered by start time, shifted and zero-padded to the length of $\mathbf{X}$. The desired output is $\mathbf{Y} = \{\mathbf{y}_n : 1 \leq n \leq N\}$, where $\mathbf{y}_n$ is the reference corresponding to $\mathbf{x}_n$. If N is small and fixed (e.g., $N = 2$), a permutation-invariant training strategy may be used for this task. However, for general multi-talker scenarios, N may be arbitrarily large.

Assuming at most two-speaker overlap, the *heuristic error assignment training* (HEAT) paradigm [25] is used to create channel-wise references $\mathbf{Y}_1$ and $\mathbf{Y}_2$ by assigning $\mathbf{y}_n$'s to the first available channel, in order of start time (Fig. 1). Formally, if $\zeta : n \to \{1, 2\}$ maps utterances to channels, θ_n^{st} and θ_n^{en} denote the start and end times for utterance n, and θ_n^{st} is monotonically increasing $\forall n$, we have

$$\zeta(n) = \begin{cases} 1, & \text{if } \theta_n^{\text{st}} \geq \max_{i \in \zeta^{-1}(1)} \theta_i^{\text{en}} \\ 2, & \text{otherwise,} \end{cases} \tag{4}$$

and $\zeta(n)$'s are assigned sequentially. SURT estimates $\hat{\mathbf{Y}} = [\hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2] = f_{\text{surt}}(\mathbf{X})$ as follows. First, an unmixing module computes $\mathbf{H}_1$ and $\mathbf{H}_2$ as

$$\mathbf{H}_1 = \mathbf{M} * \bar{\mathbf{X}}, \quad \mathbf{H}_2 = (\mathbb{1} - \mathbf{M}) * \bar{\mathbf{X}}, \tag{5}$$
$$\mathbf{M} = \sigma(\mathrm{MaskEnc}(\mathbf{X})) \text{ and } \bar{\mathbf{X}} = \mathrm{MixEnc}(\mathbf{X}),$$

where $\bar{\mathbf{X}}, \mathbf{M}, \mathbb{1} \in \mathbb{R}^{T \times D}$ (for latent dim. D) is a matrix of ones, σ is the sigmoid function, and $*$ is Hadamard product. $\mathbf{H}_1$ and $\mathbf{H}_2$ are fed into a transducer-based ASR, producing logits $\mathbf{Z}_1$ and $\mathbf{Z}_2$. Finally,

$$\mathcal{L}_{\text{heat}} = \mathcal{L}_{\text{rnnt}}(\mathbf{Y}_1, \mathbf{Z}_1) + \mathcal{L}_{\text{rnnt}}(\mathbf{Y}_2, \mathbf{Z}_2), \tag{6}$$

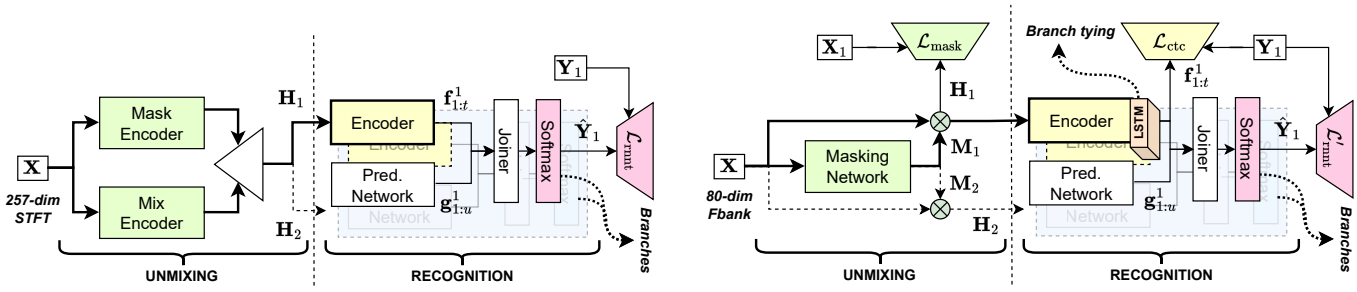
where $\mathcal{L}_{\text{rnnt}}$ is the standard RNN-T loss [20]. SURT performs *speaker-agnostic* transcription, and is evaluated using an optimal reference combination WER (ORC-WER) [28], as described in Section V-B.

IV. Methodology: SURT 2.0

Multi-talker ASR with SURT requires the model to perform well on three challenging sub-tasks:

1. continuous separation of sparsely overlapped speech;
2. long-form speech recognition; and
3. modeling quick turn-taking among multiple speakers.

This article has been accepted for publication in IEEE/ACM Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2023.3318398



(a) **Original SURT** (b) **SURT 2.0**

Fig. 2. Overview of (a) original SURT, and (b) SURT 2.0. $\mathbf{X}$ denotes the input mixture. $\mathbf{X}_1$ and $\mathbf{Y}_1$ are concatenated sources and references for the first branch (only required during training). For SURT 2.0, the "recognition" component is pre-trained on single-speaker data. Table I provides a summary of differences between the models.

TABLE I
OVERVIEW OF DIFFERENCES BETWEEN ORIGINAL SURT AND SURT 2.0 (FOR TRAINING ON LIBRISPEECH MIXTURES).

	Original SURT	**SURT 2.0**
Modeling	Features: 257-dim STFT Mask/mix encoders for unmixing	Features: 80-dim fbank Masking network for unmixing
Architecture	Mask/mix encoder: Conv2D Encoder: DP-LSTM / DP-Transformer Predictor: LSTM	Mask encoder: DP-LSTM Encoder: Branch-tied zipformer Predictor: Conv1D
Loss function	$\mathcal{L}_{\text{rnnt}}$	$\mathcal{L}'_{\text{rnnt}} + \lambda_{\text{ctc}}\mathcal{L}_{\text{ctc}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}}$
Mixture simulation	Source: LibriSpeech utterances 2 speakers, $\leqslant$ 4 turns Avg. duration (s): 25.5±5.0† Far-field: simulated RIRs	Source: LibriSpeech segments 2–3 speakers, $\leqslant$ 9 turns Avg. duration (s): 16.2±4.7 Far-field: real RIRs
Pre-training & adaptation	None	Transducer pre-training Adaptation using in-domain data

†Based on our estimate using the simulation configuration.

The failure cases in SURT can be linked to model degeneration on one or more of these sub-tasks. For instance, *omission* and *leakage* errors may be attributed to (1), while high error rates on quick turn-taking scenarios with short silences (see Section VI) may be caused by (3). Deletion errors on long sequences may be caused by both (1) and (2). SURT 2.0 contains several modifications aimed at addressing each of these sub-tasks, and to improve training efficiency of the model. Our modifications are summarized in Table I, and the resulting model is shown in Fig. 2b.

A. Modeling

For "unmixing" (or separation), SURT used dual mix/mask encoders that projected input 257-dim STFTs into high dimensional representations, as shown in (5). Clearly, this design constrains SURT to have exactly two output branches, and the separated features are not interpretable. Instead, we use 80-dimensional log Mel filter-banks as inputs, and replace the mix/mask encoders with a simple masking network that can generate arbitrary number of masks. Formally, given output channel count C, our unmixing module generates masks

$$[\mathbf{M}_1, \ldots, \mathbf{M}_C]^T = \text{MaskNet}(\mathbf{X}), \tag{7}$$

where $\mathbf{M}_c \in \mathbb{R}^{T\times F}$. These masks are applied to the input $\mathbf{X}$ to obtain channel-specific features: $\mathbf{H}_c = \mathbf{M}_c * \mathbf{X}$. Such a design has three advantages: (i) it allows the use of arbitrary number of output branches C, (ii) the masked representations

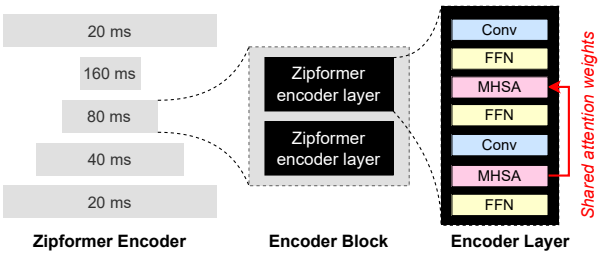


Fig. 3. Illustration of the zipformer encoder architecture. The encoder contains multiple "blocks" running at different frame rates (left). Each block contains several "layers" (middle), and each layer performs self-attention twice with shared attention weights (right).

$\mathbf{H}_c$ are interpretable as clean features, and (iii) it allows pre-training of the recognition module on single-speaker speech. We will describe (ii) and (iii) further in subsequent sections.

Our recognition module is similar to SURT, other than architectural changes (Section IV-B). Channel-wise features $\mathbf{H}_c$ are fed into an encoder which generates hidden representations $\mathbf{f}^c_{1:T}$. The corresponding label sequence $\mathbf{Y}_c$ is fed into a prediction network as per the HEAT training strategy, generating hidden representations $\mathbf{g}^c_{1:U}$. A joiner combines $\mathbf{f}^c_{1:T}$ and $\mathbf{g}^c_{1:U}$ to generate logits $\mathbf{z}^c_{t,u}$. The parameters of the encoder, prediction network, and joiner are shared among all the output branches, as shown in Fig. 2. Further details about model hyperparameters are given in Section V-C.

B. Network architecture

We made several changes to the network architecture of SURT modules. For the masking network, we use dual-path LSTMs (DP-LSTMs) instead of 2-D convolutions in order to improve long range modeling capability for unmixing [90]. Second, in the encoder, we replace the DP-LSTMs (or DP-Transformers) with the recently proposed zipformer [91]. As shown in Fig. 3, the zipformer encoder contains multiple encoder blocks running at different rates, with the middle ones more strongly down-sampled (by up to a factor of 8). This makes training more efficient as there are fewer frames to evaluate. Each block may contain one or more encoder "layers" operating at the same frame rate. Each layer performs self-attention twice with shared attention weights, and a trainable bypass is introduced for each layer dimension. We also propose "branch tying" of the encoder outputs to *jointly* learn representation across all branches, i.e.,

$$[\hat{\mathbf{h}}^{\text{enc}}_1, \ldots, \hat{\mathbf{h}}^{\text{enc}}_C]^T = \text{LSTM}\left([\mathbf{h}^{\text{enc}}_1, \ldots, \mathbf{h}^{\text{enc}}_C]^T\right). \tag{8}$$

The motivation for branch-tied encoders is to reduce errors from *omission* and *leakage*, which usually happen when output branches do not communicate. Finally, we use a "stateless" prediction network [92] instead of LSTM layers in the original SURT. In addition to improving computational efficiency, we conjecture that a stateless network should also be better suited to handle quick turn-taking (sub-task 3 described earlier). We set the segment length for the bi-directional intra-LSTM (of the DP-LSTM network) equal to the chunk size of the causal Zipformer encoder; this is the overall latency of the SURT model.

C. Training objective

SURT was trained end-to-end with the transducer loss [20], with the utterance-wise permutation resolved by HEAT (Fig. 1). However, the transducer loss suffers from high memory usage, since it requires marginalization over a logit tensor of size (B, T, U, D). Even with an efficient implementation [23], this severely limits the training sequence length — for example, in [27], the authors trained SURT with mixtures containing at most 4 speaker turns. To remedy this issue, we replace the full-sum transducer loss with the recently proposed *pruned* transducer loss, which prunes the alignment lattice using a simple linear joiner before computing the full sum on the pruned lattice [89].

Recall from Section III-A that the encoder and the prediction network generate hidden representations $\mathbf{f}_1^T$ and $\mathbf{g}_1^U$, respectively. In pruned transducer, these are first projected to $\mathbb{R}^{|\bar{\mathcal{V}}|}$, and denoted as $\hat{\mathbf{f}}_1^T$ and $\hat{\mathbf{g}}_1^U$, respectively. A simple additive joiner is then used to compute *trivial* logits $\hat{\mathbf{z}}_{t,u}$ as

$$\hat{\mathbf{z}}_{t,u} = \hat{\mathbf{f}}_t + \hat{\mathbf{g}}_u - \hat{\mathbf{z}}_{t,u}^{\text{norm}}, \tag{9}$$

$$\text{where} \quad \hat{\mathbf{z}}_{t,u}^{\text{norm}} = \log \sum_v \exp\left(\hat{\mathbf{f}}_t + \hat{\mathbf{g}}_u\right). \tag{10}$$

Here, equation (10) can be interpreted as log-space matrix multiplication, and is easy to implement through simple matmul operations. Gradients from this simple joiner are used to compute locally optimal pruning bounds for the lattice[2], and the full joiner output, $\mathbf{z}_{t,u}$, is only computed on the pruned lattice. We used the open-source implementation available in `k2`: `https://github.com/k2-fsa/k2`.

Additionally, we use auxiliary loss functions to regularize SURT training and improve separation of sparsely overlapped speech. For the former, we add a connectionist temporal classification (CTC) loss [35], which has been shown to provide regularization capabilities due to monotonicity in alignments [93], [94]. This is given as

$$\mathcal{L}_{\text{ctc}} = \log \sum_{\mathbf{a} \in \mathcal{B}_{\text{ctc}}^{-1}(\mathbf{y})} \prod_t P(\mathbf{a}_t | \mathbf{f}_1^T), \tag{11}$$

where $\mathbf{a}$ is a T-length sequence that deterministically maps to $\mathbf{y}$ through transformation $\mathcal{B}_{\text{ctc}}$, which removes repeated tokens and ϕ. To improve mask estimation, we apply a masking loss directly on the outputs $\mathbf{H}_c$ generated by the mask encoder as

$$\mathcal{L}_{\text{mask}} = \sum_{c \in C} \text{MSE}(\mathbf{H}_c, \mathbf{X}_c), \tag{12}$$

where MSE denotes mean-squared error, and $\mathbf{X}_c$ is obtained by summing clean inputs $\mathbf{x}_u$ assigned to branch c (Fig. 1). The overall training objective is given as

$$\mathcal{L} = \mathcal{L}'_{\text{rnnt}} + \lambda_{\text{ctc}} \mathcal{L}_{\text{ctc}} + \lambda_{\text{mask}} \mathcal{L}_{\text{mask}}, \tag{13}$$

where $\mathcal{L}'_{\text{rnnt}}$ denotes the pruned transducer loss and λ's are hyperparameters.

Algorithm 1: Training mixture simulation

Input: $\mathcal{X}$, $\mathcal{M}$, K, T
Output: $\mathcal{S}$

$D_{=\text{spk}}, D_{\neq\text{spk}}, D_{\text{ovl}}, \mathcal{S} = \phi$
// Fit distributions to $\mathcal{M}$
for M *in* $\mathcal{M}$ **do**
  for i *in range*$(|M|)$ **do**
    $t = M_i.start - M_{i-1}.end$
    if $M_i.spk == M_{i-1}.spk$ **then**
      $D_{=\text{spk}} = D_{=\text{spk}} \cup \{t\}$
    else
      if $t > 0$ **then**
        $D_{\neq\text{spk}} = D_{\neq\text{spk}} \cup \{t\}$
      else
        $D_{\text{ovl}} = D_{\text{ovl}} \cup \{-t\}$
$P_{\text{ovl}} = \frac{|D_{\text{ovl}}|}{|D_{\neq\text{spk}}|+|D_{\text{ovl}}|}$; $D_* = \text{histogram}(D_*)$
// Generate mixtures using $\mathcal{X}$
$\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_S\}$ // speaker wise bucketing of $\mathcal{X}$
while *any*$(|\mathcal{X}_s| > 0)$ **do**
  // Select speakers
  $k \leftarrow \text{sample}(K)$; $\mathcal{X}_{s_1}, \ldots, \mathcal{X}_{s_k} \leftarrow \text{sample}(\mathcal{X})$
  // Select utterances for each speaker
  for i *in range*(k) **do**
    $U_{s_k} \leftarrow \text{sample}(\mathcal{X}_{s_k})$, s.t. $\left(\sum_{u \in U_{s_k}} u.dur\right) < T$
    $\mathcal{X}_{s_k} \leftarrow \mathcal{X}_{s_k} \setminus U_{s_k}$
  $U = \text{shuffle}(U_{s_1}, \ldots, U_{s_k})$; offset = 0
  // Get offsets for each utterance
  $\mathcal{S}_{\text{cur}} \leftarrow \phi$ // initialize empty mixture
  for i *in range*$(|U|)$ **do**
    if $U_i.spk == U_{i-1}.spk$ **then**
      ot = sample$(D_{=\text{spk}})$
    else
      if *Bernoulli*$(P_{ovl}$*>0.5)* **then**
        ot = –sample(D_{ovl})
      else
        ot = sample$(D_{\neq\text{spk}})$
    offset = offset + ot
    $\mathcal{S}_{\text{cur}} = \mathcal{S}_{\text{cur}} \cup \{U_i, \text{offset}\}$
  $\mathcal{S} = \mathcal{S} \cup \mathcal{S}_{\text{cur}}$

D. Mixture simulation

The original SURT was trained on synthetic mixtures of *full* utterances, which may result in prohibitively long sequences;

[2]The pruning is "locally" optimal in the sense that the optimization is performed per-frame, as opposed to a "globally" optimal treatment which would consider the whole path through the lattice. This local strategy may result in path discontinuity that is later resolved through adjustments.

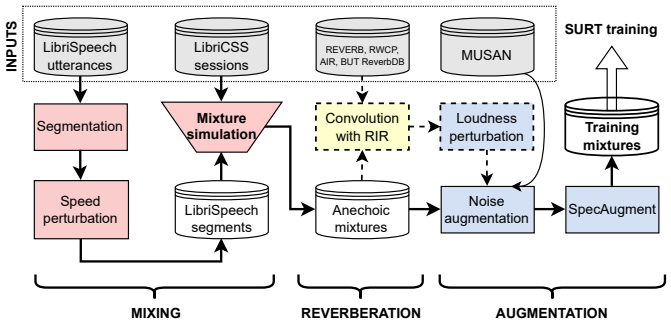


Fig. 4. Mixture simulation workflow for LibriSpeech-based training. Gray cylinders denote external data used during simulation: LibriSpeech utterances, statistics from LibriCSS, real RIRs, and MUSAN noises. The simulation workflow can be conceptually divided into three phases: (i) *mixing*, which creates anechoic meetings, (ii) *reverberation*, which convolves with RIRs, and (iii) *augmentation*, which perturbs the reverberant mixtures with various schemes. The dotted path is optional for anechoic training.

for e.g., LibriSpeech `train` has an average duration of 12.4s. Furthermore, segment overlaps were arbitrarily determined to satisfy overlap ratio constraints, which may not be representative of the target sessions.

For SURT 2.0, we use *sub-segments* instead of full utterances as the source for mixture simulation, so that resulting mixtures are shorter while retaining multiple turns of conversation. These sub-segments are obtained using word-level alignment information, by breaking up the utterances at pauses longer than a threshold τ. As an example, using $\tau = 0.2$ for LibriSpeech resulted in sub-segments that were 2.8s on average. This allowed each training session to contain up to 9 turns of conversation while still being 36.5% shorter than the original training mixtures (Table I). Second, we learn histograms of pause/overlap distribution statistics from the target sessions, and sample from these distributions for mixing the segments. Such a strategy has been successfully applied to improve end-to-end neural diarization [95]. Our mixture simulation algorithm is described in Algorithm 1, and is similar to the conversation simulation algorithm from [95]. We assume that the input to the algorithm is the source segments $\mathcal{X}$, target sessions $\mathcal{M}$ (to learn statistics), maximum number of speakers K in each mixture, and maximum duration T of a speaker in a mixture. The algorithm returns the training mixtures $\mathcal{S}$. For training, the input speech mixture $\mathbf{X}$ is obtained by digitally adding the utterances in $\mathcal{S}$ with the specified offsets. The utterance-wise labels $\mathbf{y}_n$, along with θ_n^{st} and θ_n^{en} (uniquely determined by the offset and duration of utterance n) are used to obtain the reference transcripts for both channels using equation (4). We optionally convolve $\mathcal{S}$ with real room impulse responses (RIRs) to train models for far-field reverberant conditions. The overall simulation workflow is shown in Fig. 4.

E. Pre-training & adaptation

Using sub-segments for mixture simulation improves training efficiency while allowing multiple speaker turns; however, it creates a train-test mismatch for duration of individual segments, which could degrade the model's performance on sub-task 2, i.e., to recognize long utterances from the same speaker. We solve this problem by pre-training the transducer module on single-speaker utterances (e.g., on LibriSpeech `train` set). Such a pre-training strategy also decouples the tasks of learning to separate from learning to transcribe, and helps the SURT model converge faster. Recall from Section IV-A that this pre-training is possible in SURT 2.0 because our masking network generates masked filter-banks instead of high-dimensional latent representations.

Despite convolving with real RIRs, the acoustic characteristics of the mixtures used to train SURT may still be mismatched from real meeting recordings. As a final step of SURT training, we perform model adaptation by training on in-domain data for a small number of iterations.

V. Experimental Setup

A. Data

We performed evaluations on three publicly-available meeting datasets in English: LibriCSS, AMI, and ICSI. The summary statistics for these datasets are shown in Table II.

LibriCSS consists of multi-channel audio recordings of 8-speaker *simulated* conversations that were created by combining utterances from the LibriSpeech `test-clean` set [96] and playing in real meeting rooms. It comprises 10 one-hour long sessions, each made up of six 10-minute "mini sessions" that have different overlap ratios (ranging from 0% to 40%). We show results on *anechoic* and *replayed* versions of LibriCSS, where the former consists of digitally mixed utterances, and the latter is the replayed sessions (we choose the first channel for evaluation). We used session 0 as the `dev` set and the remaining for `test`, following previous work [14].

AMI consists of 100 hours of recorded meetings containing 4 or 5 speakers per session [7]. Sessions were recorded on close-talk (headset and lapel) microphones, as well as 2 linear arrays each containing 8 mics. We used three different mic settings for our experiments: IHM-Mix (digitally mixed individual headset mics), SDM (first channel of array-1), and MDM (beamformed array-1), where the last setting uses officially provided beamformed recordings [97].

ICSI data comprises around 72 hours of natural, meeting-style overlapped speech recorded at International Computer Science Institute (ICSI), Berkley. Speech was captured using close-talk headsets and ad-hoc distant microphones. The original data did not provide any partitions, so we use the speaker-disjoint partitions suggested in [98]. We performed evaluations on the IHM-Mix and SDM settings — for SDM, we selected the third distant mic, similar to the Kaldi recipe[3].

B. Evaluation

Fig. 1 shows that SURT generates speaker-agnostic transcription with no strict correspondence between speaker and channel. This makes it impossible to evaluate SURT with the conventional word error rate (WER) metric used for single-speaker ASR systems. [27] and [28] independently proposed similar metrics that computed the minimum word error rate (WER) based on an optimal assignment of references to the channels, and this was termed optimal reference combination WER (ORC-WER) in [28]. In both studies, the authors remarked that computation of this metric was exponential in the number of reference utterances. Von Neumann et al. [99]

[3]https://github.com/kaldi-asr/kaldi/tree/master/egs/icsi

TABLE II
STATISTICS OF DATASETS USED FOR EVALUATIONS. THE k-SPEAKER DURATIONS ARE IN TERMS OF FRACTION OF TOTAL SPEAKING TIME.

	LibriCSS		AMI			ICSI		
	Dev	**Test**	**Train**	**Dev**	**Test**	**Train**	**Dev**	**Test**
Duration (h:m)	1:00	9:05	79:23	9:40	9:03	66:38	2:16	2:45
Num. sessions	6	54	133	18	16	70	2	3
Silence (%)	6.2	6.7	18.1	21.5	19.6	55.2	25.9	25.9
1-speaker (%)	81.3	81.2	75.5	74.3	73.0	82.1	90.3	84.9
2-speaker (%)	18.6	18.5	21.1	22.2	21.0	15.7	9.0	13.6
>2-speaker (%)	0.1	0.4	3.4	3.5	6.0	2.2	0.7	1.4

proposed a polynomial-time implementation of ORC-WER using multi-dimensional Levenshtein distance, and released it through the `meeteval`[4] toolkit. They showed that ORC-WER is a lower bound on the popular concatenated minimum-permutation WER (cpWER) [6], and becomes equal to the cp-WER when no speaker errors are present. We used their open-source implementation for evaluating our SURT models. In the remainder of this paper, we will use the abbreviation WER to actually mean ORC-WER, unless explicitly mentioned. For LibriCSS, we report WERs on the officially provided (approximately 1 min. long) "segments," following prior work on continuous input evaluation [27], [57], [60]. For AMI and ICSI, we use utterance-group based evaluation similar to [54], [100]. This results in `dev`/`test` segments of duration 7.0s/8.4s and 2.6s/2.9s for AMI and ICSI, respectively.

C. Implementation details

Network architecture. We experimented with two variants of SURT — *base* and *large*. The *base* model contains four 256-dim DP-LSTM layers trained with chunk width randomization [27] as the masking network. The encoder consists of 5 zipformer blocks with 2 self-attention layers per block. Each block consists of a 192-dim attention distributed across 8 heads, and a 768-dim feed-forward layer. Downsampling factors of (1,2,4,8,2) were used in the zipformer blocks. The prediction network contains a single 512-dim Conv1D layer. The *large* model contains 6 layers in the masking network, and (2,4,3,2,4) self-attention layers in the 5 zipformer blocks. The chunk size for the intra-LSTM and the Zipformer is set to 32 frames, resulting in a modeling latency of 320 ms.

Training data. For LibriCSS experiments, we first created anechoic mixtures, **LSMix-clean**, using each speed-perturbed LibriSpeech train sub-segment ($\tau = 0.2$; cf. § IV-D) once, resulting in approx. 2200h of training data. For this, we set $\mathcal{M}$ as the LibriCSS `dev` set (excluding the 0L and OV10 sessions), $K = 3$, and $T = 15$ in Algorithm 1. A reverberated copy of LSMix, named **LSMix-reverb**, was generated by convolving LSMix with real RIRs collected from the REVERB [5] dataset. We also added isotropic noises from the REVERB data, and perturbed the loudness between -20 dB and -25 dB using the `pyloudnorm` tool [101]. We will refer to the combination of LSMix-clean and LSMix-reverb as **LSMix-full** hereafter. Ablation experiments were conducted on the *anechoic* LibriCSS by training SURT on LSMix-clean. For these experiments, we used on-the-fly noise augmentation using noises from the MUSAN corpus [102]. For final evaluation, we trained SURT on LSMix-full (∼4400h), so that the same model can be used on both anechoic and replayed LibriCSS. All models were trained using on-the-fly SpecAugment [103]. We found it beneficial to use high overlap mixtures in the warm-up stage of SURT training to encourage better mask estimation [104]. For pre-training on single-speaker data, we used LibriSpeech `train` set, optionally convolved with synthetic RIRs (for the final evaluation). This pre-training was done for 10 epochs.

Since AMI and ICSI have similar characteristics, we trained a combined SURT model for them using synthetic mixtures created from close-talk utterances, again using sub-segments obtained from forced alignments ($\tau = 0.5$). These mixtures were obtained by setting $D_{=\text{spk}}$, $D_{\neq\text{spk}}$, D_{ovl}, and P_{ovl} as 0.5, 0.5, 1.0, and 0.8, respectively, in Algorithm 1. K and T were set to 3 and 15s, respectively, similar to LSMix-clean simulation. We refer to these mixtures as **AIMix-clean**, their reverberant copy as **AIMix-reverb**, and the combination as **AIMix-full**. The models were subsequently adapted by combining the real train sessions from all microphone settings.

Hyper-parameters. The auxiliary loss scales, λ_{ctc} and λ_{mask}, were set to 0.2 each. $\mathcal{L}_{\text{mask}}$ was not used during adaptation since ground-truth separated audio may not be available for real data. We trained the models with the ScaledAdam optimizer following the standard zipformer-transducer recipes in `icefall`. This is a variant of Adam where each parameter's update is scaled proportional to the norm of that parameter. The learning rate was warmed up to 0.004 for 5000 iterations, and decayed exponentially thereafter. All models were trained for 30 epochs using 4 GPUs[5].

Decoding. Checkpoints from the last 9 epochs were averaged for decoding. We conducted experiments with both greedy decoding and beam search. For the ablation experiments, we used greedy decoding for faster turn-around. For the final evaluation, we used a "modified" version of beam search with beam size of 4. This variant constraints the emission of at most 1 non-blank token at each time step, which allows batched decoding. We normalized the replayed LibriCSS recordings to -23 dB using `pyloudnorm` for inference.

VI. RESULTS & DISCUSSION

A. Comparison with baselines

First, we compared the SURT 2.0 *base* and *large* models with the baselines from literature: SURT [27] and MT-RNNT [28], on the LibriCSS "anechoic" and "replayed" recordings. These comparisons are shown in Table III. We cannot compare SURT with published results on t-SOT [24] since the latter is evaluated using the `asclite`-based speaker-agnostic WER (SAg-WER) metric. SAg-WER requires token-level time-stamp estimation, and does not penalize an utterance being split into multiple "channels," which is penalized by ORC-WER. The models were trained on LSMix-full, and adapted to LibriCSS using the `dev` set. For adaptation, we segmented the `dev` sessions in 2 ways, by cutting at maximum pause durations of 0.1s and 0.5s, respectively. This resulted

[4]https://github.com/fgnt/meeteval

[5]We used either Titan RTX (with batch size 500s) or V100 (with batch size 650s) depending on availability on our compute cluster.

TABLE III
COMPARISON WITH ORIGINAL SURT AND MT-RNNT MODELS ON "ANECHOIC" AND "REPLAYED" VERSIONS OF LIBRICSS `test` SET. THE SURT 2.0 MODELS WERE DECODED USING BEAM SEARCH WITH A BEAM SIZE OF 4.

Model	Size (M)	Anechoic							Replayed						
		0L	0S	OV10	OV20	OV30	OV40	Avg.	0L	0S	OV10	OV20	OV30	OV40	Avg.
Original SURT [27]	42.9	6.9	18.9	19.6	21.9	23.9	28.7	20.0	9.3	21.1	21.2	25.9	28.2	31.7	22.9
Multi-turn RNN-T [28]	81.0	–	–	–	–	–	–	–	14.8	14.5	18.0	25.8	30.3	32.3	22.6
SURT 2.0 (Base)	26.7	5.2	4.7	14.2	17.8	21.3	23.0	14.4	6.7	8.2	23.0	27.1	28.5	31.8	20.9
↪ w/ `dev` adaptation	26.7	5.1	4.2	13.7	18.7	20.5	**20.6**	13.8	6.8	7.2	21.4	24.5	28.6	31.2	20.0
SURT 2.0 (Large)	37.9	**4.6**	**3.8**	14.9	17.3	19.1	23.9	13.9	**5.9**	7.8	21.2	25.7	27.8	29.9	19.7
↪ w/ `dev` adaptation	37.9	**4.6**	**3.8**	**12.7**	**14.3**	**16.7**	21.2	**12.2**	6.4	**6.9**	**17.9**	**19.7**	**25.2**	**25.5**	**16.9**

in 381 sub-sessions of average duration 17.9s, totalling approximately 1.88h each from anechoic and replayed conditions and containing 18.6% overlapped speech. The model was then trained on the combined adaptation data for 8 epochs (for *base*) or 15 epochs (for *large*). The learning rate was warmed up to 0.0004 for 2 (or 4) epochs and decayed thereafter. A single V100 GPU was used for adaptation.

As the overlap ratio increased from 0% to 40%, the WER also increased, which is expected. Even without adaptation, SURT-base outperformed the original SURT on the anechoic setting, improving the average WER from 20.0% to 14.4%. This may be because the anechoic training mixtures are well matched to the evaluation condition in the absence of far-field artifacts. The largest WER difference was on the 0S (0% overlap with short pauses) condition, where SURT-base achieved 4.7%, compared to 18.9% for SURT. We attribute this improvement primarily to pre-training on single-speaker data, which allows the model to handle non-overlapping speech well. On using model adaptation, the anechoic WER further improved by 0.6% absolute, with consistent improvements across most overlap conditions. The largest improvement was obtained for the OV40 sessions, where WER reduced from 23.0% to 20.6%. When evaluated on replayed LibriCSS, SURT-base was better than SURT and MT-RNNT on average, but slightly worse on overlapped conditions. This may be because we used a limited set of real RIRs for simulating reverberant training mixtures, whereas SURT and MT-RNNT used on-the-fly simulated RIRs[6]. Adaptation on the `dev` set improved performance across all settings, with the resulting average WER reducing to 20.0%. Note that the same model was used to evaluate both settings, whereas in the original SURT, separate models were trained for each setting.

The *large* model followed similar trends as the *base* model, but provided consistent improvements in WER across most conditions. For unadapted models, larger improvement was observed on the replayed setting compared to the anechoic setting (5.7% vs. 3.5% relative). We conjecture that the larger masking network (6 DP-LSTM layers) may be better suited for unmixing reverberant features, leading to improved WERs. We also found that the *large* model benefited more from adaptation on in-domain data, perhaps due to higher representation capacity. The relative WER improvement from adaptation was 12.2% and 14.2% for the anechoic and replayed conditions, respectively, whereas for the *base* model, the improvements were 4.2% and 4.3%. Overall, our SURT-large model provided relative WER improvements of 39.0% (20.0% $\rightarrow$ 12.2%) and 26.2% (22.9% $\rightarrow$ 16.9%) over the original SURT.

[6]We experimented with using simulated RIRs, but we found that it consistently degraded WERs on the 0S condition, similar to the original SURT.

B. Effect of architectural changes

Recall from Section IV-B that SURT 2.0 contains several architectural changes compared to the original SURT. These include: (i) DP-LSTMs instead of Conv2D in the unmixing module, (ii) branch-tied encoders, and (iii) a stateless prediction network. We performed ablation experiments to evaluate the effect of each choice, as shown in Table IV. Each of the first three rows denote the performance when one of the components is changed (shown in red), while the last row shows the configuration of the final SURT 2.0 model. We selected model configurations such that all models have roughly the same number of parameters. All models were trained on LSMix-clean until convergence, and evaluated on the anechoic LibriCSS setting with greedy decoding. The recognition modules were pre-trained for all setups, but no auxiliary losses or adaptation were used. We also tried replacing the zipformer encoder with a DP-LSTM (as was used in the original SURT), but this model did not converge. This may be because the original SURT used 3-fold sub-sampled STFT features as input, and the training mixtures themselves were shorter on average (cf. Section IV-D). As a result, the input sequences for the encoder in SURT 2.0 are approximately 3-5 times longer, which may affect convergence in architectures that do not use self-attention.

The largest performance degradation was caused by replacing the DP-LSTM based masking network with Conv2D. This is consistent with speech separation research where dual-path encoders usually provide large improvements due to their ability to model long sequences [90]. Without the DP-LSTM masking, the unmixing module was effectively futile, as evident by the high WERs on the overlapping sessions.

For the encoder architecture, applying branch tying using equation (8) provided significant improvements on high overlap conditions. For example, the relative WER reduction with branch tying was 23.3% and 19.2% on the OV30 and OV40 settings, respectively. The improvement was largely due to a reduction in deletion errors (at the cost of a small increase in insertions). For example, on the OV40 setting, the

TABLE IV
EFFECT OF ARCHITECTURAL CHOICES FOR VARIOUS COMPONENTS, SHOWN ON "ANECHOIC" LIBRICSS test SET. THE LAST ROW DENOTES THE FINAL NETWORK ARCHITECTURES FOR SURT 2.0.

Masking network	Branch tying	Pred. network	Size (M)	0L	0S	OV10	OV20	OV30	OV40	Avg.
Conv2D	✓	Conv1D	25.3	13.9	6.9	27.0	35.4	41.0	45.4	28.3
DP-LSTM	✗	Conv1D	24.6	6.6	5.4	21.3	26.6	33.1	41.2	22.4
DP-LSTM	✓	LSTM	28.1	7.6	6.3	**17.2**	26.7	26.8	34.7	19.9
DP-LSTM	✓	Conv1D	26.7	**6.4**	**5.1**	17.5	**23.5**	**25.4**	**33.3**	**18.5**

TABLE V
EFFECT OF AUXILIARY OBJECTIVES ON "ANECHOIC" LIBRICSS test SET. ALL MODELS USED THE SURT-BASE ARCHITECTURE.

$\mathcal{L}_{\text{ctc}}$	$\mathcal{L}_{\text{mask}}$	0L	0S	OV10	OV20	OV30	OV40	Avg.
✗	✗	6.4	5.1	17.5	23.5	25.4	33.3	18.5
✓	✗	6.0	5.2	17.9	23.7	22.8	29.6	17.5
✗	✓	**5.6**	**4.9**	16.3	21.3	24.6	29.5	17.1
✓	✓	6.1	5.0	**13.6**	**19.0**	**21.1**	**26.5**	**15.2**

deletion error rate reduced from 20.6% to 14.8%, with the insertion increasing from 3.5% to 5.5%. This validates our conjecture that branch tying helps in alleviating errors caused by *omissions* where complete segments are skipped by both branches. We will analyze these errors further in Section VI-E.

Finally, we observed small but consistent improvements on replacing the LSTM-based prediction network (used in the original SURT) with a stateless network (i.e., using a Conv1D layer). The largest relative improvement for this change was seen in the 0S setting, where WER improved by 19.0% (compared to 7.0% relative WER improvement overall). As mentioned in Section IV-B, we conjecture that this may be because a stateless decoder is more suited to frequent context switching that is required for modeling quick turn-taking.

C. Effect of auxiliary objectives

We also performed ablation experiments to study the effect of $\mathcal{L}_{\text{ctc}}$ and $\mathcal{L}_{\text{mask}}$. Similar to Section VI-B, we trained SURT-base models on LSMix-clean for this investigation, and evaluated the models on anechoic LibriCSS using greedy search. The results are shown in Table V.

First, we see that adding CTC loss on the encoder improved WER mainly for highly overlapped sessions. We obtained relative improvements of 10.2% and 11.1% on the OV30 and OV40 sessions, respectively. For sessions with more overlaps and turn-taking, both output branches may contain several speech segments. We conjecture that an auxiliary CTC objective may be useful in aligning the segments to the corresponding audio during training, resulting in better modeling for high overlap sessions. To validate our conjecture, we plotted the occupation probabilities for the nodes in the RNN-T lattice (of shape $T \times U$), as obtained from the gradients of the simple additive joiner used in the pruned transducer loss [89]. These values should correspond to a soft alignment between the input and the label sequence[7]. In Fig. 5, we show example plots for a randomly selected mixtures from the training set, using models trained with and without $\mathcal{L}_{\text{ctc}}$. When the auxiliary CTC loss was used, the model was able to better align the silence region (time frames 100 to 250) in channel 2, as discernible through the bright horizontal line.

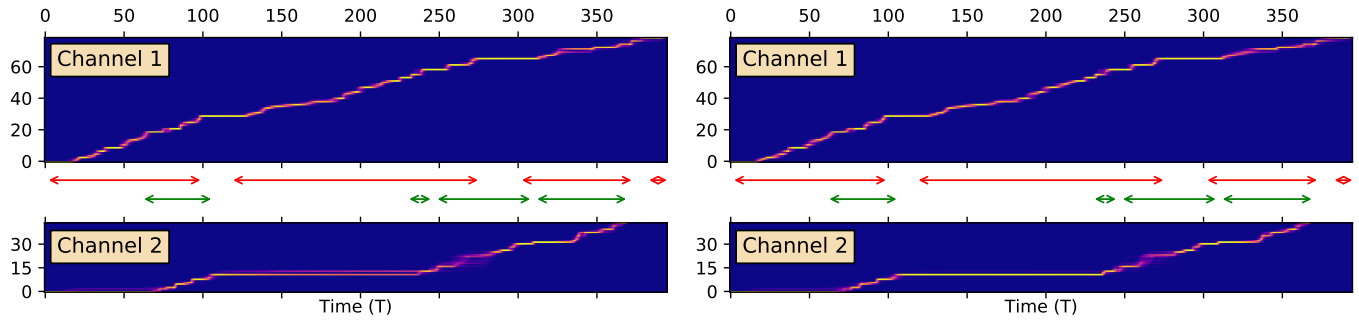
Using auxiliary masking loss $\mathcal{L}_{\text{mask}}$, as defined in equation (12), again improved WER performance over the SURT-base model, as shown in the third row of Table V. Surprisingly, we observed most improvements in the low-overlap conditions — for instance, 12.5% relative WER reduction for 0L. Most of this improvement again resulted from reduction in deletion errors (2.0% → 1.1% for 0L). We also experimented with using a graph-PIT based masking loss [65] instead of the HEAT-based loss, but it did not provide similar improvements. This may be because our final transducer objective uses the HEAT formulation. Finally, the best WER results were obtained on combining both the auxiliary objectives. The resulting model demonstrated a relative WER reduction of 17.8% over the SURT-base model trained without these objectives.

D. Effect of pre-training

Pre-training on single-speaker utterances was found to be one of the most effective strategies for faster and better convergence of SURT models[8]. To quantify this improvement, we computed the WER on the anechoic LibriCSS dev set (averaged across all overlap conditions) after each epoch of training a SURT-base model on the LSMix-clean data, as

[7]It makes more sense to use this value instead of CTC alignments because (i) the model trained without $\mathcal{L}_{\text{ctc}}$ cannot provide corresponding alignments, and (ii) we use the transducer head for the ASR task.

[8]Such a pre-training strategy has also been used in recent work on multi-channel serialized output training [53].



(a) Without $\mathcal{L}_{\text{ctc}}$ (b) With $\mathcal{L}_{\text{ctc}}$

Fig. 5. Occupation probabilities of nodes in the $T \times U$ lattice for both output channels, for models trained with and without auxiliary CTC loss ($\mathcal{L}_{\text{ctc}}$. T and U are on x and y axes, respectively. For this mixture, $T = 395$, $U_1 = 79$, and $U_2 = 44$. The double arrows between the plots denote the reference segments: red for channel 1 and green for channel 2. Brighter colors denote higher occupation probabilities.

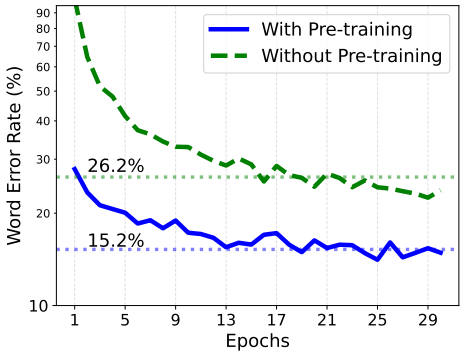


Fig. 6. Effect of transducer pre-training with LibriSpeech. WERs are shown for the LibriCSS `dev` set after each training epoch. The dotted horizontal lines show the final WERs on `test` set with model averaging.

shown in Fig. 6. We observe that when pre-training was used, the SURT model converged much faster and to a better WER. In fact, it surpassed the model without pre-training after just 5 epochs. On the anechoic `test` set, the models obtained WER of 26.2% and 15.2%, respectively, as shown by the dotted horizontal lines. The increase in computational time is marginal, since we only pre-train the transducer for 10 epochs (instead of training to convergence). Alternatively, an off-the-shelf streaming transducer may also be plugged in for this purpose, since the *branch tying* of the encoder is only added at the time of SURT training. Such a pre-training scheme is analogous to a curriculum learning strategy using single-speaker utterances, with the masks fixed as $\mathbf{M}_1 = J_{T,F}$ and $\mathbf{M}_c = 0 \cdot J_{T,F}$, $\forall c \neq 1$, where $J_{t,f} = 1$, $\forall t, f$.

E. Measuring leakage and omission

Throughout this paper, we have mentioned *leakage* and *omission*, first identified in [27], as major contributors of errors in SURT. In this section, we provide a metric for quantifying these error sources, in terms of n-gram counts on the reference $\mathbf{Y}$ and hypotheses $\hat{\mathbf{Y}}$, as defined in Section III-B.

omission@n For some n, the fraction of all unique n-grams in $\mathbf{Y}$ not present in any $\hat{\mathbf{Y}}_c$.

leakage@n For some n, the fraction of all unique n-grams in $\mathbf{Y}$ present in multiple $\hat{\mathbf{Y}}_c$.

If emission time-stamps are known, these definitions may be modified to include time windows for n-gram search. In their absence, the leakage@n and omission@n rates are upper and lower bounds on the actual leakage and omission, respectively. Nevertheless, by quantifying these error types explicitly, we can gain some insights about the model behavior. Our SURT model in this paper does not predict token time-stamps, so we estimate omission and leakage over the entire hypotheses.

In Table III, we showed WERs achieved by the SURT models with and without adaptation, when decoded using beam search with a beam of size 4. In general, ASR model performance improves on increasing the beam size, but we found that for SURT models, the WER first improved (up to a beam size of 4) and then degraded. This trend can be explained by looking at the leakage and omission errors, as shown in Fig. 7. We used the SURT-base model (with and without adaptation) for decoding the anechoic and replayed sets, using beam sizes varying from 1 to 8. In the figure, we show insertion and deletion error rates, as well as leakage@4 and omission@4. We observe that leakage first decreased (from beam size 1 to 2) but then increased gradually, while the opposite trend was observed for omission. Furthermore, adapted models reduce omissions significantly, at the cost of increase in leakage (particularly in the replayed setting).

TABLE VI
COMPARISON OF LEAKAGE@4 AND OMISSION@4 WITH INSERTION AND DELETION ERRORS ON THE "ANECHOIC" LIBRICSS `dev` SET.

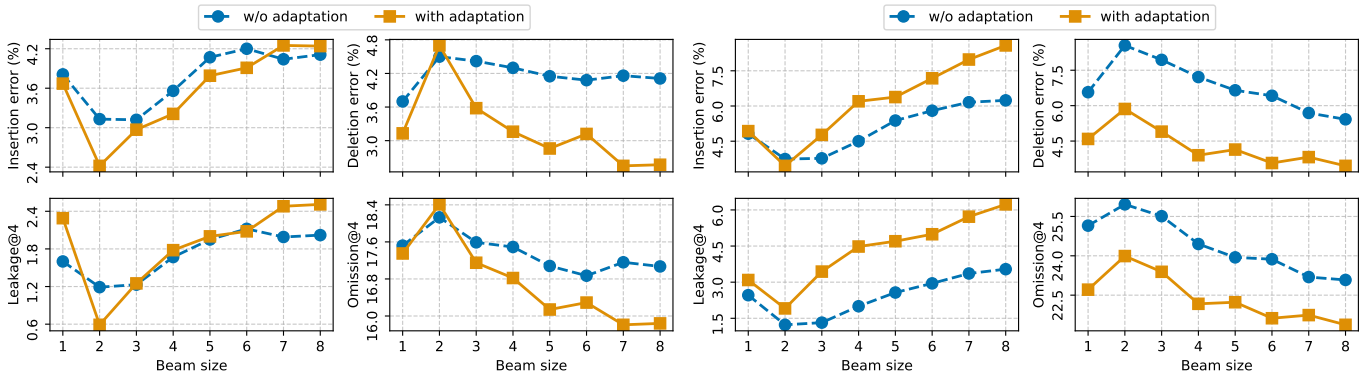
$\mathcal{L}_{\text{ctc}}$	$\mathcal{L}_{\text{mask}}$	**Ins.**	**Del.**	**L@4**	**O@4**
✗	✗	3.44	5.96	2.44	20.63
✓	✗	2.56	4.87	1.51	20.10
✗	✓	2.96	5.88	1.49	20.34
✓	✓	3.03	4.65	1.68	19.27

In the above analysis, the overall trend for leakage and omissions followed those of insertion and deletion errors, which is expected. However, this may not always be the case. In Table V, we showed results for ablation experiments done to investigate the effect of auxiliary objectives. We calculated the leakage@4 and omission@4 rates for those models on the anechoic `dev` set, and compared them with the corresponding insertion and deletion rates. The comparison is shown in Table VI. We see that although the models trained with $\mathcal{L}_{\text{ctc}}$ and $\mathcal{L}_{\text{mask}}$ have similar L@4 and O@4, the difference in insertion and deletion errors is comparatively large. This suggests that the additional insertions or deletions are not caused due to unmixing errors, and are most likely due to errors in the recognition module. This hypothesis seems reasonable because $\mathcal{L}_{\text{ctc}}$ should be more effective than $\mathcal{L}_{\text{mask}}$ in reducing purely ASR-related errors. When we further include $\mathcal{L}_{\text{mask}}$ in training (last row), both O@4 and deletion reduce by roughly 5% relative, indicating that the improvement almost entirely results from recovered sub-segments.

F. Results on AMI and ICSI

Finally, we evaluated our SURT models on two meeting benchmarks, AMI and ICSI, to verify their efficacy on real-world settings. For these experiments, we initialized SURT *base* and *large* models with the final checkpoint from the LSMix-full training, and continued training on the AIMix-full simulated mixtures for 30 epochs. The models were further adapted on the AMI and ICSI `train` sessions, by combining the IHM-Mix, SDM, and beamformed MDM recordings. For the adaptation, we cut the sessions at pauses of 0.0 and 0.5 seconds (thus creating two copies with different segmentations). We trimmed the long sessions to approximately 30s each based on utterance end time marks. This process resulted in 330k train samples of average duration 6.4s, totaling 590h and 18.4% overlapped speech. We adapted the SURT models on these sub-sessions by training with a LR of 0.0001 for 20 epochs, and used the final checkpoint for inference. The results with and without in-domain adaptation are shown in Table VII.

We found that the *large* model obtained consistently better WERs than the *base* model, which is expected. All models obtained lower WERs on ICSI, which may be due to its lower overlapped speech ratio (13.6%) compared to AMI (21.0%). A similar increase in WERs with higher overlaps was also observed for LibriCSS. Without model adaptation,



(a) **Anechoic** (b) **Replayed**

Fig. 7. Effect of decoding beam size on insertion and deletion errors, versus effect on leakage and omission (for $n = 4$), for SURT-base models with and without adaptation, for (a) "anechoic" and (b) "replayed" conditions, averaged across all overlap settings. The top row shows insertion and deletion errors, while the bottom row contains leakage@4 and omission@4.

TABLE VII
RESULTS ON THE AMI AND ICSI `test` SETS, UNDER DIFFERENT MICROPHONE SETTINGS, USING SURT *base* AND *large* MODELS.

Model	Adapt.	AMI			ICSI	
		IHM-Mix	SDM	MDM	IHM-Mix	SDM
Base	✗	39.8	65.4	46.6	28.3	60.0
	✓	37.4	46.9	43.7	26.3	33.9
Large	✗	36.8	62.5	44.4	27.8	59.7
	✓	35.1	44.6	41.4	24.4	32.2

the SURT models obtained reasonable WERs for the IHM-Mix and beamformed MDM settings, but not for SDM. For instance, the SURT-base model obtained 64.3% relatively worse WERs on SDM, as compared to IHM-Mix. This suggests that purely simulated mixtures cannot compensate for real, far-field training data. SDM performance improved significantly with in-domain adaptation, with relative WER decrease of 28.3% and 43.5% for SURT-base on AMI and ICSI, respectively.

Error analysis revealed that most of the errors were caused by deletion, as shown in Table VIII. For the unadapted models, deletion comprised 89.4% and 83.2% of the overall errors for the base and large SURT variants, respectively. We attribute this primarily to the model missing very short utterances and back-channels such as "Okay," "Hmm,", and so on, which form a significant fraction of such meetings, and which may be useful for downstream dialog understanding tasks. When we adapted the models using real, in-domain data, the deletion errors reduced significantly, with minor increase in insertion and substitution. For the adapted models, deletion comprised 68.9% and 66.8% of the total errors for the base and large SURT, respectively. In future work, it may be interesting to investigate models and training objectives which avoid suppression of short overlapping segments.

VII. CONCLUSIONS AND FUTURE WORK

We performed a detailed investigation of the SURT model for multi-talker speech recognition. By decomposing the continuous, streaming, multi-talker ASR problem into the three components of sparsely overlapped speech separation, long-form ASR, and quick turn-taking modeling, we were able to identify model design strategies that improve performance on one or more of these sub-problems. We modified SURT's unmixing component to perform mask estimation on filter-bank inputs, which allowed the use of transducer pre-training on single-speaker utterances. To improve training efficiency, we applied zipformer blocks in the encoder which aggressively subsample the input sequence and use shared attention masks within the blocks. We also used sub-segments instead of full utterances to simulate training mixtures, which resulted in more frequent turn-taking without increasing the training sequence length. We further replaced the full-sum transducer loss with the recently proposed pruned transducer to reduce memory requirement for loss computation. We found that using auxiliary objectives for the encoder and the masking network also improves the model's performance, for instance by producing better soft alignments of the input and output sequences during training, or by reducing leakage in single-speaker regions. To further reduce errors caused by leakage and omission, we used dual-path LSTMs instead of convolutional layers in unmixing, and added branch tying of encoder outputs in the recognition component. We trained SURT 2.0 in multiple stages: (i) single-speaker pre-training, (ii) training on simulated mixtures, and (iii) adaptation on in-domain real data, to outperform the larger and computationally expensive SURT models proposed previously on LibriCSS. We also demonstrated the viability of these models for real meeting benchmarks, namely AMI and ICSI.

Although SURT has shown promising results while being efficient for training and inference, there are several avenues for further enhancements. First, our quantification of omission and leakage related errors for different beam sizes suggests

TABLE VIII
WER BREAKDOWN ON THE AMI `test` SET FOR THE SDM MICROPHONE CONDITION, USING SURT *base* AND *large* MODELS.

Model	Adapt.	Ins.	Del.	Sub.	WER
Base	✗	0.8	55.8	8.8	65.4
	✓	1.8	32.3	12.8	46.9
Large	✗	0.9	52.0	9.7	62.5
	✓	2.1	29.8	12.8	44.6

that specialized decoding methods may be required which process all output branches simultaneously. For this, we can either extended the prefix beam search technique to process all output branches, or leverage HMM-based joint decoding techniques such as those proposed in [105]. In order to perform joint speaker diarization with SURT, we can predict word-level time-stamps and speaker change tokens, similar to ideas explored in [106]. Another interesting direction to exploit full dialog context may be cross-channel conversational rescoring using large language models [107]. We hope that our open-source implementation of SURT would encourage exploration along one or more of these directions.

ACKNOWLEDGMENTS

We thank Aparna Khare for discussions and insights on multi-turn RNN-T, and Piotr Zelasko, Fangjun Kuang, and Thilo von Neumann for help with Lhotse, Icefall, and `meeteval`, respectively. We are grateful to the anonymous reviewers for their feedback.

REFERENCES

[1] D. Amodei *et al.*, "Deep speech 2 : End-to-end speech recognition in english and mandarin," in *ICML*, 2016.
[2] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Toward human parity in conversational speech recognition," *IEEE/ACM TASLP*, vol. 25, pp. 2410–2423, 2017.
[3] J. Li, "Recent advances in end-to-end automatic speech recognition," *APSIPA Transactions on Signal and Information Processing*, 2021.
[4] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *IEEE ASRU*, 2015.
[5] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, A. Sehr, W. Kellermann, and R. Maas, "The REVERB challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *IEEE WASPAA*, 2013.
[6] S. Watanabe, M. Mandel, J. Barker, and E. Vincent, "CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," *ArXiv*, 2020.
[7] J. Carletta *et al.*, "The AMI meeting corpus: A pre-announcement," in *MLMI*, 2005.
[8] E. Shriberg, A. Stolcke, and D. Baron, "Observations on overlap: findings and implications for automatic processing of multi-party conversation," in *InterSpeech*, 2001.
[9] T. Yoshioka, D. Dimitriadis, A. Stolcke, W. Hinthorn, Z. Chen, M. Zeng, and X. Huang, "Meeting transcription using asynchronous distant microphones," in *InterSpeech*, 2019.
[10] J. G. Fiscus, J. Ajot, and J. S. Garofolo, "The rich transcription 2007 meeting recognition evaluation," in *CLEaR*, 2007.
[11] T. Hain, L. Burget *et al.*, "Transcribing meetings with the AMIDA systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, pp. 486–498, 2012.
[12] M. V. Segbroeck, A. F. A. Zaid, K. Kutsenko, C. Huerta, T. Nguyen, X. Luo, B. Hoffmeister, J. Trmal, M. Omologo, and R. Maas, "DiPCo - dinner party corpus," in *InterSpeech*, 2019.
[13] D. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM TASLP*, vol. 26, pp. 1702–1726, 2017.
[14] D. Raj, P. Denisov *et al.*, "Integration of speech separation, diarization, and recognition for multi-speaker meetings: System description, comparison, and analysis," in *IEEE SLT*, 2021.
[15] J. Wu, Z. Chen, S. Chen, Y. Wu, T. Yoshioka, N. Kanda, S. Liu, and J. Li, "Investigation of practical aspects of single channel speech separation for ASR," in *InterSpeech*, 2021.
[16] D. Yu, X. Chang, and Y. Qian, "Recognizing multi-talker speech with permutation invariant training," in *InterSpeech*, 2017.
[17] Y. Qian, X. Chang, and D. Yu, "Single-channel multi-talker speech recognition with permutation invariant training," *Speech Communication*, vol. 104, pp. 1–11, 2017.
[18] H. Seki, T. Hori, S. Watanabe, J. Le Roux, and J. R. Hershey, "A purely end-to-end system for multi-speaker speech recognition," in *ACL*, 2018.
[19] N. Kanda, Y. Gaur, X. Wang, Z. Meng, and T. Yoshioka, "Serialized output training for end-to-end overlapped speech recognition," in *InterSpeech*, 2020.
[20] A. Graves, "Sequence transduction with recurrent neural networks," in *ICML Representation Learning Workshop*, 2012.
[21] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *IEEE ICASSP*, 2019.
[22] C. Wu, Y. Wang, Y. Shi, C.-F. Yeh, and F. Zhang, "Streaming Transformer-Based Acoustic Models Using Self-Attention with Augmented Memory," in *InterSpeech*, 2020.
[23] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving RNN transducer modeling for end-to-end speech recognition," in *IEEE ASRU*, 2019.
[24] N. Kanda, J. Wu, Y. Wu, X. Xiao, Z. Meng, X. Wang, Y. Gaur, Z. Chen, J. Li, and T. Yoshioka, "Streaming multi-talker ASR with token-level serialized output training," in *InterSpeech*, 2022.
[25] L. Lu, N. Kanda, J. Li, and Y. Gong, "Streaming end-to-end multi-talker speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 803–807, 2020.
[26] I. Sklyar, A. Piunova, and Y. Liu, "Streaming multi-speaker ASR with RNN-T," in *IEEE ICASSP*, 2021.
[27] D. Raj, L. Lu, Z. Chen, Y. Gaur, and J. Li, "Continuous streaming multi-talker ASR with dual-path transducers," in *IEEE ICASSP*, 2022.
[28] I. Sklyar, A. Piunova, X. Zheng, and Y. Liu, "Multi-turn RNN-T for streaming recognition of multi-party speech," in *IEEE ICASSP*, 2021.
[29] L. Lu, N. Kanda, J. Li, and Y. Gong, "Streaming multi-talker speech recognition with joint speaker identification," in *InterSpeech*, 2021.
[30] L. Lu, J. Li, and Y. Gong, "Endpoint detection for streaming end-to-end multi-talker ASR," in *IEEE ICASSP*, 2022.
[31] I. Sklyar, A. Piunova, and C. Osendorfer, "Separator-transducer-segmenter: Streaming recognition and segmentation of multi-party speech," in *InterSpeech*, 2022.
[32] X. Chang, Y. Qian, and D. Yu, "Monaural multi-talker speech recognition with attention mechanism and gated convolutional networks," in *InterSpeech*, 2018.
[33] ——, "Adaptive permutation invariant training with auxiliary information for monaural multi-talker speech recognition," in *IEEE ICASSP*, 2018.
[34] T. Tan, Y. Qian, and D. Yu, "Knowledge transfer in permutation invariant training for single-channel multi-talker speech recognition," in *IEEE ICASSP*, 2018.
[35] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.
[36] L. Lu, X. Zhang, and S. Renals, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition," in *IEEE ICASSP*, 2016.
[37] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *NIPS*, 2015.
[38] C. Chiu, T. Sainath *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *IEEE ICASSP*, 2018.
[39] X. Chang, W. Zhang, Y. Qian, J. L. Roux, and S. Watanabe, "End-to-end multi-speaker speech recognition with transformer," in *IEEE ICASSP*, 2020.
[40] P. Denisov and N. T. Vu, "End-to-end multi-speaker speech recognition using speaker embeddings and transfer learning," in *InterSpeech*, 2019.
[41] W. Zhang, X. Chang, Y. Qian, and S. Watanabe, "Improving end-to-end single-channel multi-talker speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1385–1394, 2020.
[42] Y. Lin, Z. Du, S. Zhang, F. Yu, Z. Zhao, and F. Wu, "Separate-to-recognize: Joint multi-target speech separation and speech recognition for speaker-attributed ASR," in *ISCSLP*, 2022.
[43] X. Chang, Y. Qian, K. Yu, and S. Watanabe, "End-to-end monaural multi-speaker ASR system without pretraining," in *IEEE ICASSP*, 2019.
[44] J. Shi, X. Chang, S. Watanabe, and B. Xu, "Train from scratch: Single-stage joint training of speech separation and recognition," *Computer, Speech, and Language*, vol. 76, p. 101387, 2022.
[45] T. von Neumann, K. Kinoshita, L. Drude, C. Boeddeker, M. Delcroix, T. Nakatani, and R. Haeb-Umbach, "End-to-end training of time domain audio separation and recognition," in *IEEE ICASSP*, 2019.
[46] H. Inaguma, M. Mimura, and T. Kawahara, "Enhancing monotonic multihead attention for streaming ASR," in *InterSpeech*, 2020.
[47] M. Li, S. Zhang, C. Zorila, and R. Doddipatla, "Transformer-based streaming ASR with cumulative attention," in *IEEE ICASSP*, 2022.
[48] E. Tsunoo, Y. Kashiwagi, T. Kumakura, and S. Watanabe, "Transformer ASR with contextual block processing," in *IEEE ASRU*, 2019.
[49] Y. Z. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-channel multi-speaker separation using deep clustering," in *InterSpeech*, 2016.
[50] N. Kanda, Y. Gaur *et al.*, "Joint speaker counting, speech recognition, and speaker identification for overlapped speech of any number of speakers," in *InterSpeech*, 2020.
[51] N. Kanda, X. Chang, Y. Gaur, X. Wang, Z. Meng, Z. Chen, and T. Yoshioka, "Investigation of end-to-end speaker-attributed ASR for continuous multi-talker recordings," in *IEEE SLT*, 2020.
[52] N. Kanda, J. Wu, Y. Wu, X. Xiao, Z. Meng, X. Wang, Y. Gaur, Z. Chen, J. Li, and T. Yoshioka, "Streaming speaker-attributed ASR with token-level speaker embeddings," in *InterSpeech*, 2022.

[53] N. Kanda, J. Wu, X. Wang, Z. Chen, J. Li, and T. Yoshioka, "Vararray meets t-SOT: Advancing the state of the art of streaming distant conversational speech recognition," in *IEEE ICASSP*, 2023.

[54] N. Kanda, G. Ye, Y. Wu, Y. Gaur, X. Wang, Z. Meng, Z. Chen, and T. Yoshioka, "Large-scale pre-training of end-to-end multi-talker ASR for meeting transcription with single distant microphone," in *InterSpeech*, 2021.

[55] F. Yu, S. Zhang, Y. Fu, L. Xie, S. Zheng, Z. Du, W. Huang, P. Guo, Z. Yan, B. Ma, X. Xu, and H. Bu, "M2MeT: The ICASSP 2022 multi-channel multi-party meeting transcription challenge," in *IEEE ICASSP*, 2021.

[56] F. Yu, S. Zhang, P. Guo, Y. Fu, Z. Du, S. Zheng, W. Huang, L. Xie, Z. Tan, D. Wang, Y. Qian, K.-A. Lee, Z. Yan, B. Ma, X. Xu, and H. Bu, "Summary on the ICASSP 2022 multi-channel multi-party meeting transcription grand challenge," in *IEEE ICASSP*, 2022.

[57] Z. Chen, T. Yoshioka, L. Lu, T. Zhou, Z. Meng, Y. Luo, J. Wu, and J. Li, "Continuous speech separation: Dataset and analysis," in *IEEE ICASSP*, 2020.

[58] J. Wu, Z. Chen, J. Li, T. Yoshioka, Z. Tan, E. Lin, Y. Luo, and L. Xie, "An end-to-end architecture of online multi-channel speech separation," in *InterSpeech*, 2020.

[59] 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.

[60] S. Chen, Y. Wu, Z. Chen, J. Li, C. Wang, S. Liu, and M. Zhou, "Continuous speech separation with conformer," in *IEEE ICASSP*, 2021.

[61] X. Wang, D. Wang, N. Kanda, S. E. Eskimez, and T. Yoshioka, "Leveraging real conversational data for multi-channel continuous speech separation," in *InterSpeech*, 2022.

[62] Z. Chen, N. Kanda, J. Wu, Y. Wu, X. Wang, T. Yoshioka, J. Li, S. Sivasankaran, and S. E. Eskimez, "Speech separation with large-scale self-supervised learning," in *IEEE ICASSP*, 2022.

[63] K. Kinoshita, L. Drude, M. Delcroix, and T. Nakatani, "Listening to each speaker one by one with recurrent selective hearing networks," in *IEEE ICASSP*, 2018.

[64] Y. Zhang, Z. Chen, J. Wu, T. Yoshioka, P. Wang, Z. Meng, and J. Li, "Continuous speech separation with recurrent selective attention network," in *IEEE ICASSP*, 2021.

[65] 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*, 2021.

[66] ——, "Segment-less continuous speech separation of meetings: Training and evaluation criteria," *IEEE/ACM TASLP*, vol. 31, pp. 576–589, 2023.

[67] Z.-Q. Wang, P. Wang, and D. Wang, "Multi-microphone complex spectral mapping for utterance-wise and continuous speech separation," *IEEE/ACM TASLP*, vol. 29, pp. 2001–2014, 2020.

[68] T. Yoshioka, Z. Chen, C. Liu, X. Xiao, H. Erdogan, and D. Dimitriadis, "Low-latency speaker-independent continuous speech separation," in *IEEE ICASSP*, 2019.

[69] Z.-Q. Wang and D. Wang, "Localization based sequential grouping for continuous speech separation," in *IEEE ICASSP*, 2021.

[70] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: dataset, task and baselines," in *InterSpeech*, 2018.

[71] C. Boeddeker, J. Heitkaemper, J. Schmalenstroeer, L. Drude, J. Heymann, and R. Haeb-Umbach, "Front-end processing for the CHiME-5 dinner party scenario," in *CHiME Workshop*, 2018.

[72] S. Horiguchi, Y. Fujita, and K. Nagamatsu, "Block-online guided source separation," in *IEEE SLT*, 2021.

[73] D. Raj, D. Povey, and S. Khudanpur, "GPU-accelerated guided source separation for meeting transcription," in *InterSpeech*, 2023.

[74] N. Kanda, C. Böddeker, J. Heitkaemper, Y. Fujita, S. Horiguchi, K. Nagamatsu, and R. Häb-Umbach, "Guided source separation meets a strong ASR backend: Hitachi/paderborn university joint investigation for dinner party ASR," in *InterSpeech*, 2019.

[75] A. Arora, D. Raj, A. S. Subramanian, K. Li, B. Ben-Yair, M. Maciejewski, P. Żelasko, L. P. García-Perera, S. Watanabe, and S. Khudanpur, "The JHU multi-microphone multi-speaker ASR system for the CHiME-6 challenge," in *The CHiME Workshop*, 2020.

[76] I. Medennikov, M. Korenevsky, T. Prisyach, Y. Y. Khokhlov, M. Korenevskaya, I. Sorokin, T. Timofeeva, A. Mitrofanov, A. Andrusenko, I. Podluzhny, A. Laptev, and A. Romanenko, "The STC system for the CHiME-6 challenge," in *CHiME Workshop*, 2020.

[77] C. Boeddeker, T. Cord-Landwehr, T. von Neumann, and R. Haeb-Umbach, "An initialization scheme for meeting separation with spatial mixture models," in *InterSpeech*, 2022.

[78] K. Žmolíková, M. Delcroix, D. Raj, S. Watanabe, and J. H. Cernocký, "Auxiliary loss function for target speech extraction and recognition with weak supervision based on speaker characteristics," in *InterSpeech*, 2021.

[79] A. Sivaraman, S. Wisdom, H. Erdogan, and J. R. Hershey, "Adapting speech separation to real-world meetings using mixture invariant training," in *IEEE ICASSP*, 2021.

[80] F. Yu, S. Zhang, P. Guo, Y. Liang, Z. Du, Y. Lin, and L. Xie, "MFCCA: Multi-frame cross-channel attention for multi-speaker ASR in multi-party meeting scenario," in *IEEE SLT*, 2022.

[81] A. L. Janin, D. Baron, J. Edwards, D. P. W. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters, "The ICSI meeting corpus," in *IEEE ICASSP*, 2003.

[82] C. W. Fox, Y. Liu, E. Zwyssig, and T. Hain, "The sheffield wargames corpus," in *InterSpeech*, 2013.

[83] Y. Fu, L. Cheng, S. Lv, Y. Jv, Y. Kong, Z. Chen, Y. Hu, L. Xie, J. Wu, H. Bu, X. Xu, J. Du, and J. Chen, "AISHELL-4: An open source dataset for speech enhancement, separation, recognition and speaker diarization in conference scenario," in *InterSpeech*, 2021.

[84] Z. Yang, Y. Chen, L. Luo, R. Yang, L. Ye, G. Cheng, J. Xu, Y. Jin, Q. Zhang, P. Zhang, L. Xie, and Y. Yan, "Open source magicdata-RAMC: A rich annotated mandarin conversational speech dataset," in *InterSpeech*, 2022.

[85] H. Sak, M. Shannon, K. Rao, and F. Beaufays, "Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping," in *InterSpeech*, 2017.

[86] A. Tripathi, H. Lu, H. Sak, and H. Soltau, "Monotonic recurrent neural network transducer and decoding strategies," in *IEEE ASRU*, 2019.

[87] N. Moritz, F. Seide, D. Le, J. Mahadeokar, and C. Fuegen, "An investigation of monotonic transducers for large-scale automatic speech recognition," in *IEEE SLT*, 2022.

[88] J. Mahadeokar, Y. Shangguan, D. Le, G. Keren, H. Su, T. Le, C. feng Yeh, C. Fuegen, and M. L. Seltzer, "Alignment restricted streaming recurrent neural network transducer," in *IEEE SLT*, 2021.

[89] F. Kuang, L. Guo, W. Kang, L. Lin, M. Luo, Z. Yao, and D. Povey, "Pruned RNN-T for fast, memory-efficient ASR training," in *InterSpeech*, 2022.

[90] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-path RNN: Efficient long sequence modeling for time-domain single-channel speech separation," in *IEEE ICASSP*, 2019.

[91] D. Povey, https://github.com/k2-fsa/icefall/blob/master/egs/librispeech/ASR/pruned_transducer_stateless7/zipformer.py.

[92] M. R. Ghodsi, X. Liu, J. A. Apfel, R. Cabrera, and E. Weinstein, "RNN-Transducer with stateless prediction network," in *IEEE ICASSP*, 2020.

[93] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *IEEE ICASSP*, 2016.

[94] Y. Sudo, M. H. Shakeel, B. Yan, J. Shi, and S. Watanabe, "4D ASR: Joint modeling of CTC, attention, transducer, and mask-predict decoders," in *InterSpeech*, 2023.

[95] F. Landini, A. Lozano-Diez, M. Díez, and L. Burget, "From simulated mixtures to simulated conversations as training data for end-to-end neural diarization," in *InterSpeech*, 2022.

[96] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *IEEE ICASSP*, 2015.

[97] X. A. Miró, C. Wooters, and J. Hernando, "Acoustic beamforming for speaker diarization of meetings," *IEEE TASLP*, vol. 15, pp. 2011–2022, 2007.

[98] S. Renals and P. Swietojanski, "Neural networks for distant speech recognition," in *4th Joint Workshop on Hands-free Speech Communication and Microphone Arrays (HSCMA)*, 2014.

[99] T. von Neumann, C. Boeddeker, K. Kinoshita, M. Delcroix, and R. Haeb-Umbach, "On word error rate definitions and their efficient computation for multi-speaker speech recognition systems," in *IEEE ICASSP*, 2023.

[100] Z. Huang, D. Raj, L. P. García-Perera, and S. Khudanpur, "Adapting self-supervised models to multi-talker speech recognition using speaker embeddings," in *IEEE ICASSP*, 2023.

[101] C. J. Steinmetz and J. D. Reiss, "pyloudnorm: A simple yet flexible loudness meter in python," in *150th AES Convention*, 2021.

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

[103] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *InterSpeech*, 2019.

[104] C. Boeddeker, A. S. Subramanian, G. Wichern, R. Haeb-Umbach, and J. L. Roux, "TS-SEP: Joint diarization and separation conditioned on estimated speaker embeddings," *ArXiv*, vol. abs/2303.03849, 2023.

[105] M. Kocour, K. Žmolíková, L. Ondel, J. Svec, M. Delcroix, T. Ochiai, L. Burget, and J. H. Cernocký, "Revisiting joint decoding based multi-talker speech recognition with DNN acoustic model," in *InterSpeech*, 2021.

[106] W. Xia, H. Lu, Q. Wang, A. Tripathi, I. Lopez-Moreno, and H. Sak, "Turn-to-diarize: Online speaker diarization constrained by transformer transducer speaker turn detection," in *IEEE ICASSP*, 2021.

[107] L. Xu, Y. Gu, J. Kolehmainen, H. Khan, A. Gandhe, A. Rastrow, A. Stolcke, and I. Bulyko, "RescoreBERT: Discriminative speech recognition rescoring with BERT," in *IEEE ICASSP*, 2022.