STREAMING AUTOMATIC SPEECH RECOGNITION WITH THE TRANSFORMER MODEL

Niko Moritz, Takaaki Hori, Jonathan Le Roux

Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA

ABSTRACT

Encoder-decoder based sequence-to-sequence models have demonstrated state-of-the-art results in end-to-end automatic speech recognition (ASR). Recently, the transformer architecture, which uses self-attention to model temporal context information, has been shown to achieve significantly lower word error rates (WERs) compared to recurrent neural network (RNN) based system architectures. Despite its success, the practical usage is limited to offline ASR tasks, since encoder-decoder architectures typically require an entire speech utterance as input. In this work, we propose a transformer based end-to-end ASR system for streaming ASR, where an output must be generated shortly after each spoken word. To achieve this, we apply time-restricted self-attention for the encoder and triggered attention for the encoder-decoder attention mechanism. Our proposed streaming transformer architecture achieves 2.8% and 7.2% WER for the "clean" and "other" test data of LibriSpeech, which to our knowledge is the best published streaming end-to-end ASR result for this task.

***Index Terms*—** automatic speech recognition, streaming, end-to-end, transformer, triggered attention

1. INTRODUCTION

Hybrid hidden Markov model (HMM) based automatic speech recognition (ASR) systems have provided state-of-the-art results for the last few decades [1, 2]. End-to-end ASR systems, which approach the speech-to-text conversion problem using a single sequence-to-sequence model, have recently demonstrated competitive performance [3]. The most popular and successful end-to-end ASR approaches are based on connectionist temporal classification (CTC) [4], recurrent neural network (RNN) transducer (RNN-T) [5], and attention-based encoder-decoder architectures [6]. RNN-T based ASR systems achieve state-of-the-art ASR performance for streaming/online applications and are successfully deployed in production systems [7, 8]. Attention-based encoder-decoder architectures, however, are the best performing end-to-end ASR systems [9], but they cannot be easily applied in a streaming fashion, which prevents them from being used more widely in practice. To overcome this limitation, different methods for streaming ASR with attention-based systems haven been proposed such as the neural transducer (NT) [10], monotonic chunkwise attention (MoChA) [11], and triggered attention (TA) [12]. The NT relies on traditional block processing with fixed window size and stride to produce incremental attention model outputs. The MoChA approach uses an extra layer to compute a selection probability that defines the length of the output label sequence and provides an alignment to chunk the encoder state sequence prior to soft attention. The TA system requires that the attention-based encoder-decoder model is trained jointly with a CTC objective function, which has also been shown to improve attention-based systems [13], and the CTC output is used to predict an alignment that triggers the attention decoding process [12]. A frame-synchronous one-pass decoding algorithm for joint CTC-attention scoring was proposed in [14] to further optimize and enhance ASR decoding using the TA concept.

Besides the end-to-end ASR modeling approach, the underlying neural network architecture is of paramount importance as well to achieve good ASR performance. RNN-based architectures, such as the long short-term memory (LSTM) neural network, are often applied for end-to-end ASR systems. Bidirectional LSTMs (BLSTMs) achieve state-of-the-art results among such RNN-based systems but are unsuitable for application in a streaming fashion, where unidirectional LSTMs or latency-controlled BLSTMs (LC-BLSTMs) must be applied instead [15]. The parallel time-delayed LSTM (PTDLSTM) architecture has been proposed to further reduce the word error rate (WER) gap between unidirectional and bidirectional architectures and to improve the computational complexity compared to the LC-BLSTM [15]. Recently, the transformer model, which is an encoder-decoder type of architecture based on self-attention originally proposed for machine translation [16], has been applied to ASR with promising results and improved WERs compared to RNN-based architectures [17].

In this work, we apply time-restricted self-attention to the encoder, and the TA concept to the encoder-decoder attention mechanism of the transformer model to enable the application of online/streaming ASR. The transformer model is jointly trained with a CTC objective to optimize training and decoding results as well as to enable the TA concept [3, 12]. For joint CTC-transformer decoding and scoring, we employ the frame-synchronous one-pass decoding algorithm proposed in [14].

2. STREAMING TRANSFORMER

The streaming architecture of the proposed transformer-based ASR system is shown in Fig. 1. The transformer is an encoder-decoder type of architecture that uses two different attention layers: encoder-decoder attention and self-attention. The encoder-decoder attention can produce variable output lengths by using one or multiple query vectors, the decoder states, to control attention to a sequence of input values, the encoder state sequence. In self-attention (SA), the queries, values, and keys are derived from the same input sequence, which results in an output sequence of the same length. Both attention types of the transformer model are based on the scaled dot-product attention mechanism,

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^{\mathsf{T}}}{\sqrt{d_k}}\right)V, \qquad (1)$$

where $Q \in \mathbb{R}^{n_q \times d_q}$, $K \in \mathbb{R}^{n_k \times d_k}$, and $V \in \mathbb{R}^{n_v \times d_v}$ are the queries, keys, and values, where the d_* denote dimensions and the n_* denote sequence lengths, $d_q = d_k$, and $n_k = n_v$ [16]. Instead of using a single attention head, multiple attention heads are used by

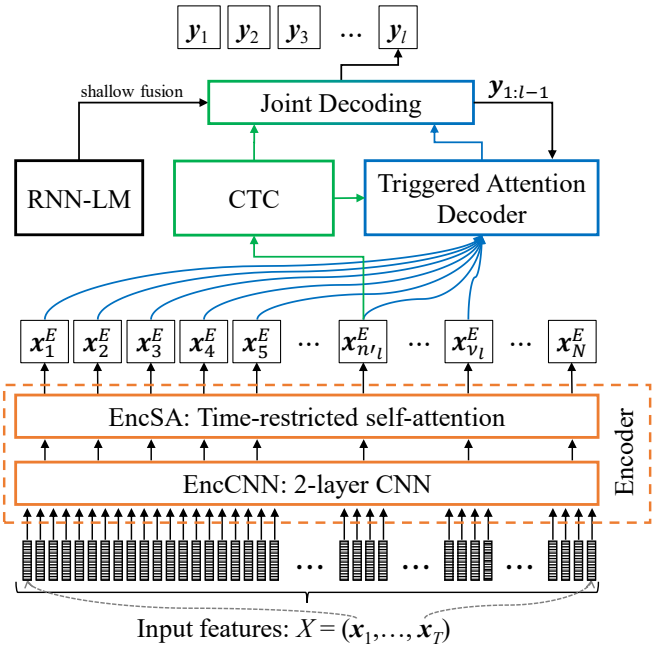

Fig. 1. Joint CTC-TA decoding scheme for streaming ASR with a transformer-based architecture.

each layer of the transformer model with

$$\text{MHA}(\hat{Q}, \hat{K}, \hat{V}) = \text{Concat}(\text{Head}_1, \ldots, \text{Head}_{d_h})W^{\text{H}} \quad (2)$$

$$\text{and Head}_i = \text{Attention}(\hat{Q}W_i^Q, \hat{K}W_i^K, \hat{V}W_i^V), \quad (3)$$

where $\hat{Q}$, $\hat{K}$, and $\hat{V}$ are inputs to the multi-head attention (MHA) layer, Head_i represents the output of the i-th attention head for a total number of d_h heads, and $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_q}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ as well as $W^H \in \mathbb{R}^{d_h d_v \times d_{\text{model}}}$ are trainable weight matrices with typically $d_k = d_v = d_{\text{model}}/d_h$.

2.1. Encoder: Time-restricted self-attention

The encoder of our transformer architecture consists of a two-layer CNN module EncCNN and a stack of E self-attention layers EncSA:

$$X_0 = \text{EncCNN}(X), \quad (4)$$

$$X_E = \text{EncSA}(X_0), \quad (5)$$

where $X = (x_1, \ldots, x_T)$ denotes a sequence of acoustic input features, which are 80-dimensional log-mel spectral energies plus 3 extra features for pitch information [18]. Both CNN layers of EncCNN use a stride of size 2, a kernel size of 3×3, and a ReLU activation function. Thus, the striding reduces the frame rate of output sequence X_0 by a factor of 4 compared to the feature frame rate of X. The EncSA module of (5) consists of E layers, where the e-th layer, for $e = 1, \ldots, E$, is a composite of a multi-head self-attention layer

$$X'_e = X_{e-1} + \text{MHA}_e(X_{e-1}, X_{e-1}, X_{e-1}), \quad (6)$$

and two feed-forward neural networks of inner dimension d_{ff} and outer dimension d_{model} that are separated by a ReLU activation function as follows:

$$X_e = X'_e + \text{FF}_e(X'_e), \quad (7)$$

$$\text{with FF}_e(X'_e) = \text{ReLU}(X'_e W^{\text{ff}}_{e,1} + b^{\text{ff}}_{e,1})W^{\text{ff}}_{e,2} + b^{\text{ff}}_{e,2}, \quad (8)$$

where $W^{\text{ff}}_{e,1} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{ff}}}$, $W^{\text{ff}}_{e,2} \in \mathbb{R}^{d_{\text{ff}} \times d_{\text{model}}}$, $b^{\text{ff}}_{e,1} \in \mathbb{R}^{d_{\text{ff}}}$, and $b^{\text{ff}}_{e,2} \in \mathbb{R}^{d_{\text{model}}}$ are trainable weight matrices and bias vectors.

In order to control the latency of the encoder architecture, the future context of input sequence X_0 is limited to a fixed size, which is referred to as restricted or time-restricted self-attention [16] and was first applied to hybrid HMM-based ASR systems [19]. We can define a time-restricted self-attention encoder EncSA^{tr}, with $n = 1, \ldots, N$, as

$$x^E_{1:n} = \text{EncSA}^{\text{tr}}(x^0_{1:n+\varepsilon^{\text{enc}}}), \quad (9)$$

where $x^0_{1:n+\varepsilon^{\text{enc}}} = X_0[1 : n + \varepsilon^{\text{enc}}] = (x^0_1, \ldots, x^0_{n+\varepsilon^{\text{enc}}})$, and ε^{enc} denotes the number of look-ahead frames used by the time-restricted self-attention mechanism.

2.2. Decoder: Triggered attention

The encoder-decoder attention mechanism of the transformer model is using the TA concept [12, 14] to enable the decoder to operate in a streaming fashion. TA training requires an alignment between the encoder state sequence X_E and the label sequence $Y = (y_1, \ldots, y_L)$ to condition the attention mechanism of the decoder only on past encoder frames plus a fixed number of look-ahead frames ε^{dec}. This information is generated by forced alignment using an auxiliary CTC objective $p_{\text{ctc}}(Y|X_E)$ [4], which is jointly trained with the decoder model, where the encoder neural network is shared [12, 13, 17].

The triggered attention objective function is defined as

$$p_{\text{ta}}(Y|X_E) = \prod_{l=1}^{L} p(y_l|y_{1:l-1}, x^E_{1:\nu_l}) \quad (10)$$

with $\nu_l = n'_l + \varepsilon^{\text{dec}}$, where n'_l denotes the position of the first occurrence of label y_l in the CTC forced alignment sequence [12, 14], $y_{1:l-1} = (y_1, \ldots, y_{l-1})$, and $x^E_{1:\nu_l} = (x^E_1, \ldots, x^E_{\nu_l})$, which corresponds to the truncated encoder sequence. The term $p(y_l|y_{1:l-1}, x^E_{1:\nu_l})$ represents the transformer decoder model

$$p(y_l|y_{1:l-1}, x^E_{1:\nu_l}) = \text{DecTA}(x^E_{1:\nu_l}, y_{1:l-1}), \quad (11)$$

with

$$z^0_{1:l} = \text{Embed}(\langle\text{sos}\rangle, y_1, \ldots, y_{l-1}), \quad (12)$$

$$\overline{z}^d_l = z^{d-1}_l + \text{MHA}^{\text{self}}_d(z^{d-1}_l, z^{d-1}_{1:l}, z^{d-1}_{1:l}), \quad (13)$$

$$\overline{\overline{z}}^d_l = \overline{z}^d_l + \text{MHA}^{\text{dec}}_d(\overline{z}^d_l, x^E_{1:\nu_l}, x^E_{1:\nu_l}), \quad (14)$$

$$z^d_l = \overline{\overline{z}}^d_l + \text{FF}_d(\overline{\overline{z}}^d_l), \quad (15)$$

for $d = 1, \ldots, D$, where D denotes the number of decoder layers. Embed converts the input label sequence $(\langle\text{sos}\rangle, y_1, \ldots, y_{l-1})$ into a sequence of trainable embedding vectors $z^0_{1:l}$, where $\langle\text{sos}\rangle$ denotes the start of sentence symbol. Function DecTA finally predicts the posterior probability of label y_l by applying a fully-connected projection layer to z^D_l and a softmax distribution over that output.

The CTC model and the triggered attention model of (10) are trained jointly using the multi-objective loss function

$$\mathcal{L} = -\gamma \log p_{\text{ctc}} - (1 - \gamma) \log p_{\text{ta}}, \quad (16)$$

where hyperparameter γ controls the weighting between the two objective functions p_{ctc} and p_{ta}.

2.3. Positional encoding

Sinusoidal positional positional encodings (PE) are added to the sequences X_0 and Z_0, which can be written as

$$\text{PE}(\text{pos}, 2i) = \sin(\text{pos}/10000^{2i/d_{\text{model}}}), \quad (17)$$

$$\text{PE}(\text{pos}, 2i+1) = \cos(\text{pos}/10000^{2i/d_{\text{model}}}), \quad (18)$$

Algorithm 1 Joint CTC-triggered attention decoding

1: **procedure** DECODE(X_E, p_{ctc}, λ, α_0, α, β, K, P, θ_1, θ_2)
2: $\quad\ell \leftarrow (\langle\text{sos}\rangle,)$
3: $\quad\Omega \leftarrow \{\ell\}, \Omega_{\text{ta}} \leftarrow \{\ell\}$
4: $\quad p_{\text{nb}}(\ell) \leftarrow 0, p_{\text{b}}(\ell) \leftarrow 1$
5: $\quad p_{\text{ta}}(\ell) \leftarrow 1$
6: $\quad$**for** $n = 1, \ldots, N$ **do**
7: $\quad\quad\Omega_{\text{ctc}}, p_{\text{nb}}, p_{\text{b}} \leftarrow$ CTCPREFIX$(p_{\text{ctc}}(n), \Omega, p_{\text{nb}}, p_{\text{b}})$
8: $\quad\quad$**for** ℓ **in** Ω_{ctc} **do** ▷ Compute CTC prefix scores
9: $\quad\quad\quad p_{\text{prfx}}(\ell) \leftarrow p_{\text{nb}}(\ell) + p_{\text{b}}(\ell)$
10: $\quad\quad\quad \widehat{p}_{\text{prfx}}(\ell) \leftarrow \log p_{\text{prfx}}(\ell) + \alpha_0 \log p_{\text{LM}}(\ell) + \beta|\ell|$
11: $\quad\quad\widehat{\Omega} \leftarrow$ PRUNE$(\Omega_{\text{ctc}}, \widehat{p}_{\text{prfx}}, K, \theta_1)$
12: $\quad\quad$**for** ℓ **in** $\widehat{\Omega}$ **do** ▷ Delete old prefixes in Ω_{ta}
13: $\quad\quad\quad$**if** ℓ **in** Ω_{ta} **and** DCOND$(\ell, \widehat{\Omega}, p_{\text{ctc}})$ **then**
14: $\quad\quad\quad\quad$delete ℓ in Ω_{ta}
15: $\quad\quad$**for** ℓ **in** $\widehat{\Omega}$ **do** ▷ Compute transformer scores
16: $\quad\quad\quad$**if** ℓ **not in** Ω_{ta} **and** ACOND$(\ell, \widehat{\Omega}, p_{\text{ctc}})$ **then**
17: $\quad\quad\quad\quad p_{\text{ta}}(\ell) \leftarrow$ DECTA$(\boldsymbol{x}^E_{1:n+\varepsilon^{\text{dec}}}, \ell)$
18: $\quad\quad\quad\quad$add ℓ to Ω_{ta}
19: $\quad\quad$**for** ℓ **in** $\widehat{\Omega}$ **do** ▷ Compute joint scores
20: $\quad\quad\quad\widehat{\ell} \leftarrow \ell$ **if** ℓ **in** Ω_{ta} **else** $\ell_{:-1}$
21: $\quad\quad\quad p \leftarrow \lambda \log p_{\text{prfx}}(\ell) + (1-\lambda) \log p_{\text{ta}}(\widehat{\ell})$
22: $\quad\quad\quad p_{\text{joint}}(\ell) \leftarrow p + \alpha \log p_{\text{LM}}(\ell) + \beta|\ell|$
23: $\quad\quad\Omega \leftarrow$ MAX$(\widehat{\Omega}, p_{\text{joint}}, P)$
24: $\quad\quad\widehat{\Omega} \leftarrow$ PRUNE$(\widehat{\Omega}, \widehat{p}_{\text{prfx}}, P, \theta_2)$
25: $\quad\quad\Omega \leftarrow \Omega + \widehat{\Omega}$
26: $\quad\quad$remove from Ω_{ta} prefixes rejected due to pruning
27: $\quad$**return** MAX$(\widehat{\Omega}, p_{\text{joint}}, 1)$

where pos and i are the position and dimension indices [16].

2.4. Joint CTC-triggered attention decoding

Algorithm 1 shows the frame-synchronous one-pass decoding procedure for joint scoring of the CTC and transformer model outputs, which is similar to the decoding scheme described in [14]. The decoding algorithm is based on the frame-synchronous prefix beam search algorithm of [20], extending it by integrating the triggered attention decoder. The joint hypothesis set Ω and the TA hypothesis set Ω_{ta} are initialized in line 3 with the prefix sequence $\ell = (\langle\text{sos}\rangle,)$, where the symbol $\langle\text{sos}\rangle$ denotes the start of sentence label. The CTC prefix beam search algorithm of [20] maintains two separate probabilities for a prefix ending in blank p_{b} and not ending in blank p_{nb}, which are initialized in line 4. The initial TA scores p_{ta} are defined in line 5.

The frame-by-frame processing of the CTC posterior probability sequence p_{ctc} and the encoder state sequence X_E is shown from line 5 to 26, where $p_{\text{ctc}}(n)$ denotes the CTC posterior probability distribution at frame n. The function CTCPREFIX follows the CTC prefix beam search algorithm described in [20], which extends the set of prefixes Ω using the CTC posterior probabilities p_{ctc} of the current time step n and returns the separate CTC prefix scores p_{b} and p_{nb} as well as the newly proposed set of prefixes Ω_{ctc}. A local pruning threshold of 0.0001 is used by CTCPREFIX to ignore labels of lower CTC probability. Note that no language model or any pruning technique is used by CTCPREFIX, they will be incorporated in the following steps.

The prefix probabilities p_{prfx} and scores $\widehat{p}_{\text{prfx}}$ are computed in lines 9 and 10, where p_{LM} represents the language model (LM) probability and $|\ell|$ denotes the length of prefix sequence ℓ without counting the start of sentence label $\langle\text{sos}\rangle$. The function PRUNE prunes the set of CTC prefixes Ω_{ctc} in line 11 in two ways: first, the K most probable prefixes are selected based on $\widehat{p}_{\text{prfx}}$, then every prefix of score smaller than $\max(\widehat{p}_{\text{prfx}}) - \theta_1$ is discarded, with θ_1 being the beam width. The remaining set of prefixes is stored in $\widehat{\Omega}$. From line 12 to 14, prefixes are removed from the set Ω_{ta} if they satisfy a delete condition DCOND, and from line 15 to 18, TA scores are computed by function DECTA if an add condition ACOND returns "true". The delete and add conditions are used to delete "old" TA scores computed at a non-optimal frame position and to delay the computation of TA scores, if a new CTC prefix appeared at a supposedly too early time frame. The interested reader is referred to [14] for more details on both conditions. Note that our ASR experiments indicated that both conditions could be skipped without any WER degradation for the LibriSpeech task, which uses word-piece output labels, whereas their usage improves WERs for tasks like WSJ [21] with character-level label outputs. Joint CTC-TA scores, computed from line 19 to 22, are used to select the P most probable prefixes for further processing, which are stored in set Ω as shown in line 23. In line 24, the set of CTC prefixes $\widehat{\Omega}$ is further pruned to a maximum number of P prefixes with prefix scores within the beam width θ_2. Line 25 adds the CTC prefix set $\widehat{\Omega}$ to the best jointly scored prefix set Ω, and line 26 removes prefixes from Ω_{ta} that are no longer in Ω for the current and previous time steps. Finally, DECODE returns the prefix sequence of highest joint probability p_{joint} as shown in line 27.

3. EXPERIMENTS

3.1. Dataset

The LibriSpeech data set, which is a speech corpus of read English audio books [22], is used to benchmark ASR systems presented in this work. LibriSpeech is based on the open-source project LibriVox and provides about 960 hours of training data, 10.7 hours of development data, and 10.5 hours of test data, whereby the development and test data sets are both split into approximately two halves named "clean" and "other". The separation into clean and other is based on the quality of the recorded utterance, which was assessed using an ASR system [22].

3.2. Settings

Two transformer model sizes are used in this work: *small* and *large*. Parameter settings of the small transformer model are $d_{\text{model}} = 256$, $d_{\text{ff}} = 2048$, $d_h = 4$, $E = 12$, and $D = 6$, whereas the large transformer model uses $d_{\text{model}} = 512$ and $d_h = 8$ instead. The Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$ and learning rate scheduling similar to [16] is applied for training using 25000 warmup steps. The initial learning rate is set to 10.0 and the number of training epochs amounts to 100 for the small model and to 120 for the large model setup [3, 23]. The set of label outputs consists of 5000 subwords obtained by the SentencePiece method [24]. Weight factor γ, which is used to balance the CTC and transformer model objectives during training, is set to 0.3. Layer normalization is applied before and dropout with a rate of 10% after each MHA and FF layer. In addition, label smoothing with a penalty of 0.1 is used [25]. An RNN-based language model (LM) is employed via shallow fusion. The RNN-LM consists of 4 LSTM layers with 2048 units each trained using stochastic gradient descent and the official LM training text data of LibriSpeech [22].

The LM weight, CTC weight, and beam size of the full-sequence based joint CTC-attention decoding method are set to 0.7, 0.5, and

Table 1. WERs [%] of the full-sequence based CTC-transformer model. Results are shown for joint CTC-attention decoding [13], CTC prefix beam search decoding only [20], and attention beam search decoding only [3]. In addition, results for including the RNN-LM, for using data augmentation [25] as well as for the large transformer setup are shown.

	CTC-attention dec.				CTC beam search				Att. beam search			
	clean		other		clean		other		clean		other	
System	dev	test	dev	test	dev	test	dev	test	dev	test	dev	test
baseline	4.7	4.9	13.0	12.9	6.1	6.1	15.7	15.9	6.0	7.8	14.5	14.9
+RNN-LM	2.9	3.1	8.0	8.4	3.1	3.4	9.3	9.6	4.7	7.2	10.7	11.5
+SpecAug.	2.4	2.8	6.4	6.7	2.9	3.2	7.6	7.9	4.2	5.2	8.3	8.6
+large	2.4	2.7	6.0	6.1	2.5	2.8	6.9	7.0	4.1	5.0	7.9	8.0

20 for the small transformer model and to 0.6, 0.4, and 30 for the large model setup. The parameter settings for CTC prefix beam search decoding [20] are LM weight $\alpha_0 = 0.7$, pruning beam width $\theta_1 = 16.0$, insertion bonus $\beta = 2.0$, and pruning size $K = 30$. Parameters for joint CTC-TA decoding are CTC weight $\lambda = 0.5$, CTC LM weight $\alpha_0 = 0.7$, LM weight $\alpha = 0.5$, pruning beam width $\theta_1 = 16.0$, pruning beam width $\theta_2 = 6.0$, insertion bonus $\beta = 2.0$, pruning size $K = 300$, and pruning size $P = 30$. All decoding hyperparameter settings are determined using the development data sets of LibriSpeech.

3.3. Results

Table 1 presents ASR results of our transformer-based baseline systems, which are jointly trained with CTC to optimize training convergence and ASR accuracy [3, 13]. Results of different decoding methods are shown with and without using the RNN-LM, SpecAugment [25], and the large transformer model. Table 1 demonstrates that joint CTC-attention decoding provides significantly better ASR results compared to CTC or attention decoding alone, whereas CTC prefix beam search decoding attains lower WERs compared to attention beam search decoding, except for the dev-clean, dev-other, and test-other conditions when no LM is used. For attention beam search decoding, we normalize the log posterior probabilities of the transformer model and the RNN-LM scores when combining both using the hypothesis lengths [17]. Still our attention results are worse compared to the CTC results, which is unexpected but demonstrates that joint decoding stabilizes the transformer results.

Table 2 shows WERs of the full-sequence and the time-restricted self-attention encoder architectures combined with the CTC prefix beam search decoding method of [20] and our joint CTC-TA decoding method of Section 2.4, which are both algorithms for streaming recognition. Different encoder look-ahead settings are compared using $\varepsilon^{\text{enc}} = 0, 1, 2, 3$, and ∞, where each consumed frame of the self-attention encoder corresponds to 40 ms of input due to the output frame rate of EncCNN. Since such look-ahead is applied at every encoder layer ($E = 12$), the theoretical latency caused by the time-restricted self-attention encoder amounts to $E \times \varepsilon^{\text{enc}} \times 40$ ms, i.e., to 0 ms ($\varepsilon^{\text{enc}} = 0$), 480 ms ($\varepsilon^{\text{enc}} = 1$), 960 ms ($\varepsilon^{\text{enc}} = 2$), and 1440 ms ($\varepsilon^{\text{enc}} = 3$), respectively. The CTC prefix beam search decoding results of Table 2 show that increasing ε^{enc} significantly improves the ASR accuracy, e.g., test-other WER drops from 9.4% to 7.0% when moving from 0 to ∞ (full-sequence) encoder look-ahead frames. The influence of different TA decoder settings are compared in Table 2 as well, using $\varepsilon^{\text{dec}} = 6, 12$, and 18 look-ahead frames. Note that unlike the encoder, the total decoder delay

Table 2. WERs [%] for different ε^{enc} settings of the time-restricted encoder using the CTC prefix beam search decoding method of [20] as well our proposed joint CTC-TA decoding method of Section 2.4 with different ε^{dec} configurations. SpecAugment [25], the RNN-LM, and the large transformer are applied for all systems.[1]

	CTC beam search				TA: $\varepsilon^{\text{dec}} = 6$				TA: $\varepsilon^{\text{dec}} = 12$				TA: $\varepsilon^{\text{dec}} = 18$			
	clean		other		clean		other		clean		other		clean		other	
ε^{enc}	dev	test	dev	test	dev	test	dev	test	dev	test	dev	test	dev	test	dev	test
0	3.3	3.7	9.4	9.4	3.2	3.3	8.4	8.6	3.0	3.4	8.4	8.5	2.9	3.2	8.1	8.0
1	3.0	3.3	8.4	8.6	2.9	3.1	7.8	8.1	2.8	3.1	7.5	8.1	2.8	3.0	7.5	7.8
2	2.9	3.1	8.0	8.2	2.8	2.9	7.4	7.8	2.7	2.9	7.2	7.6	2.7	2.9	7.3	7.4
3	2.8	2.9	7.8	8.1	2.7	2.8	7.2	7.4	2.7	2.8	7.2	7.3	2.7	2.8	7.1	7.2
∞	2.5	2.8	6.9	7.0	2.5	2.7	6.3	6.5	2.5	2.7	6.3	6.4	2.4	2.6	6.1	6.3

does not grow with its depth, since each decoder layer is attending to the encoder output sequence X_E. Thus, the TA decoder delay amounts to $\varepsilon^{\text{dec}} \times 40$ ms, i.e., to 240 ms ($\varepsilon^{\text{dec}} = 6$), 480 ms ($\varepsilon^{\text{dec}} = 12$), and 720 ms ($\varepsilon^{\text{dec}} = 18$), respectively. Results show that joint CTC-TA decoding consistently improves WERs compared to CTC prefix beam search decoding, while for larger look-ahead values WERs are approaching the full-sequence CTC-attention decoding results, which can be noticed by comparing results of the $\varepsilon^{\text{enc}} = \infty$, $\varepsilon^{\text{dec}} = 18$ TA system setup with the full-sequence CTC-attention system of Table 1.

The best streaming ASR system of Table 2 achieves a WER of 2.8% and 7.2% for the test-clean and test-other conditions of LibriSpeech with an overall processing delay of 30 ms (EncCNN) + 1440 ms (EncSA: $\varepsilon^{\text{enc}} = 3$) + 720 ms (DecTA: $\varepsilon^{\text{dec}} = 18$) = 2190 ms. For $\varepsilon^{\text{enc}} = 1$ and $\varepsilon^{\text{dec}} = 18$, the test-clean and test-other WERs amount to 3.0% and 7.8%, respectively, with a total delay of 1230 ms, which provides a good trade-off between accuracy and latency. It should be noted that a lattice-based CTC-TA decoding implementation can output intermediate CTC prefix beam search results, which are updated after joint scoring with the TA decoder, and thus the perceived latency of such an implementation will be on average smaller than its theoretical latency and close to that of the encoder alone. However, a thorough study of the user perceived latency remains to be done in future work.

4. CONCLUSIONS

In this paper, a fully streaming end-to-end ASR system based on the transformer architecture is proposed. Time-restricted self-attention is applied to control the latency of the encoder and the triggered attention (TA) concept to control the output latency of the decoder. For streaming recognition and joint CTC-transformer model scoring, a frame-synchronous one-pass decoding algorithm is applied, which demonstrated similar LibriSpeech ASR results compared to full-sequence based CTC-attention as the number of look-ahead frames is increased. Combined with the time-restricted self-attention encoder, our proposed TA-based streaming ASR system achieved WERs of 2.8% and 7.2% for the test-clean and test-other data sets of LibriSpeech, which to our knowledge is the best published LibriSpeech result of a fully streaming end-to-end ASR system.

[1] Note that results shown here are updated compared to our ICASSP submission.

5. REFERENCES

[1] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. ISCA Interspeech*, Sep. 2016, pp. 2751–2755.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. 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 Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[3] S. Karita, N. Yalta, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *Proc. ISCA Interspeech*, Sep. 2019, pp. 1408–1412.

[4] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. International Conference on Machine Learning (ICML)*, vol. 148, Jun. 2006, pp. 369–376.

[5] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:abs/1211.3711*, 2012.

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:abs/1409.0473*, 2014.

[7] J. Schalkwyk, "An all-neural on-device speech recognizer," Mar. 2019, url: https://ai.googleblog.com/2019/03/an-all-neural-on-device-speech.html.

[8] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving RNN transducer modeling for end-to-end speech recognition," *arXiv preprint arXiv:abs/1909.12415*, 2019.

[9] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly, "A comparison of sequence-to-sequence models for speech recognition," in *Proc. ISCA Interspeech*, Sep. 2017, pp. 939–943.

[10] T. N. Sainath, C. Chiu, R. Prabhavalkar, A. Kannan, Y. Wu, P. Nguyen, and Z. Chen, "Improving the performance of online neural transducer models," *arXiv preprint arXiv:abs/1712.01807*, 2017.

[11] C. Chiu and C. Raffel, "Monotonic chunkwise attention," in *Proc. International Conference on Learning Representations (ICLR)*, Apr. 2018.

[12] N. Moritz, T. Hori, and J. Le Roux, "Triggered attention for end-to-end speech recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 5666–5670.

[13] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid CTC/attention architecture for end-to-end speech recognition," *J. Sel. Topics Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[14] N. Moritz, T. Hori, and J. Le Roux, "Streaming end-to-end speech recognition with joint CTC-attention based models," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec. 2019, pp. 936–943.

[15] N. Moritz, T. Hori, and J. Le Roux, "Unidirectional neural network architectures for end-to-end automatic speech recognition," in *Proc. ISCA Interspeech*, Sep. 2019, pp. 76–80.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, Dec. 2017, pp. 6000–6010.

[17] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs RNN in speech applications," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec. 2019.

[18] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM," in *Proc. ISCA Interspeech*, Aug. 2017, pp. 949–953.

[19] D. Povey, H. Hadian, P. Ghahremani, K. Li, and S. Khudanpur, "A time-restricted self-attention layer for ASR," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5874–5878.

[20] A. L. Maas, A. Y. Hannun, D. Jurafsky, and A. Y. Ng, "First-pass large vocabulary continuous speech recognition using bi-directional recurrent DNNs," *arXiv preprint arXiv:1408.2873*, 2014.

[21] "CSR-II (WSJ1) complete," vol. LDC94S13A. Philadelphia: Linguistic Data Consortium, 1994.

[22] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2015, pp. 5206–5210.

[23] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5884–5888.

[24] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:abs/1808.06226*, 2018.

[25] 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," *arXiv preprint arXiv:abs/1904.08779*, 2019.