



# Streaming End-to-End Speech Recognition for Hybrid RNN-T/Attention Architecture

Takafumi Moriya, Tomohiro Tanaka, Takanori Ashihara, Tsubasa Ochiai, Hiroshi Sato, Atsushi Ando, Ryo Masumura, Marc Delcroix, Taichi Asami

NTT Corporation, Japan

takafumi.moriya.nd@hco.ntt.co.jp

## Abstract

We present a novel architecture with its decoding approach for improving recurrent neural network-transducer (RNN-T) performance. RNN-T is promising for building time-synchronous automatic speech recognition (ASR) systems and thus enhancing streaming ASR applications. We note that encoder-decoder-based sequence-to-sequence models (S2S) have been also used successfully by the ASR community. In this paper, we integrate these popular models in the RNN-T+S2S approach; higher recognition performance than either is achieved due to their integration. However, it is generally deemed to be complicated to use S2S in streaming systems, because the attention mechanism can use arbitrarily long past and future contexts during decoding. Our RNN-T+S2S is composed of the shared encoder, an RNN-T decoder and a triggered attention-based decoder which uses time restricted encoder outputs for attention weight computation. By using the trigger points generated from RNN-T outputs, the S2S branch of RNN-T+S2S activates only when the triggers are detected, which makes streaming ASR practical. Experiments on public and private datasets created to research various tasks demonstrate that our proposal can yield superior recognition performance.

**Index Terms:** speech recognition, end-to-end, recurrent neural network-transducer, attention-based encoder-decoder

## 1. Introduction

Automatic speech recognition (ASR) systems have been dramatically simplified with the introduction of end-to-end (E2E) frameworks. E2E ASR systems can map acoustic features to output token sequences directly. Several modeling approaches have been proposed, e.g. connectionist temporal classification (CTC), recurrent neural network-transducer (RNN-T) and encoder-decoder-based sequence-to-sequence model (S2S) [1–7]. In particular, RNN-T is a variant of CTC, and many studies have been reported recently [8–14]. S2S has also been a success in the ASR community. In this paper, we integrate these successful models, and further improve the recognition performance through their synergy.

The RNN-T with S2S integration proposed in [8]. The architecture is composed of the shared encoder, an RNN-T decoder, and an attention-based decoder. The recognition performance of RNN-T was improved through the assistance of S2S. However, they did not take full advantage of the interaction between RNN-T and S2S because they rescore the RNN-T hypotheses by using the attention-based decoder after completion of one-pass decoding. The reason for this is that the attention mechanism can use arbitrarily long past and future contexts during decoding. Note that using S2S for two-pass rescoring was first proposed in [15] for the re-scoring of NN-Hidden Markov Model hybrid ASR systems, and faced the same problem. Their

solution was to perform two-pass rescoring, which lost the advantage of streaming ASR systems. Our belief was that S2S scoring should be fused with one-pass decoding while taking advantage of RNN-T in a streaming manner.

In order to utilize S2S for streaming ASR, we need to modify the attention mechanism so that it supports streaming style operation. Many studies on streaming S2S-based ASR systems have been published [16–20]. From among them, we adopt the triggered attention (TA) framework [18, 19]. They use the hybrid CTC/Attention model (CTC+S2S) of [21] with the trigger points extracted from the leftmost CTC spikes of each output token except for redundant symbols. TA computes the attention weights using the shared encoder outputs from start-of-speech to a trigger point that is derived from the output of the CTC branch while decoding the speech from left-to-right. Unfortunately, while the TA approach can yield streaming S2S operation, implementing the decoding process is impractically complex precluding its application to RNN-T.

In this paper, we propose an architecture, called RNN-T+S2S; it replaces the CTC branch of the CTC+S2S using trigger points with RNN-T. RNN-T can naturally output trigger points because the output tokens other than redundant symbols are unique and not repeated unlike CTC. Moreover, RNN-T is an extension of CTC and able to take account of language information resulting in higher performance than CTC, yielding the expectation that using RNN-T instead of CTC will attain more accurate trigger points. By applying the trigger points to our RNN-T+S2S, we can integrate RNN-T and S2S scores on-the-fly in one-pass decoding while completely retaining the RNN-T benefits. We also investigate and modify TA for RNN-T+S2S, called RNN-T+S2S+TA, to suit streaming ASR. We expect that S2S with TA can further improve the streaming ASR results of RNN-T. To the best of our knowledge, this is the first work to realize streaming RNN-T+S2S. We evaluate our proposal on two English published corpus and a Japanese private dataset using popular long short-term memory (LSTM) encoder types. The results demonstrate that our RNN-T+S2S (+TA) yields better results than RNN-T or CTC+S2S even though, in particular, the proposed RNN-T+S2S+TA uses the narrower context window used by TA to compute the attention weights compared to CTC+S2S, which has access to the full encoder outputs.

## 2. ASR models

We explain here RNN-T and S2S, which are the foundations of this work. Both models deal with  $\mathbf{X} = [x_1, \dots, x_{T'}]$  and  $Y = [y_1, \dots, y_U]$  which are the input acoustic feature sequence of length- $T'$  and token sequence of length- $U$ , respectively, where  $y_u \in \{1, \dots, K\}$ .  $K$  is the number of tokens including special symbols, i.e. “blank” label for RNN-T, and “end-of-sentence” label for S2S. Here tokens can be characters or subword units.

## 2.1. RNN Transducer (RNN-T)

RNN-T uses the target labels and the extra label,  $\phi$ , and learns the mapping between sequences of different lengths as in CTC [3]. The unique characteristic of RNN-T is its introduction of the prediction network as it does not assume conditional independence between predictions at different time steps. This means that the posterior probabilities are jointly conditioned on not only encoded features but also previous predictions as shown in the following steps. First,  $\mathbf{X}$  is downsampled and encoded into  $\mathbf{H}^{\text{enc}} = [\mathbf{h}_1^{\text{enc}}, \dots, \mathbf{h}_T^{\text{enc}}]$  with length- $T$  via multi-layer LSTM encoder “ $f^{\text{enc}}(\cdot)$ ”. Next, the tokens,  $Y$ , are also encoded into  $\mathbf{H}^{\text{pred}} = [\mathbf{h}_1^{\text{pred}}, \dots, \mathbf{h}_U^{\text{pred}}]$  via prediction network “ $f^{\text{pred}}(\cdot)$ ” which contains an embedding layer and a multi-layer LSTM. Then, these encoded features are fed to the feed-forward network of “ $f^{\text{joint}}(\cdot)$ ”. The above operations, which yield prediction  $\mathbf{y}_{t,u}$ , are defined as follows:

$$\mathbf{h}_t^{\text{enc}} = f^{\text{enc}}(\mathbf{x}_t; \theta^{\text{enc}}), \quad (1)$$

$$\mathbf{h}_u^{\text{pred}} = f^{\text{pred}}(y_{u-1}; \theta^{\text{pred}}), \quad (2)$$

$$\hat{\mathbf{y}}_{t,u} = \text{Softmax}\left(f^{\text{joint}}(\mathbf{h}_t^{\text{enc}}, \mathbf{h}_u^{\text{pred}}; \theta^{\text{joint}})\right), \quad (3)$$

where “ $\text{Softmax}(\cdot)$ ” represents a softmax preactivation layer without learnable parameters. All networks with learnable parameters  $\theta^{\text{RNN-T}} \triangleq [\theta^{\text{enc}}, \theta^{\text{pred}}, \theta^{\text{joint}}]$  are optimized by using RNN-T loss. RNN-T loss and its gradient with respect to the network parameters are efficiently computed by the forward-backward algorithm [3].

## 2.2. Attention-based encoder-decoder (S2S)

We use the attention-based decoder because the encoder shared with RNN-T is used in this paper. The attention-based decoder is formulated as follows. The speech encoded features  $\mathbf{H}^{\text{enc}}$  are obtained from the shared encoder. For decoding the prediction  $\hat{\mathbf{y}}_u$ , the hidden state (memory) activation  $\mathbf{s}_u$  of the LSTM-based decoder  $f^{\text{dec}}(\cdot)$  at the  $u$ -th step is computed as:

$$\mathbf{s}_u = f^{\text{dec}}(\mathbf{s}_{u-1}, \mathbf{g}_u, y_{u-1}; \theta^{\text{dec}}), \quad (4)$$

where  $\mathbf{g}_u$  and  $y_{u-1}$  denote the context vector at the  $u$ -th step and the target token at the previous step, respectively.  $\mathbf{g}_u$  is the weighted sum of the encoder output sequence:

$$\mathbf{g}_u = \sum_{t=1}^T a_{t,u} \mathbf{h}_t^{\text{enc}}, \quad (5)$$

where  $a_{t,u}$  is the attention weight of  $\mathbf{h}_t^{\text{enc}}$ . It is calculated as:

$$a_{t,u} = f^{\text{att}}\left(\{a_{t,u-1}\}_{t=1}^T, \mathbf{s}_{u-1}, \mathbf{h}_t^{\text{enc}}; \theta^{\text{att}}\right), \quad (6)$$

where  $f^{\text{att}}(\cdot)$  computes attention weights; location-aware attention is used in this paper. Using  $\mathbf{s}_u$ , the attention-based decoder predicts the next label distribution  $\hat{\mathbf{y}}_u$  as:

$$\hat{\mathbf{y}}_u = \text{Softmax}\left(f^{\text{out}}(\mathbf{s}_u; \theta^{\text{out}})\right), \quad (7)$$

where the function  $f^{\text{out}}(\cdot)$  denotes a linear output layer. The learnable parameters  $\theta^{\text{S2S}} \triangleq [\theta^{\text{enc}}, \theta^{\text{dec}}, \theta^{\text{att}}, \theta^{\text{out}}]$  are optimized by the cross entropy loss.

### 2.2.1. Triggered attention (TA) with RNN-T

Our goal is to further improve RNN-T performance while still supporting streaming ASR. Since the original attention mechanism can involve arbitrarily long past and future contexts during decoding, it should be modified to realize streaming style operation. In this work, we utilize a triggered attention framework that can support streaming S2S [18], and modify Eq. (5) and (6) for our TA with RNN-T as follows:

$$\mathbf{g}_u = \sum_{t=1}^{\tau_u} a_{t,u} \mathbf{h}_t^{\text{enc}}, \quad (8)$$

$$a_{t,u} = f^{\text{att}}\left(\{a_{t,u-1}\}_{t=1}^{\tau_u}, \mathbf{s}_{u-1}, \mathbf{h}_t^{\text{enc}}; \theta^{\text{att}}\right), \quad (9)$$

where  $\tau_u$  indicates a restricted time context; it is defined as  $\tau_u = t_u + \varepsilon$ .  $\varepsilon$  is the look-ahead hyperparameter.  $t_u$  denotes the trigger point generated from RNN-T, the time index of the  $u$ -th non-blank token occurrence. In the training step, we use the RNN-T alignments  $\hat{\boldsymbol{\pi}}$  generated by the forward-backward algorithm [3]. Note that we remove the probability paths corresponding to blank symbols;  $\Pi$  of Fig. 1 illustrates an example of path removal. The  $u$ -th token path  $\hat{\boldsymbol{\pi}}_u \in \mathbb{R}^T$  indicates a  $u$ -th row vector of  $\hat{\boldsymbol{\pi}}$  in  $\Pi$  of Fig. 1. Therefore, the  $u$ -th trigger point  $t_u$  (time indexes of each yellow plot in  $\Pi$  of Fig. 1) is determined by the argument of the maximum of  $\hat{\boldsymbol{\pi}}_u$  of the time axis.

### 2.3. Hybrid RNN-T/Attention architecture (RNN-T+S2S)

Both RNN-T and S2S have been used successfully by the ASR community. In this paper, we integrate those successful models and achieve higher recognition performance by their interaction; it is overviewed in Fig. 1. The key is that we replace the CTC branch of the attention-based CTC+S2S using trigger points with RNN-T. The resulting architecture is similar to that of [8]. The difference from [8] lies in the decoding algorithm. While their model performs two-pass rescoring, our RNN-T+S2S uses trigger points, which offers one-pass decoding. Moreover, as the proposal applies TA to the training step it realizes streaming ASR, and its architecture is called as RNN-T+S2S+TA. In the following, we describe training and decoding approaches adopted our RNN-T+S2S (+TA) proposal.

**Training:** We perform multi-step training following [8] for stable training and improved RNN-T+S2S (+TA) performance<sup>1</sup>. At first, we train the shared encoder and RNN-T decoder from scratch. We apply SimpleFlat pre-training (SFPT) initialization in this step [14]. Then the shared encoder is frozen, and only the (triggered) attention-based decoder parameters  $\theta^{\text{S2S (RNN-T+S2S)}} \triangleq [\theta^{\text{dec}}, \theta^{\text{att}}, \theta^{\text{out}}]$  are trained. We also use SFPT for attention-based decoder initialization. The fine-tuned model is used in the evaluation.

**Decoding:** We investigate two different decoding methods. One allows the attention-based decoder to perform one-pass decoding but not streaming ASR. This means that the attention weights of RNN-T+S2S are computed using all encoder outputs whenever the trigger points (time indexes of each red circle in IV of Fig. 1) are detected. The other approach offers full streaming ASR by using RNN-T+S2S+TA. The TA weights with restricted time context are illustrated in I of Fig. 1. The colored part represents the valid attention weights and the restricted time context  $\tau_u$  in Eq. (9) is determined by trigger point  $t_u$ . The attention weights, white color, are not computed or used in the prediction.

<sup>1</sup>We tried “Deep fine-tuning” for all models as in [8], but no improvements were found in our experiments.

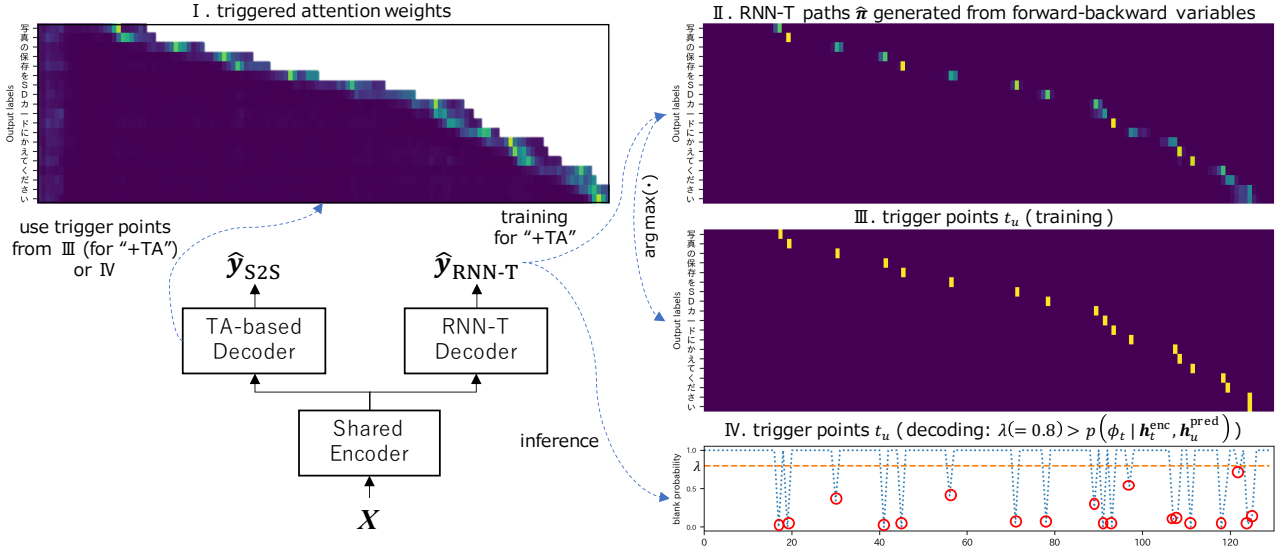


Figure 1: Schematic diagram of our RNN-T+S2S (+TA) training and decoding steps. Each picture, I - IV, indicates the triggered attention weights, RNN-T forward-backward variables, trigger points in training (yellow plots) and decoding (red circles) steps when using the ULSTM encoder on NVS development set respectively. The reference is “写真の保存をSDカードにかえてください (Save the photos to the SD card.)”. The trigger points generated from III are only used for RNN-T+S2S+TA training.

We explain trigger point creation and joint decoding approach for our RNN-T+S2S (+TA) here. In the inference step, we use the blank probability  $p(\phi_t | \mathbf{h}_t^{\text{enc}}, \mathbf{h}_u^{\text{pred}})$  of the RNN-T output for the trigger point generation as described in IV of Fig. 1. The time indexes of each red circle indicate the trigger points that are less than tunable threshold  $\lambda$ . Therefore, RNN-T+S2S computes the attention weights using full encoder outputs, and RNN-T+S2S+TA computes the attention weights using only the time restricted encoder outputs from the beginning to  $\tau_u$ ; it is activated when  $p(\phi_t | \mathbf{h}_t^{\text{enc}}, \mathbf{h}_u^{\text{pred}})$  is less than  $\lambda$ . This can efficiently suppress the attention-based decoder activation at every time step, which reduces the attention weight computation costs. The most probable hypothesis of RNN-T+S2S (+TA) at each trigger step is given by:

$$\hat{Y} = \arg \max_Y \{(1 - \rho) \log p_{\text{RNN-T}}(Y | \mathbf{X}) + \rho \log p_{\text{S2S}}(Y | \mathbf{X})\}, \quad (10)$$

where  $\rho$  is a tunable parameter.  $p_{\text{RNN-T}}(Y | \mathbf{X})$  and  $p_{\text{S2S}}(Y | \mathbf{X})$  can be computed from Eq. (3) and (7) respectively. Our beam search implementation is based on alignment-length synchronous decoding [11], and retains all its fundamental components. S2S computation and scoring are employed on-the-fly during beam search as in language model (LM) shallow fusion [22]. We expect that S2S (+TA) can further improve the one-pass decoding results of RNN-T by this interaction.

### 3. Experiments

#### 3.1. Data

We evaluated our proposal on three speech recognition tasks; Switchboard [23], TED-LIUM release-2 (TED2) [24], and NTT Japanese voice search production dataset (NVS). The datasets contain voiced samples totaling 260, 210 and 1000 hours, respectively. The test set of NVS task consists of 3 hours of speech data. In this paper, we adopt 1000 subwords, 500 subwords and 3500 characters (plus blank  $\phi$ ) as the output tokens for Switchboard, TED2 and NVS, respectively. Speed perturbation (x3) [25] was applied only to TED2 as in the ESPnet setups [26]. The subword units of Switchboard and TED2 were determined by using the SentencePiece method [27]. The train-

ing and evaluation data were preprocessed following the Kaldi and modified ESPnet toolkits [26, 28].

#### 3.2. System configuration

The input feature, a 40-dimensional log Mel-filterbank appended with delta and acceleration coefficients, is concatenated by non-overlapping frame stacking [29]; three frames were stacked and skipped to make each new super-frame. Note that SpecAugment was applied to each input feature [30].

We investigated three different encoder types; unidirectional LSTM (ULSTM), bidirectional LSTM (BLSTM) and latency-controlled BLSTM (LCBLSTM) [31–33]. We adopted six-layer (LC) BLSTM encoders with 512 cells per direction. Each output of (LC) BLSTM was followed by a linear projection layer to reduce the dimensionality to 512. The chunk sizes for LCBLSTM,  $N_c$  and  $N_r$ , were set to 30. The memory cells were doubled when using the ULSTM encoder. As the prediction network, we stacked an embedding layer of size 512 and two ULSTM layers each with 512 cells. The outputs of the encoder and prediction network were fed to the joint network, which linearly projected to 512 dimensions and then classified the target tokens of each dataset. The (TA-based) S2S decoder has also an embedding layer of size 512 followed by two ULSTM layers each with 512 cells. All S2S models used a single-head location-based attention mechanism [6]. For all TA models, the look-ahead length  $\varepsilon$  was set to 8 in both training and decoding steps. Moreover, we trained “CTC+S2S” models to allow comparison with our proposal “RNN-T+S2S (+TA)”. The setups of encoder and decoder of CTC+S2S were the same as the above S2S setting of RNN-T+S2S. As the CTC output layer, we additionally stacked the shared encoder of CTC+S2S, and all network parameters were trained as in [21, 26].

All network parameters before training were initialized with random values following [34]. These networks with RNN-T loss were trained for 25 epochs using the AdaDelta optimizer [35]. Then the S2S decoders of RNN-T+S2S were trained for 25 epochs using also AdaDelta. In the SFPT step, we used the early stopping strategy with a validation set. The dropout rates of the encoders and the other networks were 0.4 and 0.2, respectively [36]. We also used gradient clipping with a thresh-

Table 1: Comparisons of CTC, RNN-T and with SimpleFlat pre-training (SFPT) on Switchboard, TED2 and NVS when **greedy search decoding**. Note that “\*” means convergence failure in our experiments.

Encoder	System	Switchboard		TED2	NVS
		SWB	CH		
BLSTM	CTC	19.6	32.9	15.3	12.8
	RNN-T	18.3	31.1	15.6	11.3
	RNN-T+SFPT	<b>16.4</b>	<b>29.5</b>	<b>13.9</b>	<b>7.4</b>
ULSTM	CTC	25.0	40.7	21.8	19.4
	RNN-T	24.3	40.2	*	18.3
	RNN-T+SFPT	<b>23.0</b>	<b>38.7</b>	<b>19.9</b>	<b>13.6</b>
LCBLSTM	CTC	24.6	37.0	16.7	14.5
	RNN-T	19.6	31.6	16.2	13.7
	RNN-T+SFPT	<b>18.3</b>	<b>30.9</b>	<b>15.1</b>	<b>7.7</b>

Table 2: Comparisons of S2S, CTC+S2S, RNN-T and RNN-T+S2S (proposal) with BLSTM encoder. All systems used beam search decoding and performed **offline decoding** although the one-pass decoding manner was kept.

System	Switchboard		TED2	NVS
	SWB	CH		
S2S (CTC+S2S)	16.2	30.2	14.9	10.7
CTC+S2S	15.5	28.7	13.4	9.7
RNN-T	15.2	28.0	13.3	6.8
S2S (RNN-T+S2S)	18.0	31.9	14.0	7.2
RNN-T+S2S	<b>14.9</b>	<b>27.4</b>	<b>12.6</b>	<b>6.5</b>

old of 5.0. The minibatch size was set to 32 in all experiments. All models were trained using the PyTorch toolkit [37].

For most experiments, we used alignment-length synchronous decoding with beam size of 20 when performing beam search decoding [11]. The tunable parameters, i.e. threshold  $\lambda$  and S2S weight  $\rho$  of Eq. (10), for RNN-T+S2S (+TA) were consistently set to 0.8 and 0.3 in all experiments. For CTC+S2S joint decoding, we adopted the tuned parameters in [26]. We evaluated performance in terms of word error rate (WER) for Switchboard and TED2, as well as character error rate (CER) for NVS due to the ambiguity of Japanese word boundaries.

### 3.3. Results

First, we compared the performance of RNN-T with CTC; each error rate when greedy search decoding is shown in Table 1. “CTC” and “RNN-T” mean that those models were trained with random initialization, and “RNN-T+SFPT” is the fine-tuned models from SFPT initialization. “CTC” systems used just the CTC branch of “CTC+S2S” to perform decoding. We can see that the WER/CERs of “RNN-T” were better than those of “CTC” on almost tasks. Moreover, “RNN-T+SFPT” yielded further improvements in all tasks. Hereafter, our RNN-T+S2S (+TA) adopted the frozen parameters of “RNN-T+SFPT”.

We evaluated our RNN-T+S2S which is compared with “CTC+S2S” and “RNN-T” using the BLSTM encoders with offline decoding, although one-pass decoding was kept in all tasks; the WER/CERs of the test set are shown in Table 2. “CTC+S2S” indicates that those models were jointly trained with CTC and performed joint decoding. “CTC+S2S” used all encoder outputs for attention weight computation, which provides richer terms than the triggered attention-based CTC+S2S [18, 19]. “S2S (CTC+S2S)” systems used S2S branch of “CTC+S2S” when decoding only. “RNN-T” was the same system as “RNN-T+SFPT” in Table 1. “S2S (RNN-T+S2S)” represents S2S branch of “RNN-T+S2S”, and were trained using only the attention-based decoder part detailed in Section 2.3. “RNN-T+S2S” is our proposal and performed joint one-pass decoding by using trigger points and both branches of “RNN-T” and “S2S (RNN-T+S2S)”. We can see “RNN-

Table 3: Comparisons of offline CTC+S2S, streaming RNN-T, offline RNN-T+S2S and streaming RNN-T+S2S (+TA). All systems used beam search decoding. The systems with “✓” in column “fs” indicate **fully streaming ASR results (fs)**, and the others are offline decoding results.

Enc.	System	fs	Switchboard		TED2	NVS
			SWB	CH		
ULSTM	S2S+CTC	-	19.8	35.6	<b>17.5</b>	14.7
	RNN-T	✓	20.8	35.8	18.5	12.2
	RNN-T+S2S	-	<b>19.3</b>	<b>34.8</b>	17.9	<b>10.7</b>
	RNN-T+S2S	✓	20.8	36.5	18.7	11.8
	RNN-T+S2S+TA	✓	19.6	35.0	<b>17.5</b>	11.2
LCBLSTM	S2S+CTC	-	17.4	29.7	14.2	9.9
	RNN-T	✓	16.7	29.4	14.3	7.1
	RNN-T+S2S	-	16.2	<b>28.5</b>	13.7	<b>6.8</b>
	RNN-T+S2S	✓	16.5	28.9	14.0	7.1
	RNN-T+S2S+TA	✓	<b>15.9</b>	<b>28.5</b>	<b>13.6</b>	<b>6.8</b>

T+S2S” yielded generally better WER/CERs than “CTC+S2S” and “RNN-T”, and thus our proposed “RNN-T+S2S” architecture and its decoding approach were effective in realizing further RNN-T improvements in one-pass decoding manner.

Finally, we evaluated our fully streaming RNN-T+S2S (+TA) models in comparison with offline “CTC+S2S” and streaming “RNN-T” on all tasks; the results are shown in Table 3. Note that the difference from the offline experiments was the use of the narrower context window for the attention weight computation by using trigger points as in I of Fig. 1. Therefore, our models with checkmark in “fs” column could perform streaming ASR. “RNN-T” was the same system as “RNN-T+SFPT” in Table 1. The “RNN-T+S2S” systems without checkmark computed the attention weights using all encoder outputs whenever each trigger point was detected, and thus one-pass and offline decoding were performed. “RNN-T+S2S” with checkmark represents offline systems but those were forced to perform streaming ASR by using trigger points as in TA. “RNN-T+S2S+TA” means that the trigger points were used in both “RNN-T+S2S” training and decoding steps. We can see that the systems of “RNN-T+S2S” with checkmark used in forced streaming ASR manner were worse than “RNN-T+S2S” without checkmark. Applying TA to “RNN-T+S2S” in both training and decoding steps, i.e. “RNN-T+S2S+TA”, mitigated the problem in the decoding step, and performance was improved. Moreover, our “RNN-T+S2S+TA” could achieve the same or better performance than “CTC+S2S” even though the latter used offline decoding. We argue that the TA approach is also effective for improving RNN-T performance in streaming ASR manner. Note that we presented results with LSTM-based models because we aim at streaming processing. Our framework can be also applied to large and self-attention-based model setups [38] with LM integration, and confirming this is a future work.

## 4. Conclusion

We have proposed hybrid RNN-T/Attention architecture, called RNN-T+S2S, and its decoding approach that offers one-pass decoding. Our proposal integrates RNN-T and S2S so as to further improve RNN-T performance by their interaction. Replacing the CTC branch of the CTC+S2S using trigger points with RNN-T yields one-pass decoding. We also investigated and modified TA for RNN-T+S2S, called RNN-T+S2S+TA, to suit streaming ASR. Tests on three different datasets covering different speaking styles and languages confirmed that RNN-T+S2S+TA and its decoding approach attained better performance than RNN-T and CTC+S2S even though the CTC+S2S used offline decoding.

## 5. References

- [1] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification : Labelling unsegmented sequence data with recurrent neural networks," in *Proc. of ICML*, 2006, pp. 369–376.
- [2] A. Graves and N. Jaitly, "Towards End-To-End speech recognition with recurrent neural networks," in *Proc. of ICLR*, 2014, pp. 1764–1772.
- [3] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. of ICML*, 2012.
- [4] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. of ICASSP*, 2013, pp. 6645–6649.
- [5] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: first results," in *Advances in NIPS*, 2014.
- [6] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in NIPS*, 2015, pp. 577–585.
- [7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in NIPS*, 2017, pp. 5998–6008.
- [8] T. N. Sainath, R. Pang, D. Rybach, Y. He, R. Prabhavalkar, W. Li, M. Visontai, Q. Liang, T. Strohmaier, Y. Wu, I. McGraw, and C. Chiu, "Two-pass end-to-end speech recognition," in *Proc. of Interspeech*, G. Kubin and Z. Kacic, Eds., 2019, pp. 2773–2777.
- [9] H. Hu, R. Zhao, J. Li, L. Lu, and Y. Gong, "Exploring pre-training with alignments for RNN Transducer based end-to-end speech recognition," in *Proc. of ICASSP*, 2020, pp. 7074–7078.
- [10] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S. Chang, W. Li, R. Alvarez, Z. Chen, C. Chiu, D. Garcia, A. Gruenstein, K. Hu, A. Kannan, Q. Liang, I. McGraw, C. Peyser, R. Prabhavalkar, G. Pundak, D. Rybach, Y. Shangguan, Y. Sheth, T. Strohmaier, M. Visontai, Y. Wu, Y. Zhang, and D. Zhao, "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *Proc. of ICASSP*, 2020, pp. 6059–6063.
- [11] G. Saon, Z. Tuske, and K. Audhkhasi, "Alignment-length synchronous decoding for RNN Transducer," in *Proc. of ICASSP*, 2020, pp. 7799–7803.
- [12] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A new training pipeline for an improved neural transducer," in *Proc. of INTERSPEECH*, 2020, pp. 2812–2816.
- [13] G. Kurata and G. Saon, "Knowledge distillation from offline to streaming RNN transducer for end-to-end speech recognition," in *Proc. of INTERSPEECH*, 2020, pp. 2117–2121.
- [14] 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. of ICASSP*, 2021, pp. 5664–5668.
- [15] T. Tanaka, R. Masumura, T. Moriya, and Y. Aono, "Neural speech-to-text language models for rescoring hypotheses of DNN-HMM hybrid automatic speech recognition systems," *Proc. of AP-SIPA ASC*, pp. 196–200, 2018.
- [16] C.-C. Chiu\* and C. Raffel\*, "Monotonic chunkwise attention," in *Proc. of ICLR*, 2018.
- [17] C. Chiu, A. Kannan, R. Prabhavalkar, Z. Chen, T. N. Sainath, Y. Wu, W. Han, Y. Zhang, R. Pang, S. Kishchenko, P. Nguyen, A. Narayanan, H. Liao, and S. Zhang, "A comparison of end-to-end models for long-form speech recognition," in *Proc. of ASRU*, 2019, pp. 889–896.
- [18] N. Moritz, T. Hori, and J. L. Roux, "Triggered attention for end-to-end speech recognition," in *Proc. of ICASSP*, 2019, pp. 5666–5670.
- [19] —, "Streaming end-to-end speech recognition with joint CTC-Attention based models," in *Proc. of ASRU*, 2019, pp. 936–943.
- [20] H. Inaguma, M. Mimura, and T. Kawahara, "CTC-synchronous training for monotonic attention model," in *Proc. of INTERSPEECH*, 2020, pp. 571–575.
- [21] S. Watanabe, T. Hori, S. Kim, J. Hershey, and T. Hayashi, "Hybrid CTC/Attention architecture for end-to-end speech recognition," *IEEE Journal on Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.
- [22] A. Kannan, Y. Wu, P. Nguyen, T. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *Proc. of ICASSP*, 2018, pp. 5824–5828.
- [23] J. Godfrey, E. Holliman, and J. McDaniel, "SWITCHBOARD: telephone speech corpus for research and development . acoustics," in *Proc. of ICASSP*, 1992, pp. 517–520.
- [24] A. Rousseau, P. Deléglise, and Y. Estève, "TED-LIUM: an automatic speech recognition dedicated corpus," in *Proc. of LREC*, 2012, pp. 125–129.
- [25] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. of INTERSPEECH*, 2015, pp. 3586–3589.
- [26] 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. of INTERSPEECH*, 2018, pp. 2207–2211.
- [27] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proc. of EMNLP: System Demonstrations*, 2018, pp. 66–71.
- [28] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovski, G. Stemmer, and K. Veseli, "The Kaldi speech recognition toolkit," in *Proc. of ASRU*, 2011.
- [29] H. Sak, A. W. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," in *Proc. of INTERSPEECH*, 2015, pp. 1468–1472.
- [30] D. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. Cubuk, and Q. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. of INTERSPEECH*, 2019, pp. 2613–2617.
- [31] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [32] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, and J. R. Glass, "Highway long short-term memory RNNs for distant speech recognition," in *Proc. of ICASSP*, 2016, pp. 5755–5759.
- [33] S. Xue and Z. Yan, "Improving latency-controlled blstm acoustic models for online speech recognition," *Proc. of ICASSP*, pp. 5340–5344, 2017.
- [34] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. of ICCV*, 2015, pp. 1026–1034.
- [35] M. D. Zeiler, "Adadelta: An adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012.
- [36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, pp. 1929–1958, 2014.
- [37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.
- [38] A. Gulati, J. Qin, 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. of INTERSPEECH*, 2020, pp. 5036–5040.