Streaming Multi-talker Speech Recognition with Joint Speaker Identification

Liang Lu, Naoyuki Kanda, Jinyu Li and Yifan Gong

Microsoft Corp., USA
{lili, naoyuki.kanda, jin, yifan.gong}@microsoft.com

Abstract

In multi-talker scenarios such as meetings and conversations, speech processing systems are usually required to transcribe the audio as well as identify the speakers for downstream applications. Since overlapped speech is common in this case, conventional approaches usually address this problem in a cascaded fashion that involves speech separation, speech recognition and speaker identification that are trained independently. In this paper, we propose Streaming Unmixing, Recognition and Identification Transducer (SURIT) – a new framework that deals with this problem in an end-to-end streaming fashion. SURIT employs the recurrent neural network transducer (RNN-T) as the backbone for both speech recognition and speaker identification. We validate our idea on the LibrispeechMix dataset – a multi-talker dataset derived from Librispeech, and present encouraging results.

Index Terms: Overlapped speech recognition, Streaming, Unmixing transducer, Joint recognition and identification

1. Introduction

In multi-talker scenarios such as meetings and conversations, speech processing systems are usually required to transcribe the audio as well as identify the speakers for downstream applications. Since overlapped speech is common in this case, conventional approaches usually address this problem in a cascaded fashion that involves speech separation, speech recognition and speaker identification that are trained independently. In this paper, we propose Streaming Unmixing, Recognition and Identification Transducer (SURIT) – a new framework that deals with this problem in an end-to-end streaming fashion. SURIT employs the recurrent neural network transducer (RNN-T) as the backbone for both speech recognition and speaker identification. We validate our idea on the LibrispeechMix dataset – a multi-talker dataset derived from Librispeech, and present encouraging results.

1. Introduction

In multi-talker scenarios such as meetings and conversations, speech processing systems are usually required to transcribe the audio as well as identify the speakers for downstream applications. Since overlapped speech is common in this case, conventional approaches usually address this problem in a cascaded fashion that involves speech separation, speech recognition and speaker identification that are trained independently. In this paper, we propose Streaming Unmixing, Recognition and Identification Transducer (SURIT) – a new framework that deals with this problem in an end-to-end streaming fashion. SURIT employs the recurrent neural network transducer (RNN-T) as the backbone for both speech recognition and speaker identification. We validate our idea on the LibrispeechMix dataset – a multi-talker dataset derived from Librispeech, and present encouraging results.

2. Related Work

There were limited studies on the joint modeling of end-to-end ASR and SID. One simple approach is inserting a speaker-role tag [27] or a speaker-identity tag [28] at the end of each utterance. However, [27] and [28] did not cope with overlapping speech and also their approach cannot identify a speaker who is not included in the training data. Recently, Kanda et al. [29] proposed an offline sequence-to-sequence (SST) based approach that can transcribe and identify the speakers from overlapping speech given a set of speaker profiles. Our work differs from [29] mainly in that SURIT applies the RNN-T for both ASR and SID, which can run in a streaming fashion. The idea of using an RNN-T for SID is inspired by [30], which proposed RNN-T based language identification. Our speaker verification module depends on a speaker inventory as an auxiliary information as same with [29]. Overall, our work is the first study on RNN-T based SID that is trained jointly with multi-talker ASR in a streaming fashion.

3. RNN-T: Review

RNN-T is the backbone in our model, which directly computes the sequence-level conditional probability of the label sequence given the acoustic features by marginalizing all the possible alignments. Given an acoustic feature sequence $X = \{x_1, \cdots, x_T\}$ and its corresponding label sequence $Y = \{y_1, \cdots, y_T\}$, an RNN-T can be trained to maximize the conditional probability $P(Y|X)$. The RNN-T model consists of an encoder and a decoder. The encoder is a recurrent neural network (RNN) that takes the acoustic features as input and produces a hidden representation. The decoder is also an RNN that takes the hidden representation and the previous output as input and produces the label sequence.

$$P(Y|X) = \sum_{Y'} P(Y'|X) P(Y'|Y) = \sum_{Y'} P(Y'|X) P(Y')$$

where $Y'$ is a possible alignment between $X$ and $Y$. The RNN-T model is trained to maximize the log-likelihood of the label sequence:

$$\log P(Y|X)$$

The output of the model is the sequence of label probabilities, which can be used for decoding to obtain the most likely label sequence $Y^*$. The RNN-T model has been shown to achieve good performance in various speech recognition tasks, including multi-talker speech recognition and speaker identification.

4. Methodology

Our proposal is the first attempt to jointly model end-to-end multi-talker ASR and SID. The overall framework consists of two main components: the unmixing module and the identification module. The unmixing module aims to separate the overlapping speech signals and produce clean speech representations for each speaker. The identification module then takes these clean speech representations and identifies the speakers accordingly. We introduce an asynchronous pipeline for our method, which includes the following steps:

- **Unmixing**: We apply an unmixing module to the overlapping speech signal to produce clean speech representations for each speaker. The unmixing module is trained to minimize the difference between the clean speech representations and the original overlapping speech signal.

- **Recognition**: We use the clean speech representations to perform speech recognition, which is a critical component in any speech processing system. The recognition module is trained to maximize the conditional probability of the label sequence given the clean speech representations.

- **Identification**: We use the clean speech representations to perform speaker identification, which is necessary for many downstream applications such as meeting transcription. The identification module is trained to minimize the difference between the actual and predicted speaker identities.

5. Experimental Setup

We evaluate our method on the LibrispeechMix dataset, which simulates the overlapped speech data from the LibriSpeech corpus. We compare our method to several baselines, including a simple method that uses RNN-T based ASR and SID, and a state-of-the-art method that uses a multi-task learning approach. Our method achieves a significant improvement in both speech recognition and speaker identification tasks, which demonstrates the effectiveness of our proposed framework.
Figure 1: An illustration of using RNN-T for streaming speaker identification. The arrowed path corresponds to \( \{ (bs) \} \), where \((bs)\) denotes the blank speaker token. The probability of the arc \( \rightarrow \) is \(b_{l,u}\), and the probability of the arc \( \Rightarrow \) is \((1 – b_{l,u})P(s | z_{l,u})\) in Eq. (6).

\[ \{y_1, \ldots, y_u\}, \] where \( T \) is the length of the acoustic sequence, and \( U \) is the length of the label sequence, RNN-T defines the conditional probability

\[ P(Y | X) = \sum_{Y \in B^{-1}(Y)} P(\tilde{Y} | X), \] (1)

where \( \tilde{Y} \) is a path that contains the blank token \( \emptyset \), and the function \( B \) denotes mapping the path to \( Y \) by removing the blank tokens in \( \tilde{Y} \). The probability can be efficiently computed by the forward-backward algorithm, which requires to compute the probability of each time step. Let \( f(\cdot) \) denote the audio encoder, and \( g(\cdot) \) the label encoder, the feature representations can be obtained as

\[ h_{1,t}^t = f(x_{1:t}), \quad h_{l,u}^t = g(y_{u-1:t}), \] where \( y_0 \) is a blank token. RNN-T fuses the two feature representations by a joint network \( j(\cdot) \) as

\[ z_{l,u} = j(h_{1,t}^t, h_{l,u}^t), \] (2)

and the conditional probability of the next token is obtained by a Softmax function, i.e.,

\[ P(y_{u} | z_{l,u}) = \text{Softmax}(W z_{l,u} + b), \] (3)

where \( W \) and \( b \) are the weight matrix and bias. Given \( P(y_{u} | z_{l,u}) \), the sequence-level conditional probability can be computed by dynamic programming, and the RNN-T loss can be defined as the negative log-likelihood as:

\[ \mathcal{L}_{\text{mx}}(Y, X) = - \log P(Y | X) \] (4)

4. Streaming SID with Transducers

In this paper, we propose to apply transducers for streaming SID. Here, we assume a single-talker audio to introduce the idea of transducer-based SID. It will be extended for the joint modeling with multi-talker ASR with speech overlap in Section 5.

Suppose we have a speaker inventory \( D = \{ d_k | k = 1, \ldots, K \} \), where \( K \) is the total number of the profiles, \( d_k \) is a speaker embedding (e.g., d-vector [31]) of the \( k \)-th speaker in the speaker inventory. Given an input acoustic feature \( X \), the goal of SID is estimating the speaker index \( k \) corresponding to \( X \). To simulate the real scenario, the speaker inventory \( D \) can be different for each audio segment \( X \), indicating that the participants of the meeting or conversation may vary each time. Standard SID approaches typically addresses this problem by extracting the vector representation of \( X \), and measure its distance against the \( d \)-vectors in \( D \) according to a metric such as the cosine distance. However, those approaches usually operate in an offline fashion and require a voice activity detector (VAD) if the audio is not segmented. On the other hand, in this work, we focus on streaming SID that can emit the speaker label before the utterance is ended, and can handle both silent and voiced regions without a VAD.

4.1. Transducer-based SID

For the proposed transducer-based SID, we treat each speaker index \( s \in \{1, \ldots, K\} \) as the token that should be predicted by the RNN-T, as illustrated in Fig. 1. To tailor the RNN-T model to fit in this scenario, we need to accommodate a few model structure changes. Since we do not capture the dependencies among the speaker labels, we omit the recurrent layers in the label encoder of RNN-T. To train such a model, we need to calculate the probabilities of emitting both the speaker label \( s \) and the blank token \( (bs) \). Given \( z_{l,u} \) from Eq (2), the probability of emitting the speaker label \( s \) can be represented as

\[ P(s = k | z_{l,u}) = \frac{\exp(d_k^T z_{l,u})}{\sum_{k=1}^K \exp(d_k^T z_{l,u})}. \] (5)

Where \( d_k \) is the \( d \)-vector of the target speaker. However, in the speaker inventory, there is no embedding vector for \((bs)\), so we cannot apply Eq. (5) to compute the probability of \( P((bs) | z_{l,u}) \). To deal with this problem, we apply a special type of RNN-T model, i.e., Hybrid Autoregressive Transducer (HAT) [32], which was originally proposed for external language model integration for RNN-T. The key idea of HAT is that it reformulates the token probability, and uses two distributions for blank and label tokens, i.e.,

\[ P(v_u | z_{l,u}) = \begin{cases} b_{l,u} & \text{ otherwise} \end{cases} \] (6)

where \( b_{l,u} \) is a Bernoulli distribution that can be estimated using a Sigmoid function, while \( P(s | z_{l,u}) \) is defined as Eq. (5). Note that the value space of \( s \) is \([1, K]\), while the value space of \( v_u \) is \([1, K] \cup \{ (bs) \} \). With HAT, we can compute the emission probability of \((bs)\) without its \( d \)-vector.

4.2. Optimizing the SID latency

Naively applying the RNN-T loss as Eq. (4) could cause the high latency of the HAT-based SID model. The reason is that the vanilla RNN-T loss does not encourage the model to emit the labels early. More critically, for each path in Fig. 1, the number of blank tokens is \( T - 1 \) while there is only one speaker label for an audio input of \( T \) frames. Consequently, the gradient of \((bs)\) is significantly larger than the gradient of speaker labels. This encourages the model to prefer emitting \((bs)\) during inference. To counteract this effect, we scale the gradient of \((bs)\) as

\[ \frac{\partial \mathcal{L}}{\partial b_{l,u}} = \alpha \frac{\partial \mathcal{L}}{\partial b_{l,u}} \quad \alpha \in (0, 1), \] (7)

which is very similar to FastEmit [33], but we scale the gradient of \((bs)\) since it is dominant. Inspired by the method to optimize the latency of end-pointer [34], another approach is to penalize the probability of emitting the speaker label late, i.e.,

\[ \log P(s | z_{l,u}) = \max(0, \beta(t – t_{\text{buffer}} – t_{\text{delay}})) \] (8)

where \( \beta \) is a scaling parameter, \( t_{\text{buffer}} \) is the grace period, and \( t_{\text{delay}} \) means the delay when we simulate the overlapped audio. This equation means that for a path in Fig. 1, if the time step \( t \) that emits the speaker label is larger than \( t_{\text{buffer}} + t_{\text{delay}} \), then the probability of this path will be penalized.

5. Joint Multi-talker ASR and SID

In a multi-talker scenario, dealing with the overlapped speech is the key challenge. In [24], we proposed a streaming end-to-end
model for multi-talker speech recognition, referred as SURT. The idea of SURT is illustrated in Figure 2-a) for the 2-speaker case, in which, the Unmixing module emits two streams of audio representation where each stream contains the feature representations of one speaker. In this work, we use convolutional neural networks (CNNs) for the Unmixing module, in particular

\[
M = \sigma(\text{CNN}_{\text{enc}}(X)), \quad H = \text{CNN}_{\text{enc}}(X),
\]

\[
H_1 = H \ast M, \quad H_2 = H - H_1,
\]

where \(\sigma\) denotes the Sigmoid function; \(\text{CNN}_{\text{mask}}\) and \(\text{CNN}_{\text{enc}}\) are two CNN encoders. The two feature representations are then fed into the same RNN-T network to compute the corresponding losses. It is well understood that label permutation is a key challenge for multi-talker speech processing [15]. Suppose the two ASR label sequences are \(Y^1\) and \(Y^2\), it is unclear if \(H_1\) corresponds to \(Y^1\) or \(Y^2\), and similar for \(H_2\). A widely used approach to address this problem is Permutation Invariant Training (PIT) [15], which considers all the possible label assignments. In [24], we evaluated another simple label assignment strategy, referred as Heuristic Error Assignment Training (HEAT) [23]. Unlike PIT, HEAT only considers one possible label assignment based on some heuristic information, for example, the order that they were spoken according to their starting times. Suppose \(Y^1\) is from the speaker who spoken first, HEAT defines the loss function as

\[
\mathcal{L}_{\text{asr}}(X, Y^1, Y^2) = \mathcal{L}_{\text{unmix}}(Y^1, H_1) + \mathcal{L}_{\text{unmix}}(Y^2, H_2),
\]

which means that we always assign the feature representation \(H_1\) to the first spoken speaker. The loss function can drive the network to learn the corresponding feature mapping. In [24], we show that HEAT works slightly better than PIT in terms of recognition accuracy, and it consumes less memory with lower computational cost.

Similarly to SURT, we can represent the SID network for multi-talker case as shown in Fig. 2-b). We can also define the speaker identification loss with HEAT as

\[
\mathcal{L}_{\text{sid}}(X, S^1, S^2) = \mathcal{L}_{\text{sid}}(S^1, H_1) + \mathcal{L}_{\text{sid}}(S^2, H_2),
\]

where \(S^1\) is the speaker label of the first spoken speaker, and \(\mathcal{L}_{\text{sid}}(S^1, H_1)\) refers to the loss function defined by the HAT network. Given the two loss functions, we can train the two networks jointly by interpolating the two loss functions as

\[
\mathcal{L} = \mathcal{L}_{\text{asr}}(X, Y^1, Y^2) + \lambda \mathcal{L}_{\text{sid}}(X, S^1, S^2),
\]

where \(\lambda\) is the interpolation weight. Note that the Unmixing module is shared both for the ASR and SID networks.

6. Experiments and Results

6.1. Dataset

Our experiments were performed on the LibriSpeechMix\footnote{https://github.com/NaoyukiKanda/LibriSpeechMix} dataset [22], which is the simulated overlapped audio derived from the LibriSpeech corpus [26]. We used the same protocol to simulate the training and evaluation data as in [22], and we focus on the 2-speaker case. To generate the simulated training data, for each utterance in the original LibriSpeech\_train960 set, we randomly picked another utterance from a different speaker, and mix the latter with the previous one with a random delay sampled from 0.5 seconds and the length of the first utterance. For each overlapped audio segments, the size of the speaker inventory varies within the range \(K \in [2, 8]\) for training including the 2 speakers presented in the overlapped audio, while for evaluation, we fixed \(K = 8\) and the minimum delay is 0 when mixing the two utterances. The dimension of the d-vectors in our experiments is 128. To generate the d-vectors in \(D\), we randomly select 10 utterances for each speaker in Librispeech, and fed them into a pre-trained 17-layer convolutional network trained on VoxCeleb [35, 36]. The model structure of the d-vector extractor is the modified version of the one in [37]. We used the same approach to generate the evaluation dataset.

6.2. Experimental Setup

In our experiments, we used the 80-dimensional log-mel filterbanks (FBANKs) as features, which were sampled the features at the 10 millisecond frame rate. We then spliced the features by a context window of 3 and finally downsampled super-frames by a factor of 3. Before feeding the features into the CNN blocks in the Unmixing module, we reshaped the input tensor to have 3 input channels, and consequently, the feature dimension became 80. For tokenization, we used 4,000 word-pieces as output tokens for RNN-T, which are generated by byte-pair encoding (BPE) [38]. We used a 4-layer 2-dimensional CNN network for both encoder and mask component in the Unmixing module. We used a 6-layer LSTM as the audio encoder and 2-layer LSTM as the label encoder for the ASR RNN-T network, and a 1-layer LSTM for the audio encoder in the SID HAT network. The number of hidden units was set to be 1024 for all LSTM layers unless specified otherwise. The number of model parameters without the SID network is around 81 million, while the number of parameters in the SID network is around 9 million. We set the dropout ratio as 0.2 for LSTM layers, and applied one layer of time-reduction to reduce the input sequence length by the factor of 2 [39, 9, 40]. We also applied speed perturbation for data augmentation by creating two additional versions of the acoustic features with the speed ratios as 0.9 and 1.1. All our experiments are targeting the 2-speaker scenario.

We report the word error rate (WER) results by considering all the possible label permutations as in [23, 22, 24, 25]. To measure the SID accuracy, we follow the similar protocol as we
6.3. Multi-talker ASR Results

Table 1 shows the multi-talker speech recognition results on the test-clean set with the SURIT model shown in Fig. 2-a). Previously, we achieved 10.8% WER with Short-time Fourier Transform (STFT) features in [24], and with FBANK features, we obtained slightly lower WER of 10.3%. Our streaming results compares favorably with the offline sequence-to-sequence (S2S) based models [29, 41] and the RNN-T-based streaming model proposed in [25]. We also evaluated the accuracy of an offline SURIT model with bi-directional LSTMs (BLSTMs). For a fair comparison, we used the BLSTM model size by reducing the hidden size to be 600. The offline SURIT model achieved 7.2% WER, demonstrating the large gap between the offline and streaming SURIT models.

6.4. Joint ASR and SID Results

Table 2 shows the results of SURIT with or without joint training of the ASR and SID modules. For joint ASR and SID training, the SID loss is much smaller than ASR loss, and similarly for the $\ell_2$ norms of the gradients. To compensate for the difference, we set $\lambda = 10$ and the results are given in Table 2. We also evaluated the stepwise training approach, in which we trained the model with the ASR loss first, and then froze the Unmixing module before training the model with SID loss only. The stepwise training approach is not as effective as the joint training approach in terms of accuracy as Table 2 shows. The reason may be that the Unmixing module was not trained with SID loss, and thus the features from the Unmixing module are not very discriminative for speakers. However, this approach has the advantage that it is suitable for training the model with transcription-only or speaker label only data. We also evaluated adding the SpecAugment [42] as an additional regularization approach on top of dropout and speed perturbation. For SID, this approach is effective for stepwise training, reducing the SER from 8.7% to 8.2%. However, for the joint training approach, additional regularization did not yield improvement, since multi-task learning itself is a kind of regularization.

6.5. SID Latency

Finally, we evaluated the SID latency by measuring the average time step $t_e$ that HAT emitted the speaker label of the first utterance in an overlapped audio during inference. From the baseline system B4 in Table 3, we observe that HAT delays the output of the speaker label until almost at the end of the audio segment. While this is not unexpected, practical applications usually favor low SID latency. To this end, we evaluated the two approaches discussed in Section 4.2. To rule out the effect of the ASR module, we firstly used B4 in Table 2 as the baseline system, and tuned the two hyper-parameters $\alpha$ and $\beta$ while freezing the Unmixing module. From the results in Table 3, we observed that a smaller $\alpha$ resulted in a significant latency reduction, albeit with the increase of SER.

We then evaluated the latency penalty approach as Eq. (8). We set $T_{\text{buffer}} = 3$ for all our experiments. However, this approach itself did not work well in our experiments (C4 vs. B4), as this approach further reduces the gradient of speaker labels during training, and consequently it increases the probability to emit $(bs)$ during inference. To mitigate this effect, we scaled down the gradient of $(bs)$ with a smaller $\alpha$ during training, and we can achieve further latency reduction (C5 vs. C1). Finally, we performed joint training as Eq. (12) with different values of $\alpha$ and $\beta$. Again, we set $\lambda = 10$ for these experiments. We observed a similar trend (D1-D3 vs. B1), though the latency reductions were much more significant since the Unmixing module was also updated with the SID loss during joint training.

7. Conclusions

In this paper, we aim at a simpler, end-to-end framework for rich transcription of meetings and conversational speech. In particular, we focused on joint ASR and SID from multi-talker audio in a streaming fashion. We proposed the SURIT framework, with the Unmixing module as the feature extractor, while ASR and SID are both based on RNN-T and its variant. We proposed a transducer-based SID model, and evaluated two methods to reduce the SID latency. We studied the joint training approach for streaming ASR and SID, and demonstrated the effectiveness of the new framework with strong results. Our future work will focus on reducing the SID latency further as well as a large scale study of the proposed model.
8. References


