Received 20 January 2023, accepted 3 February 2023, date of publication 9 February 2023, date of current version 14 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3243690

RESEARCH ARTICLE

Streaming End-to-End Target-Speaker Automatic Speech Recognition and Activity Detection

TAKAFUMI MORIYA[1,2], (Member, IEEE), HIROSHI SATO[1], TSUBASA OCHIAI[1], (Member, IEEE), MARC DELCROIX[1], (Senior Member, IEEE), AND TAKAHIRO SHINOZAKI[2], (Member, IEEE)
[1]NTT Corporation, Yokosuka, Kanagawa 239-0847, Japan
[2]Tokyo Institute of Technology, Yokohama, Kanagawa 226-8502, Japan

Corresponding author: Takafumi Moriya (takafumi.moriya.nd@hco.ntt.co.jp)

ABSTRACT Automatic speech recognition of a target speaker in the presence of interfering speakers remains a challenging issue. One approach to tackle this problem is target-speaker speech recognition, which conditions the recognition process on an embedding that characterizes the voice of the target speaker. This enables recognizing only the speech of the target speaker while ignoring interferences. In this work, we propose an end-to-end target-speaker speech recognition system based on a neural transducer architecture to allow streaming and on-device recognition. Moreover, a target-speaker speech recognition system should be able to detect when the target speaker is inactive and output nothing in such a case. We introduce training and decoding schemes to allow target-speaker activity detection within our proposed recognition system. We confirm experimentally that our proposed end-to-end system performs competitively to conventional cascade approaches of a target speech extraction module and a recognition module while reducing computation costs and allowing streaming decoding.

INDEX TERMS Automatic speech recognition, decoding, deep learning, interference, machine learning, neural networks, real-time systems, signal processing, speech processing, speech recognition.

I. INTRODUCTION

Automatic speech recognition (ASR) systems have been dramatically improved with the introduction of deep neural networks [1]. Conventional ASR systems, i.e., the Hidden Markov Model hybrid system, currently consist of several modules, including acoustic, lexicon, and language models. End-to-end (E2E) ASR, which can directly map acoustic features to output tokens, has recently attracted increasing interest from the ASR research community. Recent E2E ASR systems achieve high recognition performance for single-talker conditions [2], [3], [4], [5], [6], [7], [8], [9].

The size of E2E ASR systems is much smaller than that of conventional ASR systems because they do not need a statistical language model, a dominant factor determining computation cost and memory usage. Therefore, E2E ASR is also suited for on-device streaming applications requiring fast and accurate responses in real time. Recurrent neural network-transducer (RNNT) [10] is a promising technology for streaming E2E ASR applications and has been extensively investigated for single-speaker ASR [3], [4], [5], [6], [7], [8], [9].

However, in practice, the input to an ASR system often consists of a mixture of multi-speakers, such as the target speaker's speech mixed with interfering speakers' voices and background noise [11]. It remains challenging to recognize speech in such conditions. This paper addresses the challenging problem of recognizing a target speaker in a mixture, i.e., target-speaker ASR (TS-ASR), in a streaming manner to support applications such as voice search or user-dependent voice interactive systems. To realize streaming TS-ASR, we need to 1) identify the target speaker in the mixture, 2) recognize that speaker while ignoring other speakers, and 3) realize this process in a streaming manner.

We can realize TS-ASR by using a cascade of a target speech extraction (TSE) front-end with an ASR backend as in [12], [13], and [14]. TSE focuses on extracting a single target speaker by conditioning the process on a pre-recorded

The associate editor coordinating the review of this manuscript and approving it for publication was Gerard-Andre Capolino.

enrollment utterance of the target speaker. TSE realizes thus speaker identification and separation simultaneously. Cascade systems are modular, making it easy to visualize the different processing steps. However, although they can achieve high performance, this comes at the expense of very high computational costs.

Another approach consists of directly performing TS-ASR using an E2E system [15], [16], [17]. This can be realized by conditioning the recognition process on an enrollment utterance of the target speaker. Such approaches can get rid of the TSE front-end and its high computation costs. E2E ASR systems were first investigated for offline systems, where they conditioned the encoder of an attention-based encoder-decoder [15], [16], or an RNNT [17] on speaker embeddings. They demonstrated that such E2E TS-ASR systems could output the transcription of the target speaker while ignoring interference. However, these works relied on non-streaming and computationally intensive models.

We have recently extended these approaches to yield streaming E2E systems by including a similar mechanism in an RNNT [18]. We call this framework target-speaker RNNT (TS-RNNT). The system operates as follows. First, we compute a target speaker embedding from the enrollment utterance using a speaker encoder module. The speaker embedding is used as an auxiliary input to the encoder of RNNT to inform the system of which speaker is to be recognized in the mixture. We can incorporate the embedding within the RNNT encoder with a simple element-wise operation at an intermediate layer. Note that the embedding can be computed in advance. Therefore, our proposed TS-RNNT does not worsen the computational complexity or the latency of a vanilla RNNT model.

This paper extends our previous study in two directions. First, in [18], we reported a preliminary investigation where we implicitly assumed that the target speaker always exists in the input mixture. However, such an assumption does not always hold since the target speaker may sometimes be silent or absent. We call such a situation the inactive target speaker case, i.e., the enrollment utterance does not correspond to any of the speakers in the mixture. A TS-ASR system should be able to identify inactive target speaker cases and output nothing in such cases. In this paper, we introduce a novel target-speaker activity detection (TSAD)[1] framework within the TS-ASR systems and evaluate the TSAD performance of the TS-ASR framework. To the best of our knowledge, this is the first study to focus on the TSAD in the TS-ASR framework.

Second, [18] was a preliminary study on the effectiveness of Conformer-based TS-RNNT to show that it can achieve better recognition performance than the cascade system while keeping the inference speed of the vanilla RNNT. However, on-device applications may require more light-weight models such as long short-term memory (LSTM)-based models [19].

[1]Note that the problem TSAD tackles differ from target-speaker voice activity detection (TS-VAD), since we only detect if a target speaker is active or not in a mixture but not the fine starting and end time of the speaker utterance as TS-VAD does.

In this study, we investigate whether our proposed TS-RNNT framework is also effective for other light-weight types of streaming encoders such as unidirectional LSTM and latency-controlled bidirectional LSTM.

Our proposed system meets thus the requirement of a streaming ASR system, as it 1) can identify a target speaker in a mixture and detect target speaker inactivity, 2) output the transcription associated with the speech of the target speaker and nothing if it is inactive and 3) perform this process in a streaming manner and with lightweight models.

We conduct experiments to compare our proposed TS-RNNT with a cascade combination of TSE and RNNT (TSE+RNNT) for offline and streaming modes. Experiments show that our TS-RNNT is effective for LSTM- and Conformer-based models, and our proposed TS-RNNT matches the performance of TSE+RNNT in offline conditions at significantly lower computation cost. More importantly, it greatly outperforms a TSE+RNNT system in streaming mode. Moreover, our TSAD framework for TS-RNNT also equals the performance of the TSE module [20], [21].

To summarize, the main contributions of this paper are as follows.

- We present an E2E TS-ASR framework, i.e., TS-RNNT, which can recognize only the target speaker's voice in both offline and streaming manners. Especially, compared to prior E2E TS-ASR works as [12], [16], and [17], our proposed TS-RNNT framework is the first work that achieves streaming TS-ASR. Noticeably, it can perform TS-ASR while keeping the decoding speed of a vanilla RNNT.
- We also introduce a new decoding scheme that allows us to perform TSAD within our proposed TS-RNNT system. Our TSAD can detect when the target speaker is inactive and output nothing in that case. The TSAD process can also work in a streaming manner. Compared to prior works that perform TSAD in the TSE front-end [20], [21], it is the first work to propose an E2E TS-ASR that can simultaneously perform TSAD.
- We demonstrate that our target-speaker ASR matches the recognition and detection performance of conventional state-of-the-art cascade systems while significantly reducing computation costs and realizing practical streaming target-speaker ASR.
- Finally, we explore lightweight model configurations for our proposed TS-RNNT, which allows on-device processing.

In the remainder of the paper, we introduce related works in Section II. In Section III, we define the problem of TS-ASR and introduce the conventional cascade approach and our proposed E2E TS-ASR based on RNNT. In Section IV, we discuss the problem of TSAD and introduce a modified decoding scheme to perform TSAD. We then validate our proposed method experimentally in Section V. Finally, Section VI concludes our paper.

II. RELATED WORK

A. RELATION WITH SEPARATION-BASED APPROACH

Using speech separation instead of TSE or TS-ASR is another way to perform ASR when the input is a mixture of multiple speakers. Speech separation estimates every speaker's speech signal contained in the mixture. It does not perform speaker identification and would thus need to be enhanced with a speaker identification module to achieve TS-ASR.

Like TS-ASR, separation-based approaches have been used for both cascade and E2E systems. Some works, particularly related to our study, proposed streaming multi-talker ASR systems with RNNT [22], [23], [24]. In [22] and [23], they recognize the speech of all the speakers in a mixture; this is computationally demanding because the number of decoders equals the number of speakers in the mixture. Moreover, [22], [23], and [24] do not identify the speakers and thus cannot be used directly for TS-ASR. Arguably, it would be challenging to identify the target speakers when the output of the system is text. In [25] and [26], multi-talker ASR with speaker identification was proposed, but extra parameters and computations are needed for the speaker identification module. In contrast to [22], [23], and [24], our proposed TS-RNNT focuses only on the target speaker, which can significantly reduce the computational complexity when dealing with TS-ASR applications.

B. HANDLING OF INACTIVE SPEAKERS FOR TSE

There have been few works dealing with the inactive speaker issue of TSE [20], [21], [27], [28], [29]. Previous works [20], [27], [28], [29] have proposed a TSAD framework using the TSE model that determines whether the speaker in the input speech is the target speaker or not. The TSE-based TSAD uses two TSE outputs. One is a target speaker's embedding extracted from an enrollment speech of the target speaker. The other is a speaker embedding extracted from the output signal of TSE, i.e., the enhanced speech. The cosine similarity between these embeddings is used to detect the presence of the target speaker. The TSAD determines the target speaker is active if the cosine similarity is larger than a threshold. This TSAD does not need extra data or modification to the training scheme of a conventional TSE system. Unfortunately, this framework cannot be applied to TS-RNNT directly because it does not output enhanced speech. Thus, we propose a feasible but effective approach to realizing TSAD in TS-ASR.

III. TARGET-SPEAKER ASR (TS-ASR)

In this section, we introduce the TS-ASR systems used in our experiments. We first explain the TSE front-end and RNNT-based ASR back-end modules, which are the foundations of this work. Then we introduce our proposed TS-RNNT system.

Let $\boldsymbol{X} = [x_1, \ldots, x_{T'}] \in \mathcal{R}^{T'}$ be the single microphone input speech mixture of duration T'. Mixture $\boldsymbol{X}$ consists of the target speaker's speech $\boldsymbol{X}^{\text{target}}$ and $\boldsymbol{X}^{\text{interference}}$, which contains the interference speakers' speech and background

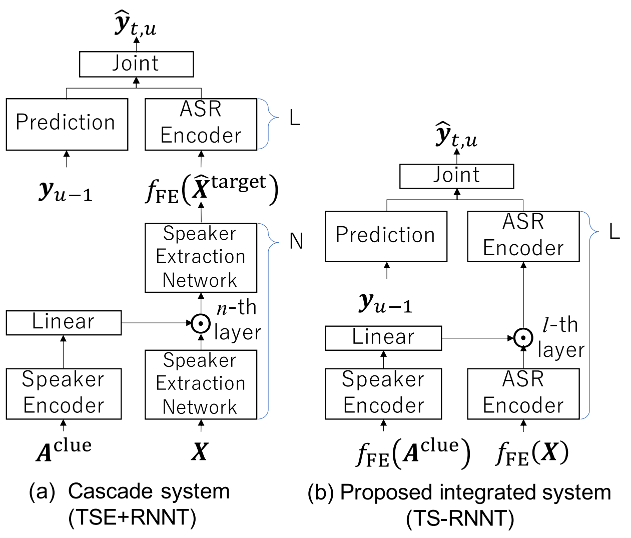


FIGURE 1. Overview of TS-ASR systems. The left (a) and right (b) figures illustrate the cascade system (conventional method) and the integrated system (proposed method), respectively.

noise. Note that we consider only a single interfering speaker in our experiments, but the derivation of the method does not depend on the number of interference speakers. $Y = [y_1, \ldots, y_U]$ is the transcription, i.e., the sequence of tokens of length U, associated with the utterance spoken by the target speaker, where $y_u \in \{1, \ldots, K\}$. K is the number of tokens in the vocabulary. Here tokens can be characters or subword units. For TS-ASR, we use enrollment speech $\boldsymbol{A}^{\text{clue}}$ that only contains the target speaker's speech for extracting the speaker information. Thus $\boldsymbol{A}^{\text{clue}}$ determines the target speaker.

A. OVERVIEW OF CASCADE TS-ASR (TSE+RNNT)

Fig. 1 (a) is a diagram of the cascade TS-ASR system composed of the TSE and ASR modules described below.

1) TSE FRONT-END

The TSE module extracts the target-speaker's speech, $\boldsymbol{X}^{\text{target}}$, from mixture $\boldsymbol{X}$ using enrollment speech $\boldsymbol{A}^{\text{clue}}$. We use the time-domain TSE system we introduced in [12], [13], and [14], which is similar to [30] and [31].

First, we compute the intermediate representation of the target-speaker, $\boldsymbol{H}^{\text{target}}$, from enrollment speech $\boldsymbol{A}^{\text{clue}}$ by using speaker encoder $f^{\text{Spk-Enc}}(\cdot)$, which consists of a multi-layer neural network followed by a linear layer. $\boldsymbol{H}^{\text{target}}$ is averaged into the embedding, $\boldsymbol{h}^{\text{target}}$, by a time-average pooling layer. Then, we use a speaker extraction network, $f^{\text{SE}}(\cdot)$, to extract the target speech given the embedding, $\boldsymbol{h}^{\text{target}}$. The above operations, which yield the estimated target speech signal $\hat{\boldsymbol{X}}^{\text{target}}$, are defined as follows:

$$\boldsymbol{H}^{\text{target}} = f^{\text{Spk-Enc}}(\boldsymbol{A}^{\text{clue}}; \theta^{\text{Spk-Enc}}), \tag{1}$$

$$\boldsymbol{h}^{\text{target}} = \frac{1}{T'} \sum_{t'=1}^{T'} \boldsymbol{h}_{t'}^{\text{target}}, \tag{2}$$

$$\hat{\boldsymbol{X}}^{\text{target}} = f^{\text{SE}}(\boldsymbol{X}, \boldsymbol{h}^{\text{target}}; \theta^{\text{SE}}). \tag{3}$$

In this work, we insert embedding $\boldsymbol{h}^{\text{target}}$ at the n-th layer of $f^{\text{SE}}(\cdot)$ by determining the Hadamard product between the embedding and the output of that layer. The parameters $\theta^{\text{TSE}} \triangleq [\theta^{\text{Spk-Enc}}, \theta^{\text{SE}}]$ are jointly learned by optimizing the source-to-distortion ratio (SDR) [32]. The training loss of the TSE module, $\mathcal{L}_{\text{TSE}}$, is thus the negative SDR defined as follows:

$$\mathcal{L}_{\text{TSE}} = -10\log_{10} \frac{\|\boldsymbol{X}^{\text{target}}\|^2}{\|\boldsymbol{X}^{\text{target}} - \hat{\boldsymbol{X}}^{\text{target}}\|^2}. \tag{4}$$

We can see that TSE module training requires parallel data consisting of speech mixture and the clean speech signal of the target speaker, $\boldsymbol{X}^{\text{target}}$. Given the difficulties of producing real recordings of mixtures and clean targets, we rely on simulated data for TSE system training.

2) ASR BACK-END

We adopt an RNNT-based ASR back-end [10] that can perform streaming ASR. RNNT learns the mapping between sequences of different lengths. It consists of an ASR encoder and a prediction network, which allows the posterior probabilities to be jointly conditioned on not only the ASR encoder outputs but also previous predictions. It proceeds as follows.

First, target speech signal $\boldsymbol{X}^{\text{target}}$ is transformed into a sequence of acoustic features, i.e., log Mel-filterbank, using a feature extractor $f^{\text{FE}}(\cdot)$. The features are then encoded into length-T sequence, $\boldsymbol{H}^{\text{ASR}} = \left[\boldsymbol{h}_1^{\text{ASR}}, \ldots, \boldsymbol{h}_T^{\text{ASR}}\right]$, via an ASR encoder network $f^{\text{ASR-Enc}}(\cdot)$. Next, the tokens, Y, are also encoded into $\boldsymbol{H}^{\text{Pred}} = \left[\boldsymbol{h}_1^{\text{Pred}}, \ldots, \boldsymbol{h}_U^{\text{Pred}}\right]$ via prediction network $f^{\text{Pred}}(\cdot)$. These two encoded features are passed to a feed-forward network, $f^{\text{Joint}}(\cdot)$, to compute the token posterior probabilities, $\hat{\boldsymbol{y}}_{t,u}$. The above operation is expressed as follows:

$$\boldsymbol{h}_t^{\text{ASR}} = f^{\text{ASR-Enc}}(f^{\text{FE}}(x_{t'}^{\text{target}}); \theta^{\text{ASR-Enc}}), \tag{5}$$
$$\boldsymbol{h}_u^{\text{Pred}} = f^{\text{Pred}}(y_{u-1}; \theta^{\text{Pred}}), \tag{6}$$
$$\hat{\boldsymbol{y}}_{t,u} = \text{Softmax}\left(f^{\text{Joint}}(\boldsymbol{h}_t^{\text{ASR}}, \boldsymbol{h}_u^{\text{Pred}}; \theta^{\text{Joint}})\right), \tag{7}$$

where Softmax$(\cdot)$ indicates a softmax operation. All learnable parameters, $\theta^{\text{RNNT}} \triangleq [\theta^{\text{ASR-Enc}}, \theta^{\text{Pred}}, \theta^{\text{Joint}}]$, are optimized by using RNNT loss and the forward-backward algorithm [10].

For training, we use target speech signals $\boldsymbol{X}^{\text{target}}$ corrupted with noise to attain a robust system; ground truth tokens Y are used for loss computation. At test time, we use the extracted signal estimated with TSE, $\hat{\boldsymbol{X}}^{\text{target}}$, and previous prediction $\hat{\boldsymbol{y}}_{t,u}$ of RNNT itself is fed to the prediction network. Note that we do not optimize the TSE and ASR modules jointly. This is because the joint training of TSE and ASR is required each time the TSE module is updated. Therefore, joint optimization is too expensive for practical use.

B. PROPOSED TARGET-SPEAKER RNNT (TS-RNNT)

In this paper, we propose an integrated modeling approach for TS-ASR using RNNT, which operates in a fully E2E manner, and yields streaming ASR.

1) TS-RNNT ARCHITECTURE

Fig. 1 (b) is a schematic diagram of our proposed TS-RNNT system. The TS-RNNT architecture is based on the vanilla RNNT described in III-A2, and we incorporate TSE essence into it. The difference is that the encoder of our TS-RNNT inputs the speech mixture directly and uses the target speaker embedding to inform which speaker in the mixture is to be decoded. This is performed in a similar process as the TSE front-end but within the encoder, i.e., a similar speaker encoder module and a fusion mechanism at an intermediate layer using the Hadamard product.

TS-RNNT encoder $f^{\text{ASR-Enc}'}(\cdot)$ is a modified version of ASR encoder $f^{\text{ASR-Enc}}(\cdot)$; it receives the speaker embedding extracted from speaker encoder $f^{\text{Spk-Enc}}(\cdot)$. The TS-RNNT encoder output $\boldsymbol{h}_t^{\text{ASR}'}$ is defined as follows:

$$\boldsymbol{H}^{\text{target}'} = f^{\text{Spk-Enc}'}(f^{\text{FE}}(\boldsymbol{A}^{\text{clue}}); \theta^{\text{Spk-Enc}'}), \tag{8}$$
$$\boldsymbol{h}^{\text{target}'} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{h}_t^{\text{target}'}, \tag{9}$$
$$\boldsymbol{h}_t^{\text{ASR}'} = f^{\text{ASR-Enc}'}(f^{\text{FE}}(x_{t'}), \boldsymbol{h}^{\text{target}'}; \theta^{\text{ASR-Enc}'}), \tag{10}$$

where $\boldsymbol{H}^{\text{target}'}$ with length-T is the speaker encoder outputs given input $\boldsymbol{A}^{\text{clue}}$ as the enrollment utterance, which is averaged on the time axis and embedded into $\boldsymbol{h}^{\text{target}'}$. $\boldsymbol{h}^{\text{target}'}$ and l-th layer intermediate output of the ASR encoder are multiplied using the Hadamard product in $f^{\text{ASR-Enc}'}(\cdot)$. Therefore, the prediction and joint networks have the same architecture as the vanilla RNNT in III-A2. All networks with learnable parameters $\theta^{\text{TS-RNNT}} \triangleq [\theta^{\text{Spk-Enc}'}, \theta^{\text{ASR-Enc}'}, \theta^{\text{Pred}}, \theta^{\text{Joint}}]$ are jointly optimized by using RNNT loss.

When decoding with the TS-RNNT, we first need to register in advance the speaker embedding, $\boldsymbol{h}^{\text{target}'}$, which is extracted from the enrollment utterance using the speaker encoder. Then, to decode the target speech from a mixture, we compute the acoustic features of the mixture signal, $f^{\text{FE}}(x_{t'})$, and input them with the speaker embedding $\boldsymbol{h}^{\text{target}'}$, to the encoder. finally, the ASR decoder outputs the ASR results $\hat{Y}$. Therefore, TS-RNNT can perform TS-ASR faster than the cascade system since there is no external extraction module. Moreover, we can train TS-RNNT using only the target-speaker's transcription Y, unlike the TSE, which requires the clean speech signal of the target speaker. This may facilitate training on real multi-speaker mixture recordings since, arguably, it is easier to transcribe than collect clean speech associated with a real mixture.

2) STREAMING TARGET-SPEAKER ASR BY TS-RNNT

In order to extend TS-RNNT into a streaming model, we replace the full-context ASR encoder with a left-to-right encoding module, i.e., causal encoder, as in [3] and [6]. Note that we do not need to change the speaker encoder module streaming mode because it only needs to operate once to register the target-speaker's clues $\boldsymbol{h}^{\text{target}'}$ in advance. The performance of current TSE front-ends tends to degrade severely when operating in streaming mode. On the

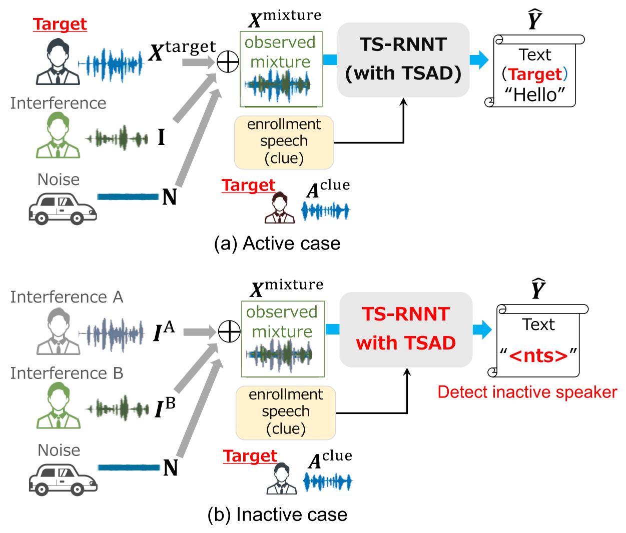


FIGURE 2. **Overview of TS-RNNT with target-speaker activity detection (TSAD). Top (a) and bottom (b) figures illustrate active and inactive cases, respectively.**

other-hand, streaming-type ASR models can well model speech sequences. As we set the TSE functionality within the RNNT encoder, we expect superior performance over the cascade models.

In addition, tuning a cascade system is more involving than the integrated system because the trade-offs between recognition performance and latency of each module, i.e., TSE and ASR, must be separately tuned. On the other hand, the integrated modeling, i.e., TS-RNNT, needs to tune just the ASR encoder to address the trade-off. Thus TS-RNNT would be easier to tune than cascade systems.

IV. TARGET-SPEAKER ACTIVITY DETECTION (TSAD)

TS-ASR, i.e., TS-RNNT, only recognizes the target speaker's speech from the mixture, as depicted in Fig. 2 (a). In terms of user experience and speech privacy, the users do not want TS-ASR to recognize other speakers' speech for the inactive speaker cases. Fig. 2 (b) shows the desired behavior of a TS-ASR system for inactive speaker case; TS-RNNT ignores the interference speakers' voices and outputs the non-target speaker label "<nts>", which indicates the absence of the target speaker in the mixture. We call this framework, which determines whether the target speaker's speech is contained in the input speech or not, TSAD. TSAD is essential for practical use-cases of TS-ASR. In the following subsections, we explain a conventional TSAD using TSE and our proposed approach that performs TSAD within the TS-RNNT system.

A. TSAD USING TSE

For cascade systems, we can use a simple TSAD framework using a TSE system [21]. First, the auxiliary network computes a speaker embedding by applying the speaker encoder to extracted speech $\boldsymbol{h}^{\hat{X}^{\text{target}}}$ since we showed in prior works that this can extract discriminative speaker embeddings [33]. We then make the TSAD decision by looking at the cosine similarity between the embeddings computed from the enrollment and the extracted speech as follows,

$$c^{\text{Cosine}} = \begin{cases} 1 & \text{if } f^{\text{Cosine}}(\boldsymbol{h}^{\text{target}}, \boldsymbol{h}^{\hat{X}^{\text{target}}}) > \eta \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

where $f^{\text{Cosine}}(\boldsymbol{h}^{\text{target}}, \boldsymbol{h}^{\hat{X}^{\text{target}}})$ is cosine similarity between $\boldsymbol{h}^{\text{target}}$ and $\boldsymbol{h}^{\hat{X}^{\text{target}}}$, and η is a threshold. c^{Cosine} means active case if it is 1, and otherwise inactive case. Therefore, this simple TSAD simply verifies whether it matches the enrollment characteristics or not.

B. TS-RNNT WITH INTERNAL TSAD

The above TSAD approach can detect inactive speaker relatively well. However, it cannot be applied to TS-RNNT because TS-RNNT does not explicitly generate the enhanced speech $\hat{X}^{\text{target}}$, and thus TS-RNNT cannot compute the cosine similarity as in (11). To solve this problem, we propose an alternative TSAD framework for TS-RNNT, which performs TSAD during decoding. To this end, we introduce a non-target speaker label "<nts>" that represents whether the target speaker's speech is not present in the input mixture. TS-RNNT can implicitly model TSAD using "<nts>" by including it in training data, and thus it can also handle the inactive speaker case. We explain the modifications of the training and decoding steps to realize the TS-RNNT with the TSAD framework in the following subsections.

1) TRAINING

In the training step, our TSAD approach requires only a simple modification of the conventional training scheme. The training data for TS-ASR consists of mixtures, the target speaker's transcriptions, and the enrollment utterance of the target speaker. In TS-RNNT without the TSAD framework, a mixture always contains the target speaker's voice. The enrollment utterance always comes from one of the speakers in the input mixture. The system is trained with a loss function between the network output and the target speaker's transcription.

Our approach for TS-RNNT with the TSAD framework randomly replaces the target speaker's enrollment utterances with those from another speaker, which is not contained in the input mixture. The corresponding ground truth tokens are replaced with a single non-target speaker label, "<nts>". The percentage ρ of random replacement for each epoch is a hyperparameter. These replaced enrollment speech and ground truth sequence are used as speaker encoder input and the target of RNNT loss computation, respectively. Thus we expect that the TS-RNNT with an internal TSAD framework can directly determine inactive case with the output of the "<nts>" symbol, and thus no external module for TSAD is needed.

2) DECODING

In the decoding step, TS-RNNT trained with the above TSAD framework simultaneously recognizes the target speaker's speech in the input mixture and determines

TABLE 1. Data generation setup. The number of utterances in (a) is double that of (b) due to the single-speaker case of (b). Single-talker ASR system (RNNT) was trained using (a). (b) was used for TSE and TS-RNNT training. All experiments were performed on (c).

	dataset	mixture type	SIR [dB]	SNR [dB]	#speakers (train set / dev set)	#mixtures or utterances (train set / dev set)
(a)	training data for ASR backend (RNNT)	1 speaker and noise	-	0 - 20	3054 / 160	400,000 / 10,000
(b)	training data for TSE and TS-RNNT	2 speakers and noise	-5 - 5	0 - 20	3054 / 160	200,000 / 5,000
(c)	evaluation data	2 speakers and noise	-5 - 5	0, 5, 10, 15, 20	30	$6,000 \times 5 = 30,000$

Algorithm 1 Alignment-Length Synchronous Decoding [34] With Target-Speaker Activity Detection (ALSD-TSAD)

```
1:  function ALSD-TSAD($\boldsymbol{h}^{\text{ASR}}_{1:T}$, $U_{\max}$, $N_{\text{bs}}$, $N_{\text{best}}$, $\lambda$);
    Input: ASR encoder output, maximum output length, beam size, N-best size, threshold for TSAD
    Output: Most likely hypotheses
2:  $B = \{\phi, 1, \boldsymbol{s}_0\}$; $F = \{\}$; $V = \{\}$
3:  $\boldsymbol{h}^{\text{Pred}}_{\phi}, \boldsymbol{s}_{\phi} = f^{\text{Pred}}(\phi, \boldsymbol{s}_0)$
4:  for $i = 1, \ldots, T + U_{\max}$ do
5:      $A = \{\}$
6:      if $i \leq T$ then
7:          $\hat{\boldsymbol{y}}'_{t,u} = \text{Softmax}\left(f^{\text{Joint}}(\boldsymbol{h}^{\text{ASR}}_i, \boldsymbol{h}^{\text{Pred}}_{\phi})\right)$
8:          $V = V \cup \{(\text{<nts>}, \hat{\boldsymbol{y}}'_{t,u}(\text{<nts>}), \boldsymbol{s}_{\phi})\}$
9:      end if
10:     for $(y, \delta_{i-1}(y), \boldsymbol{s}_{u-1}) \in B$ do
11:         $u = |y|$
12:         $t = i - u + 1$
13:         if $t > T$ then
14:             continue
15:         end if
16:         $\boldsymbol{h}^{\text{Pred}}_u, \boldsymbol{s}_u = f^{\text{Pred}}(y_{u-1}, \boldsymbol{s}_{u-1})$
17:         $\hat{\boldsymbol{y}}_{t,u} = \text{Softmax}\left(f^{\text{Joint}}(\boldsymbol{h}^{\text{ASR}}_t, \boldsymbol{h}^{\text{Pred}}_u)\right)$
18:         $\delta_i(y) = \delta_{i-1}(y) * \hat{\boldsymbol{y}}_{t,u}[\phi]$
19:         $A = A \cup \{(y, \delta_i(y), \boldsymbol{s}_{u-1})\}$
20:         if $t == T$ then
21:             $F = F \cup \{(y, \delta_i(y))\}$
22:         end if
23:         for $k \in \mathcal{Y}$ do
24:             $\delta_i(y + k) = \delta_{i-1}(y) * \hat{\boldsymbol{y}}_{t,u}[k]$
25:             $A = A \cup \{(y + k, \delta_i(y + k), \boldsymbol{s}_u)\}$
26:         end for
27:     end for
28:     $B = \text{PruneAndRecombineHyps}(A)$ [: $N_{\text{bs}}$]
29: end for
30: $V_{\text{best}} = \text{Sorted}(V)$ [1]
31: if $\lambda \geq \delta_{\text{best}}(\text{<nts>}) \in V_{\text{best}}$ then
32:     return Sorted($F$) [: $N_{\text{best}}$] #Default ALSD
33: else
34:     return $V_{\text{best}}$ #Determine inactive speaker
35: end if
```

whether the mixture contains the target speaker's speech. We add TSAD functionality to our decoding algorithm, which is based on alignment-length synchronous decoding (ALSD) [34], [35]. Algorithm 1 shows the ALSD algorithm, where we emphasize in red the modifications introduced to enable TSAD. In the following, we first describe the original ALSD algorithm and then explain our modifications to support TSAD functionality, which is referred to as ALSD-TSAD.

The ALSD algorithm operates decoding along the length of $T + U_{\max}$, where $U_{\max}$ is a hyperparameter that is an estimate of the maximum output sequence length. For decoding, we prepare two hypothesis sets A and B. The hypothesis $B = \{y, \delta, \boldsymbol{s}\}$ is initialized with blank symbol ϕ, its score 1, and the prediction network state $\boldsymbol{s}_0$. At the i-th iteration of the for-loop, output token sequence y and its score are computed for each hypothesis of B that contains token sequence y, its score $\delta_{i-1}(y)$, and its prediction network state $\boldsymbol{s}_{u-1}$. First, the blank hypothesis is added to A with current token sequence y, its score $\delta_{i-1}(y)$, which is updated by the blank transition probability denoted as $\hat{\boldsymbol{y}}_{t,u}[\phi]$, and its state $\boldsymbol{s}_{u-1}$. Note that the hypothesis is also added to the final hypothesis F if t reaches T. Then token k, its score, and current state $\boldsymbol{s}_u$ in token set $\mathcal{Y}$, which is a vocabulary set excluding ϕ, are added to A. Next, we prune and merge duplicate token sequences in A; this yields B which is reduced to beam size N_{bs} for the next step $i + 1$. Finally, at the end of the procedure after reaching $T + U_{\max}$, the default ALSD returns the N-best hypotheses in F sorted in descending score.

ALSD-TSAD is delineated by the red lines in Algorithm 1. First, we introduce an additional hypothesis set V for TSAD (in line 2) and preliminarily compute the prediction network output $\boldsymbol{h}^{\text{Pred}}_{\phi}$ and its state $\boldsymbol{s}_{\phi}$ for blank token ϕ (in line 3). For each step i in the for-loop, we compute the output posterior probability $\hat{\boldsymbol{y}}'_{t,u}$ conditioned on the prediction network output $\boldsymbol{h}^{\text{Pred}}_{\phi}$ (in line 7), and then add a hypothesis to V with non-target speaker token "<nts>", its probability $\hat{\boldsymbol{y}}_{t,u}$[<nts>], and its prediction network state $\boldsymbol{s}_{\phi}$ (in line 8). Note that a non-target speaker token probability $\hat{\boldsymbol{y}}_{t,u}$(<nts>) is computed for every time frame i. Finally, we extract the highest hypothesis V_{best} from V sorted in descending probability $\hat{\boldsymbol{y}}_{t,u}$[<nts>] (in line 30). Our modified ASLD-TSAD algorithm returns the N-best hypotheses if δ_{best}(<nts>) is larger than threshold hyperparameter λ (in line 32), otherwise it returns the non-target speaker hypothesis V_{best} (in line 34). Therefore, we simultaneously perform TSAD during the decoding process while performing the original ALSD. As we can see, the TSAD framework is independent of ALSD, and thus it can be easily implemented with other decoding algorithms.

V. EXPERIMENTS

A. EXPERIMENTAL SETUPS

1) DATA

We evaluated our proposal on the simulated two-speaker mixtures with background noise. The speech recordings were taken from the Corpus of Spontaneous Japanese (CSJ) [36], sampled at 16 kHz. The noise recordings were taken from CHiME-3 corpus [37], which consists of real noise recordings recorded on the bus, cafe, pedestrian area, and street junction. The signal-to-noise (SNR) ratio of the mixtures was set between 0 and 20 dB. The overlap ratio of the input mixture was about 89% on average for both training and evaluation sets.

We used almost the same data as in previous works [12], [13], [14], and the difference is that the amount of data was increased. The details are shown in Table 1. The total training data amounts to 800 hours of speech. The training, development, and evaluation datasets consisted of different speakers. We evaluated performance in terms of character error rate (CER) for ASR and equal error rate (EER) for TSAD. We also measured inference speed in terms of real time factor (RTF: *decoding time/data time*). We measured the RTF using a Python implementation of the algorithm running on an Intel Xeon 2.40GHz CPU.

2) SYSTEM CONFIGURATION OF TSE MODULE

We adopted the time-domain SpeakerBeam as the structure of the TSE module [13], [31]. Time-domain SpeakerBeam is an extension of Conv-TasNet [32] to extract a target speaker's speech given an enrollment utterance. We used the open-source implementation of Conv-TasNet[2] as a basis for our implementation of time-domain SpeakerBeam. The hyperparameters of the speaker extraction network were set as follows: $N = 256$, $L = 20$, $B = 256$, $R = 4$, $X = 8$, $H = 512$ and $P = 3$ following the notation in [32]. The speaker encoder consisted of a 1-D convolutional encoder and a single layer of stacked 1-D convolutional blocks.

We also implemented a streaming/causal TSE model as an extension of the time-domain SpeakerBeam. We replaced every convolution function of the TSE model with causal convolution functions and replaced global layer normalization (LN) with channel-wise LN to satisfy causality. The algorithmic latency of the streaming TSE model is 1.25ms ($= 20\text{samples}/16k$), which is negligible with regard to the latency of ASR decoding. All TSE models were trained with the dataset (b) in Table 1.

3) SYSTEM CONFIGURATION OF ASR MODULE

The input features for ASR models were 80-dimensional log Mel-filterbank coefficients. We used SpecAugment [38] during training. In this paper, we adopted 3262 characters as the output tokens. The training and evaluation data were preprocessed following the Kaldi and ESPnet toolkits [35], [39], [40]. The minibatch size was set to 64 in all experiments. In this work, we adopted LSTM-based and Conformer-based encoders as detailed below and briefly described the resulting architecture in Fig. 3.

Our LSTM-based experiments investigated three different encoder types: unidirectional LSTM (ULSTM), bidirectional LSTM (BLSTM), and latency-controlled

[2]https://github.com/funcwj/conv-tasnet

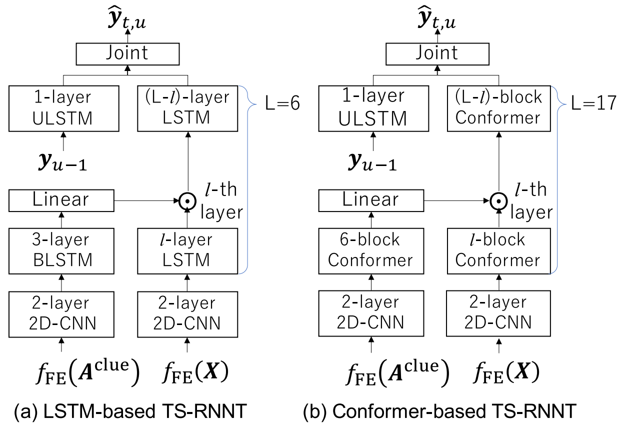


FIGURE 3. Overview of TS-RNNT configurations. (a) and (b) illustrate LSTM- and Conformer-based TS-RNNT, respectively.

BLSTM (LC-BLSTM) [41]. All encoders had two-layer 2D-convolutional neural networks (CNNs) followed by six-layer (LC-) BLSTM with 512 cells per direction. The CNNs in all layers had max-pooling layers with two strides. The memory cells were doubled when using the ULSTM encoder. The chunk sizes for LC-BLSTM, N_c and N_r, were set to 60 in the input feature space before CNNs. The average latency of ULSTM- and LC-BLSTM-based systems were 30ms and 930ms, respectively. As the prediction network, we adopted a ULSTM layer with 768 cells. The joint network receives the output of the encoder and the prediction network, reduces their dimensions to 640, and then predicts the target token. The speaker encoder for LSTM-based TS-RNNT had two-layer 2D-CNNs followed by three-layer BLSTM with 512 cells per direction due to memory usage during training. ULSTM and BLSTM encoders were first pre-trained [7] using the AdaDelta optimizer [42] with early stopping; the parameters were then fine-tuned. In the fine-tuning step, learning rate scheduling is essential, so we used the Adam optimizer [43] with the learning rate scheduler [38]. For the LC-BLSTM models, we initialized the parameters with those of a BLSTM model and then fine-tuned them.

For the Conformer-based experiments, we used an offline conformer consisting of the same encoder architecture as Conformer (L) [2] with a kernel size of 15. We used two-layer 2D-CNNs followed by 17 Conformer blocks, with the stride sizes of both max-pooling layers at each layer set to 2. The prediction network had a 768-dimension uni-directional LSTM (ULSTM) layer followed by a 640-dimension feed-forward layer. The speaker encoder for TS-RNNT had the same architecture as the ASR encoder, but the number of blocks was reduced from 17 to 6. The model parameters of the offline Conformer were randomly initialized.

For the streaming systems, we used two similar configurations as the offline system except that the encoder was replaced by a causal Conformer (Uni-Conformer) [3] and a latency-controlled Conformer (LC-Conformer) [6]. For all the streaming systems, we replaced the depth-wise convolution and batch normalization of the ASR encoders with causal equivalent and LN, respectively. The streaming

TABLE 2. CER as a function of the layer-l at which the speaker encoder outputs were fused with the ASR encoder. The results are for offline TS-RNNTs. "All" indicates that the fusion was applied to all layers of the encoder.

Fusion in	CER [%] on each SNR					
layer-l	20dB	15dB	10dB	5dB	0dB	Avg.
1	**8.6**	**9.3**	**11.4**	**17.3**	**32.7**	**15.8**
5	9.8	10.4	12.5	18.0	33.1	16.8
1-5	9.3	9.8	11.6	17.6	32.9	16.4
All	9.7	10.2	12.5	18.9	35.1	17.3

TABLE 3. Comparisons of cascade and proposed TS-RNNT systems using LSTM and Conformer (Conf.) in terms of CERs and RTFs at each SNR. All models performed offline decoding. Note that RTFs exclude computation time of speaker encoder in TSE and TS-RNNT, since it can be performed offline.

System		CER [%] on each SNR [dB]						RTF
		20	15	10	5	0	Avg.	
LSTM	RNNT	88.6	87.8	87.2	88.2	92.7	88.9	**0.42**
	TSE+RNNT	**11.4**	**13.3**	**17.8**	28.9	50.8	**24.4**	1.24
	TS-RNNT	13.5	14.8	18.2	**27.5**	**48.3**	24.5	**0.42**
Conf.	RNNT	76.0	75.7	75.4	75.7	78.0	76.1	**0.40**
	TSE+RNNT	**7.9**	**8.8**	**11.1**	18.3	36.3	16.5	1.22
	TS-RNNT	8.6	9.3	11.4	**17.3**	**32.7**	**15.8**	**0.40**

systems were trained using attention masks, as in [6]. The latency of the Uni-Conformer with infinite history and few look-ahead frames, i.e., CNN module, was 30ms. The left and current chunk sizes of LC-Conformer were set to ∞/680ms and 600ms, respectively, so the average latency was 330ms ($=$ 600ms$/2$ $+$ 30ms). Here ∞ means that the encoder sees all past frames. The model parameters of the streaming Conformer parameters were initialized with those of an offline Conformer.

"RNNT" and "TS-RNNT" were trained with the dataset (a) and (b) in Table 1, respectively. All models were trained with RNNT loss by using the Adam optimizer with 25k warmup for a total of 100 epochs. For training the TS-RNNT with TSAD, we set the percentage, ρ, of enrollment utterances replaced by an utterance not in the mixture at 5%. For decoding, we used alignment-length synchronous decoding with a beam size of 8 [34].

B. EXPERIMENTAL RESULTS

1) ABLATION STUDY

First, we investigate the best position for the fusion of the ASR encoder and the speaker encoder outputs using offline Conformer-based TS-RNNTs. Table 2 shows the CERs at each SNR. We can see that applying the speaker encoder output to just the first ASR encoder output, $l = 1$, attained the best performance under all SNR conditions. Hereafter, we adopted $l = 1$ for all TS-RNNT models.

2) CASCADE VS. INTEGRATED SYSTEM USING OFFLINE MODELS

Next, we compare the baseline cascade systems, TSE+RNNT, with the proposed integrated system, TS-RNNT, for offline decoding. The comparisons were performed using LSTM- and Conformer-based (TS-) RNNT. Table 3 shows the CERs and RTFs for each SNR condition. The RNNT baseline shows the CER obtained when recognizing the mixture without performing TSE using the RNNT back-end. As expected, this system performs poorly as it cannot identify the target speaker in a mixture. By using the TSE model as front-end, TSE+RNNT using LSTM- and Conformer-based encoders could recognize the target-speaker's speech and achieve CER of 24.4% and 16.5% on average, respectively. This result confirms the importance of the TSE module. However, the RTF of TSE+RNNT was much larger than that of RNNT decoding. We could reduce the RTF of the TSE+RNNT system by adopting a smaller TSE model such as [44], but this would increase the CER. Regardless of how efficient this TSE front-end could be, it would inevitably increase the RTF.

On the other hand, both LSTM- and Conformer-based TS-RNNT achieved competitive or better CERs than the cascade system while equaling the RTF of RNNT. Our proposed framework, i.e., TS-RNNT, is effective for both LSTM- and Conformer-based systems, and the CERs of Conformer-based TS-RNNT were better than those of the LSTM-based system. Moreover, in particular, the CERs of TS-RNNT are better than those of the cascade system under severely noisy conditions, i.e., SNR 5 and 0 dB. This is probably because the TS-RNNT system is trained in an E2E manner, and thus is not affected by processing artifacts that may limit the performance of the TSE+RNNT system under severe noise conditions. We could improve the performance of the TSE+RNNT cascade system by jointly training both modules or retraining the ASR back-end on processed speech [17]. However, the RTF of such a system would remain higher than our proposed TS-RNNT.

3) CASCADE VS. INTEGRATED SYSTEM USING STREAMING MODELS

We investigated the effectiveness of TS-RNNT for streaming models using both LSTM and Conformer. Table 4 shows the CERs under each SNR condition. "ASR-Enc type" indicates the type of streaming LSTM or Conformer, and "#frames" is the number of look-back and look-ahead frames for the ASR encoders of the RNNT and TS-RNNT models. Note that we investigated two types of LC-Conformer; LC-Conformer with infinite history context and LC-Conformer with a historical context limited to 71 frames. A "✓" and "✗" in the "Streaming" column indicates that the module is operating in streaming mode or offline mode, respectively. All systems are fully streaming systems except system ID L1, L4, C1, C4, and C7, which operate offline due to the TSE front-end. The SDR improvements in offline and streaming TSE models were 15.1dB and 11.1dB, respectively, which mirrors the tendency reported in prior separation studies [32].

From the table, we observe that the proposed streaming TS-RNNT models (systems C3, C6 and C9) offer comparable performance to cascade TSE+RNNT with the offline TSE module (systems C1, C4 and C7). On the other hand, the performance of LSTM-based TS-RNNT models (systems L3 and L6) were worse than those of cascade TSE+RNNT with the offline TSE module (systems L1 and L4), and self-attention-based model, i.e., Conformer, is superior compared

TABLE 4. Comparisons of cascade and proposed TS-RNNT systems. "✓" and "✗" indicate streaming and offline mode, respectively. Systems that have a "✓" for the "Streaming All" column attained fully streaming inference. TS-RNNT does not use any TSE ("N/A"). Boldfonts indicate the best performance for fully streaming systems for each type of ASR encoder.

ID	ASR-Enc type	#frames of look-		System	Streaming			CER [%] on each SNR					
		back	ahead		TSE	ASR	All	20dB	15dB	10dB	5dB	0dB	Avg.
L1	ULSTM	∞	3	TSE+RNNT	✗	✓	✗	14.7	16.9	22.2	34.1	55.3	28.6
L2					✓	✓	✓	22.6	26.1	34.4	50.4	71.3	41.0
L3				TS-RNNT	N/A	✓	✓	**19.7**	**21.2**	**25.7**	**36.5**	**57.7**	**32.2**
L4	LC-BLSTM	∞	63	TSE+RNNT	✗	✓	✗	12.0	13.2	16.4	24.8	43.9	22.1
L5					✓	✓	✓	17.4	19.2	23.9	35.2	55.2	30.2
L6				TS-RNNT	N/A	✓	✓	**14.0**	**15.4**	**19.2**	**28.9**	**50.6**	**25.6**
C1	Uni-Conformer	∞	3	TSE+RNNT	✗	✓	✗	11.3	12.5	16.0	24.6	43.6	21.6
C2					✓	✓	✓	16.9	18.9	24.1	36.2	56.8	30.6
C3				TS-RNNT	N/A	✓	✓	**13.1**	**14.2**	**17.3**	**25.1**	**42.8**	**22.5**
C4	LC-Conformer	∞	[0, 63]	TSE+RNNT	✗	✓	✗	10.0	11.0	14.2	22.5	41.6	19.9
C5					✓	✓	✓	15.3	17.1	21.8	32.9	53.5	28.1
C6				TS-RNNT	N/A	✓	✓	**11.4**	**12.4**	**15.0**	**21.9**	**38.6**	**19.8**
C7	LC-Conformer′	71	[0, 63]	TSE+RNNT	✗	✓	✗	11.0	12.3	15.8	24.7	44.2	21.6
C8					✓	✓	✓	16.8	18.6	23.8	35.6	56.3	30.2
C9				TS-RNNT	N/A	✓	✓	**11.9**	**13.0**	**16.0**	**23.8**	**41.2**	**21.2**

to LSTM even for TS-RNNT models. Both LSTM- and Conformer-based TS-RNNT systems significantly outperform streaming cascade systems, i.e., TSE+RNNT with the streaming TSE module (systems L2, L5, C2, C5 and C8) for all five encoder types. Comparing the performance of the offline TS-RNNT system of Table 3 and the best streaming system of Table 4 (system C6), we observe a relative performance gap of about 20%, which may be slightly worse than the gap observed for tasks with clean speech [9]. Closing this gap will be part of our future work.

The LC-Conformer with infinite lookback context (system C6) achieves the best overall performance with a latency of just 63 frames, i.e., 330 ms. Note that the performance of the LC-conformer′ degrades when we limit the lookback context (system C7, C8, and C9) toward practical use, which indicates that the future context is important, but so is the past context when processing overlapped speech.

These results demonstrate the effectiveness of the integrated system for streaming decoding using both LSTM- and Conformer-based TS-RNNT models, as it avoids the performance degradation caused by the streaming TSE module and achieves performance competitive with the offline TSE module while matching the inference speed of a vanilla RNNT. In both offline and streaming experiments, Conformer-based TS-RNNT systems had much better CERs than LSTM-based systems. The results of Table 4 show that the Conformers model outperforms the ULSTM and LC-BLSTM models. However, LSTM-based models are easy to carry over long histories, i.e., the hidden state, for long-form speech, making it lightweight in terms of memory usage. Consequently, although there is a recognition performance degradation, LSTM-based RNNT systems could be advantageous for on-device processing. Hereafter, we adopted Conformer-based TS-RNNT models for TSAD experiments.

4) TARGET-SPEAKER ACTIVITY DETECTION RESULT

Finally, we evaluated the TSAD capability of the proposed TS-RNNT. In this experiment, we prepared 6,000 × 5 additional evaluation samples to simulate inactive speaker cases. We used the same mixtures as in the previous experiments but randomly selected enrollment utterances from speakers, not in the mixtures. For the active speaker cases, we selected the target speaker as the speaker contained in the input mixture as in the previous experiments. In this experiment, we also considered the inactive speaker cases where the target speaker is inactive for the whole mixture. Based on the active and inactive speaker samples, we evaluated the performance of TSAD.

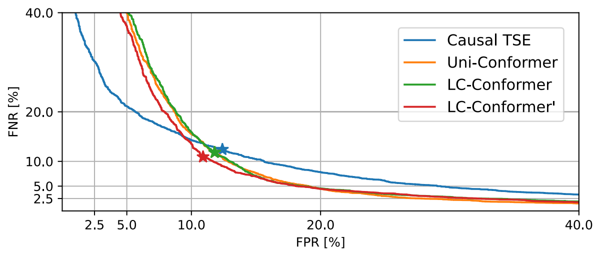
First, we evaluated the impact of training with TSAD on ASR performance for the active examples. Table 5 lists the CERs of each TS-RNNT system trained without and with the proposed TSAD framework in IV-B1. Here, to eliminate the effect of the TSAD errors, decoding is conducted without

TABLE 5. CERs of each system without ("-") and with ("✓") TSAD training. Note that threshold λ in Algorithm 1 was not set, and we measured the upper bound of CERs. Each system without TSAD training corresponds to offline TS-RNNT models in Table 3, and streaming TS-RNNT models in Table 4.

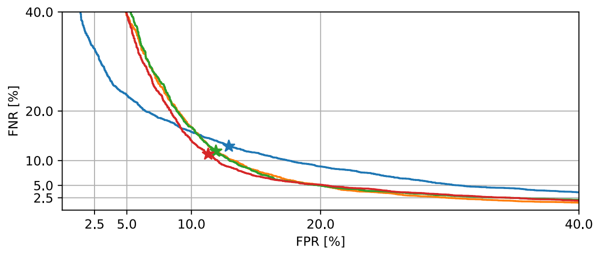
System	TSAD training	CER [%] on each SNR					
		20dB	15dB	10dB	5dB	0dB	Avg.
Conformer	-	8.6	9.3	11.4	17.3	32.7	15.8
	✓	8.8	9.4	**11.2**	**16.9**	**31.9**	**15.7**
Uni-Conformer	-	13.1	14.2	17.3	25.1	42.8	22.5
	✓	13.2	14.3	17.4	25.7	44.6	23.0
LC-Conformer	-	11.4	12.4	15.0	21.9	38.6	19.8
	✓	11.6	12.6	15.1	22.0	38.8	20.0
LC-Conformer′	-	11.9	13.0	16.0	23.8	41.2	21.2
	✓	12.0	13.1	16.0	23.9	41.4	21.3

TABLE 6. Comparisons of TSE and proposed TS-RNNT systems for TSAD. "✓" and "✗" in full streaming (fs) column indicate streaming and offline mode, respectively.

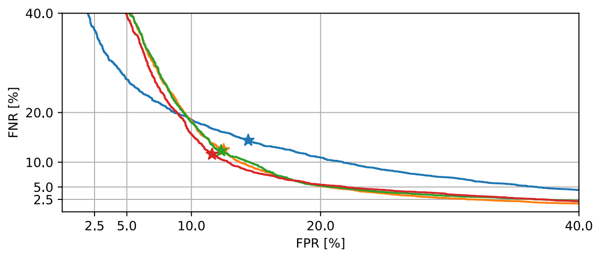
System	fs	EER [%] on each SNR [dB]					
		20dB	15dB	10dB	5dB	0dB	Avg.
TSE	✗	9.1	9.3	10.0	12.4	15.8	11.3
Causal TSE	✓	12.4	12.9	14.4	17.0	21.2	15.6
Conformer	✗	12.7	13.0	13.0	14.4	16.6	14.0
Uni-Conformer	✓	11.8	11.9	12.5	**13.6**	17.2	13.4
LC-Conformer	✓	11.8	11.9	12.3	**13.6**	**16.9**	**13.2**
LC-Conformer′	✓	**10.9**	**11.3**	**11.6**	13.9	18.1	13.3



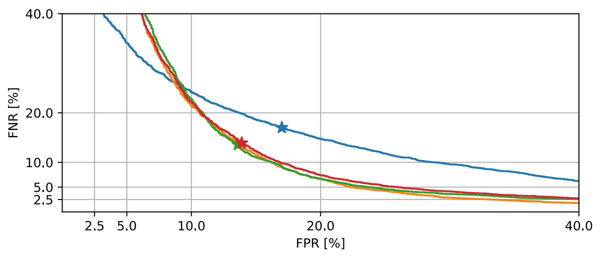
(a) 20dB



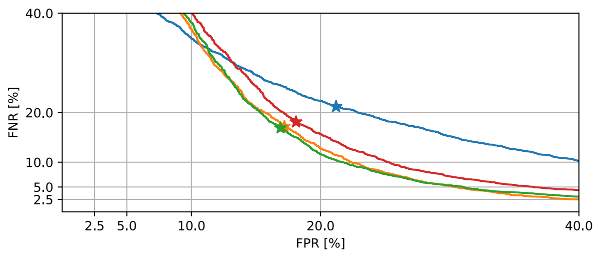
(b) 15dB



(c) 10dB



(d) 5dB



(e) 0dB

FIGURE 4. DET curves for TSAD with streaming TSE or TS-RNNT on each SNR. Horizontal and vertical axes mean false positive rate (FPR) and false negative rate (FNR), resectively. Each plot "⋆" indicates corresponding EERs in Table 6.

TSAD functionality by setting the TSAD threshold $\lambda = 0$ in Algorithm 1 (i.e., the system always outputs the hypothesis). These systems with "-" mean TS-RNNT without TSAD training, and the results are the same as in Table 3 and 4. We can see that the systems with TSAD training achieved comparable CERs to those without TSAD training, and the TSAD training framework did not degrade the ASR performance of TS-RNNT.

Next, we evaluated the TSAD performance of the proposed TS-RNNT with TSAD. Table 6 shows the EERs under each SNR condition using TSE models and TS-RNNT models with TSAD; we varied threshold hyperparameters η for TSE as in (11), and λ for TS-RNNT as in Algorithm 1. "✓" and "✗" in the "fs" column indicates that the module is operating in streaming mode and offline mode, respectively. Note that these Conformer models in Table 6 were trained with the TSAD framework introduced in Section IV-B1, the same as the evaluated systems in Table 5.

From the table, we observe that the offline TSE model offers better results in terms of EERs than the streaming TSE model. For TSE-based systems, TSAD is performed by computing the cosine similarity between speaker embeddings derived from the enrollment utterance and the extracted speech signal. Thus, the EERs of TSE-based approaches heavily depend on the quality of the extracted speech. Consequently, EERs increase with lower input SNRs.

For TS-RNNT systems with TSAD, the EERs of the streaming model are better than those of causal TSE-based TSAD under all SNR conditions. Moreover, compared to the causal TSE-based models, EERs of TS-RNNT with TSAD degrade much less with lower SNRs. The EERs of streaming TS-RNNT systems are better than those of offline systems. The results are similar regardless of the type of encoder. We argue that our proposal, i.e., streaming TS-RNNT with TSAD, achieves competitive TSAD performance with those of TSE while keeping the model size reasonable.

Fig. 4 (a)-(e) plot the detection error tradeoff (DET) curves for streaming TSE-based TSAD system and TSAD within streaming TS-RNNT models under each SNR condition, and each plot "⋆" means corresponding EERs in Table 6. We observe that all streaming TSAD within TS-RNNT models exhibits very similar DET curves. The tendency for TSE-based TSAD and TSAD within TS-RNNT models is relatively different. Indeed, looking at the curves, by modifying the operating point (threshold η for TSE-based TSAD), we may achieve lower FPR (wrongly outputting transcription for inactive speakers) with TSE-based TSAD. In contrast, we may achieve lower FNR (or missed target speaker when they are active) with TSAD in TS-RNNT models. However, in all conditions, TSAD within TS-RNNT models achieves lower EER than streaming cascade approaches.

These results demonstrate the effectiveness of the TSAD framework for streaming TS-RNNT, as it achieves superior TSAD performance to streaming TSE while keeping the TS-ASR performance. Moreover, as we can see in Algorithm 1, our TSAD framework can keep the decoding speed of default ALSD, with the difference being that the joint network computation times are increased by T.

VI. CONCLUSION

We have proposed an integrated modeling approach for streaming TS-ASR, called TS-RNNT. It can recognize the speech of the target speaker in a mixture by fusing a target-speaker embedding derived from the enrollment speech with the intermediate outputs of the RNNT encoder. Our proposed system offers comparable performance to a cascade TSE+RNNT system in the offline setting, with significantly lower complexity. Indeed, our system retained the low complexity of a basic RNNT. Moreover, it greatly outperforms cascade systems in the streaming mode for LSTM- and Conformer-based TS-RNNT model configurations. It is the first work that proposes a streaming E2E TS-ASR system.

We discussed the importance of performing TSAD for TS-ASR. We proposed a TSAD scheme that can be implemented with an existing TS-RNNT, with a slight modification of its training and decoding scheme. The proposed approach can detect inactive speakers with an EER of about 13%, which is competitive with cascade TSE-ASR models. The results of this work confirm the potential of TS-RNNT with internal TSAD to achieve robust streaming ASR against interfering speakers.

Future works will include further improving the recognition performance, particularly for larger SNR conditions (e.g., SNR larger than 15 dB), where performance remains slightly worse than that of a cascade system. Moreover, we will investigate knowledge distillation [45] from a large model to a small one to boost the performance of compact models for on-device applications.

REFERENCES

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[2] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, Oct. 2020, pp. 5036–5040.

[3] T. N. Sainath, Y. He, A. Narayanan, R. Botros, R. Pang, D. Rybach, C. Allauzen, E. Variani, J. Qin, Q.-N. Le-The, S.-Y. Chang, B. Li, A. Gulati, J. Yu, C.-C. Chiu, D. Caseiro, W. Li, Q. Liang, and P. Rondon, "An efficient streaming non-recurrent on-device end-to-end model with improvements to rare-word modeling," in *Proc. Interspeech*, Aug. 2021, pp. 1777–1781.

[4] G. Kurata and G. Saon, "Knowledge distillation from offline to streaming RNN transducer for end-to-end speech recognition," in *Proc. Interspeech*, Oct. 2020, pp. 2117–2121.

[5] B. Li, S.-Y. Chang, T. N. Sainath, R. Pang, Y. He, T. Strohman, and Y. Wu, "Towards fast and accurate streaming end-to-end ASR," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 6069–6073.

[6] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming transformer transducer for speech recognition on large-scale dataset," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 5904–5908.

[7] T. Moriya, T. Ashihara, T. Tanaka, T. Ochiai, H. Sato, A. Ando, Y. Ijima, R. Masumura, and Y. Shinohara, "SimpleFlat: A simple whole-network pre-training approach for RNN transducer-based end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 5664–5668.

[8] T. Moriya, T. Tanaka, T. Ashihara, T. Ochiai, H. Sato, A. Ando, R. Masumura, M. Delcroix, and T. Asami, "Streaming end-to-end speech recognition for hybrid RNN-T/attention architecture," in *Proc. Interspeech*, Aug. 2021, pp. 1787–1791.

[9] T. Moriya, T. Ashihara, A. Ando, H. Sato, T. Tanaka, K. Matsuura, R. Masumura, M. Delcroix, and T. Shinozaki, "Hybrid RNN-T/attention-based streaming ASR with triggered chunkwise attention and dual internal language model integration," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 8282–8286.

[10] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 2012, pp. 1–15.

[11] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Interspeech*, Sep. 2018, pp. 1561–1565.

[12] H. Sato, T. Ochiai, M. Delcroix, K. Kinoshita, N. Kamo, and T. Moriya, "Learning to enhance or not: Neural network-based switching of enhanced and observed signals for overlapping speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 6287–6291.

[13] H. Sato, T. Ochiai, M. Delcroix, K. Kinoshita, T. Moriya, and N. Kamo, "Should we always separate?: Switching between enhanced and observed signals for overlapping speech recognition," in *Proc. Interspeech*, Aug. 2021, pp. 1149–1153.

[14] H. Sato, T. Ochiai, M. Delcroix, K. Kinoshita, T. Moriya, N. Makishima, M. Ihori, T. Tanaka, and R. Masumura, "Strategies to improve robustness of target speech extraction to enrollment variations," in *Proc. Interspeech*, Sep. 2022, pp. 996–1000.

[15] P. Denisov and N. T. Vu, "End-to-end multi-speaker speech recognition using speaker embeddings and transfer learning," in *Proc. Interspeech*, Sep. 2019, pp. 4425–4429.

[16] M. Delcroix, S. Watanabe, T. Ochiai, K. Kinoshita, S. Karita, A. Ogawa, and T. Nakatani, "End-to-end SpeakerBeam for single channel target speech recognition," in *Proc. Interspeech*, Sep. 2019, pp. 451–455.

[17] J. Shi, C. Zhang, C. Weng, S. Watanabe, M. Yu, and D. Yu, "Improving RNN transducer with target speaker extraction and neural uncertainty estimation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6908–6912.

[18] T. Moriya, H. Sato, T. Ochiai, M. Delcroix, and T. Shinozaki, "Streaming target-speaker ASR with neural transducer," in *Proc. Interspeech*, Sep. 2022, pp. 2673–2677.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] C. Zhang, M. Yu, C. Weng, and D. Yu, "Towards robust speaker verification with target speaker enhancement," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6693–6697.

[21] M. Delcroix, K. Kinoshita, T. Ochiai, K. Zmolikova, H. Sato, and T. Nakatani, "Listen only to me! How well can target speech extraction handle false alarms?" in *Proc. Interspeech*, Sep. 2022, pp. 1–5.

[22] I. Sklyar, A. Piunova, and Y. Liu, "Streaming multi-speaker ASR with RNN-T," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6903–6907.

[23] L. Lu, N. Kanda, J. Li, and Y. Gong, "Streaming end-to-end multi-talker speech recognition," *IEEE Signal Process. Lett.*, vol. 28, pp. 803–807, 2021.

[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 *Proc. Interspeech*, Sep. 2022, pp. 3774–3778.

[25] L. Lu, N. Kanda, J. Li, and Y. Gong, "Streaming multi-talker speech recognition with joint speaker identification," in *Proc. Interspeech*, Aug. 2021, pp. 1782–1786.

[26] 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 *Proc. Interspeech*, Sep. 2022, pp. 521–525.

[27] W. Rao, C. Xu, E. S. Chng, and H. Li, "Target speaker extraction for multi-talker speaker verification," in *Proc. Interspeech*, Sep. 2019, pp. 1273–1277.

[28] M. Borsdorf, C. Xu, H. Li, and T. Schultz, "Universal speaker extraction in the presence and absence of target speakers for speech of one and two talkers," in *Proc. Interspeech*, Aug. 2021, pp. 1469–1473.

[29] Z. Zhang, B. He, and Z. Zhang, "X-TaSNet: Robust and accurate time-domain speaker extraction network," in *Proc. Interspeech*, Oct. 2020, pp. 1421–1425.

[30] C. Xu, W. Rao, E. S. Chng, and H. Li, "Time-domain speaker extraction network," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2019, pp. 327–334.

[31] M. Delcroix, T. Ochiai, K. Zmolikova, K. Kinoshita, N. Tawara, T. Nakatani, and S. Araki, "Improving speaker discrimination of target speech extraction with time-domain speakerbeam," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 691–695.
[32] Y. Luo and N. Mesgarani, "Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 27, no. 8, pp. 1256–1266, Aug. 2019.
[33] K. Zmolikova, M. Delcroix, K. Kinoshita, T. Ochiai, T. Nakatani, L. Burget, and J. Cernocky, "SpeakerBeam: Speaker aware neural network for target speaker extraction in speech mixtures," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 4, pp. 800–814, Aug. 2019.
[34] G. Saon, Z. Tuske, and K. Audhkhasi, "Alignment-length synchronous decoding for RNN transducer," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 7799–7803.
[35] F. Boyer, Y. Shinohara, T. Ishii, H. Inaguma, and S. Watanabe, "A study of transducer based end-to-end ASR with ESPnet: Architecture, auxiliary loss and decoding strategies," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2021, pp. 16–23.
[36] K. Maekawa, H. Koiso, S. Furui, and H. Isahara, "Spontaneous speech corpus of Japanese," in *Proc. Int. Conf. Lang. Res. Eval. (LREC)*, May 2000, pp. 947–952.
[37] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: Analysis and outcomes," *Comput. Speech Lang.*, vol. 46, pp. 605–626, Nov. 2017.
[38] 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 *Proc. Interspeech*, Sep. 2019, pp. 2613–2617.
[39] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlıcek, Y. Qian, P. Schwarz, J. Silovskı, G. Stemmer, and K. Veselı, "The Kaldi speech recognition toolkit," in *Proc. Workshop Autom. Speech Recognit. Understand.*, Dec. 2011, pp. 1–4.
[40] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech*, Sep. 2018, pp. 2207–2211.
[41] Y. Zhang, G. Chen, K. Yao, S. Khudanpur, J. Glass, and D. Yu, "Highway long short-term memory RNNs for distant speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jan. 2016, pp. 5755–5759.
[42] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*.
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, May 2015, pp. 1–15.
[44] Q. Wang, I. L. Moreno, M. Saglam, K. Wilson, A. Chiao, R. Liu, Y. He, W. Li, J. Pelecanos, M. Nika, and A. Gruenstein, "VoiceFilter-lite: Streaming targeted voice separation for on-device speech recognition," in *Proc. Interspeech*, Oct. 2020, pp. 2677–2681.
[45] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1–13.

TAKAFUMI MORIYA (Member, IEEE) received the M.E. degree from the Tokyo Institute of Technology, Tokyo, Japan, in 2016, where he is currently pursuing the Ph.D. degree. He is also a Researcher at NTT Corporation, Japan. His research interests include speech and spoken language processing, machine learning, and machine intelligence. He is a member of the Acoustical Society of Japan (ASJ). He was a recipient of the 12th Best Student Presentation Award of ASJ and the Awaya Prize Young Researcher Award from ASJ.

HIROSHI SATO received the B.E. and M.E. degrees from The University of Tokyo, in 2016 and 2018, respectively.

He is currently a Researcher at NTT Corporation, Japan. He joined NTT in 2018, where has been engaged in research, development, and practical application of speech-processing technologies. His research interests include speech enhancement, robust speech recognition, and speech dialog systems. He is a member of ASJ.

TSUBASA OCHIAI (Member, IEEE) received the Ph.D. degree in information engineering from Doshisha University, Kyoto, Japan, in 2018. He is currently a Research Scientist at NTT Corporation, Japan. His research interests include speech and spoken language processing, such as automatic speech recognition, array signal processing, and machine learning. He is a member of the Acoustic Society of Japan.

MARC DELCROIX (Senior Member, IEEE) received the M.Eng. degree from the Free University of Brussels, Brussels, Belgium, and the Ecole Centrale Paris, Paris, France, in 2003, and the Ph.D. degree from the Graduate School of Information Science and Technology, Hokkaido University, in 2007.

He is currently a Distinguished Researcher at the NTT Communication Science Laboratories, NTT Corporation, Japan. He was a Research Associate at NTT Communication Science Laboratories, from 2007 to 2008 and from 2010 to 2012, where he became a permanent Research Scientist, in 2012. He was a Visiting Lecturer at the Faculty of Science and Engineering, Waseda University, Tokyo, from 2015 to 2018. He took an active part in the development of NTT robust speech recognition systems for the REVERB, CHiME 1, and CHiME 3 challenges, which all achieved the best performance results in the tasks. His research interests include speech and audio processing, such as target speech and sound extraction, speech enhancement, robust speech recognition, model adaptation, and speaker diarization.

Dr. Delcroix is a member of ASJ. He is also a member of the IEEE Signal Processing (SP) Society and the Speech and Language Processing Technical Committee (SL-TC). He also served as a member of the organizing committee for the REVERB challenge 2014, the ASRU 2017, and the SLT 2022. He received the 2005 Young Researcher Award from the Kansai Section of the Acoustic Society of Japan (ASJ), the 2006 Student Paper Award from the IEEE Kansai Section, the 2006 Sato Paper Award from ASJ, the 2015 IEEE ASRU Best Paper Award Honorable Mention, and the 2016 ASJ Awaya Young Researcher Award.

TAKAHIRO SHINOZAKI (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in computer science from the Tokyo Institute of Technology, Tokyo, Japan, in 1999, 2001, and 2004, respectively. He is currently an Associate Professor at the Tokyo Institute of Technology. From 2004 to 2006, he was a Research Scholar at the Department of Electrical Engineering, University of Washington, Seattle. His research interests include semi-supervised and unsupervised learning of spoken languages, automatic spoken language acquisition, and their applications.