INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

VAD-free Streaming Hybrid CTC/Attention ASR for Unsegmented Recording

Hirofumi Inaguma, Tatsuya Kawahara

Graduate School of Informatics, Kyoto University, Kyoto, Japan
{inaguma,kawahara}@sap.ist.i.kyoto-u.ac.jp

Abstract

In this work, we propose novel decoding algorithms to enable streaming automatic speech recognition (ASR) on unsegmented long-form recordings without voice activity detection (VAD), based on monotonic chunkwise attention (MoChA) with an auxiliary connectionist temporal classification (CTC) objective. We propose a *block-synchronous* beam search decoding to take advantage of efficient batched output-synchronous and low-latency input-synchronous searches. We also propose a VAD-free inference algorithm that leverages CTC probabilities to determine a suitable timing to reset the model states to tackle the vulnerability to long-form data. Experimental evaluations demonstrate that the block-synchronous decoding achieves comparable accuracy to the label-synchronous one. Moreover, the VAD-free inference can recognize long-form speech robustly for up to a few hours.

Index Terms: Streaming automatic speech recognition, monotonic chunkwise attention, CTC, voice activity detection

1. Introduction

Recent progress of end-to-end (E2E) automatic speech recognition (ASR) enables us to build competitive systems to conventional hybrid systems with much smaller development efforts. For live streaming applications, frame-synchronous models such as connectionist temporal classification (CTC) [1] and RNN transducer (RNN-T) [2] are promising approaches because of the robustness for long-form speech [3, 4]. Attention-based encoder-decoder (AED) [5, 6] have shown outstanding performances in the offline task [7–9] and have been intensively investigated for streaming extensions [10–13]. Among them, monotonic chunkwise attention (MoChA) [11] is attractive because of the monotonic constraint of alignments and linear-time decoding complexity at test time. The notable advantages of MoChA over frame-synchronous models are faster decoding thanks to label-wise predictions [14] and availability of large vocabularies because of lower memory consumption. Moreover, MoChA does not have to expand hypotheses over a silence region because it treats silence in the internal attention module.

However, the generalization capability of the AED models to long-form speech is poor [4, 15], and how to mitigate this problem is still an open question. Several methods have tackled this problem by incorporating alignment information to the training as supervision [14, 16, 17], window-based overlapped offline inference [4, 18], modifying LSTM encoder states [3], and adopting new architecture [12, 15]. It is also a common practice to segment long-form audio with a separate voice activity detection (VAD) model in advance [19]. CTC can also be used for that purpose [20–22]. Joint end-pointing was also investigated for RNN-T in the voice search task [23–25].

In this work, we propose novel decoding algorithms to recognize speech of unlimited length in a streaming way with MoChA trained jointly with an auxiliary CTC loss [26, 27]. Instead of pursuing better generalization to long-form speech from a training perspective, we seek a solution to find a suitable timing to reset the model states from a decoding perspective. Firstly, we propose a *block-synchronous* beam search decoding, in which the advantages of breadth- and best-first searches are taken to achieve efficient batched inference and low display latency.[1] We allow continuing search within a block by relaxing the label-synchronous hypothesis pruning to consider (potentially) various lengths of candidates in the beam given a partial observation. Secondly, we propose a VAD-free streaming inference algorithm leveraging CTC probabilities to determine a timing to reset the states. Instead of performing audio segmentation before ASR, our method recognizes all speech, including silence frames, and therefore the ASR model does not have to wait for the segmentation to be completed. Moreover, the unified framework is suitable for context management and on-device applications. Although our base model is MoChA, the proposed VAD-free inference can also be applied to any streaming ASR model trained jointly with the CTC objective.

Experimental evaluations on English and Japanese lecture corpora demonstrate that the block-synchronous decoding achieves comparable accuracy to the label-synchronous decoding and even outperforms it in some cases. We also show that the VAD-free inference does not degrade accuracy so much without the ground-truth segmentation and achieves better performance than cascading an external VAD model.

2. Streaming Hybrid CTC/Attention ASR

MoChA extended hard monotonic attention (HMA) [10] by equipping an additional chunkwise soft attention module restricted to local w frames. To generate the i-th token with a linear-time complexity at test time, HMA introduces a discrete decision $z_{i,j} \in \{0, 1\}$ (j: encoder time index) and samples it from a Bernoulli random variable, Bernoulli$(p_{i,j})$, where $p_{i,j} \in [0, 1]$ is a selection probability as a function of encoder and decoder outputs. To enable the backpropagation training, the expected alignment score $\alpha_{i,j}$ is calculated with $p_{i,j}$ by considering all alignment paths as

$$\alpha_{i,j} = p_{i,j}\left((1 - p_{i,j-1})\frac{\alpha_{i,j-1}}{p_{i,j-1}} + \alpha_{i-1,j}\right). \quad (1)$$

The chunkwise attention score for a context vector is calculated with $\alpha_{i,j}$ at training time and with discrete indices at test time. We also apply *StableEmit* [32] to reduce the emission latency by multiplying $p_{i,j}$ in Eq. (1) by a constant factor $1 - \lambda_{\text{se}}$ ($\lambda_{\text{se}} > 0$) during training.

Applying an auxiliary CTC loss $\mathcal{L}_{\text{ctc}}$ on top of the encoder of AED models is effective in encouraging the decoder to learn

[1] We define display latency as a delay to emit tokens caused by the search procedure and distinguish it from the emission latency [14, 16, 25, 28–31], which is caused by the E2E model training. We have already tackled to reduce the emission latency of MoChA in [14, 32].

http://dx.doi.org/10.21437/Interspeech.2021-1107

a monotonic alignment [26, 27, 33, 34]. We also introduce a quantity loss $\mathcal{L}_{\text{qua}}$ [16] or a CTC-synchronous training (CTC-ST) loss $\mathcal{L}_{\text{sync}}$ [17] to improve the performance and reduce the emission latency [14, 32]. The total objective $\mathcal{L}_{\text{total}}$ is formulated as

$$\mathcal{L}_{\text{total}} = (1-\lambda_{\text{ctc}})\mathcal{L}_{\text{mocha}} + \lambda_{\text{ctc}}\mathcal{L}_{\text{ctc}} + \lambda_{\text{qua}}\mathcal{L}_{\text{qua}} + \lambda_{\text{sync}}\mathcal{L}_{\text{sync}}, \quad (2)$$

where λ_* is a corresponding task weight. Unlike [27], we do not perform joint CTC decoding during beam search because they were not helpful in our experiments.

3. Efficient block-synchronous decoding

When using MoChA, beam search decoding is conducted in a label-synchronous way (i.e., breadth-first search) at test time to find the most probable output sequence. However, active hypotheses in the current beam are pruned after the expansion when and only when (1) all the hypotheses find the next token boundaries (i.e., j s.t. $z_{i,j}=1$) or (2) their pointers to the encoder outputs reach the last encoder output observed so far. In other words, all the active hypotheses must have the same output sequence length at each output step. Therefore, the search cannot proceed forward if some active hypotheses fail to detect the next boundaries correctly, even when a new acoustic observation comes in. Moreover, when applying subword tokenization (e.g., byte pair encoding (BPE) [35]) to word sequences, hypotheses in the beam could have different lengths even when they correspond to the same word sequence. This problem becomes more serious when recognizing long-form speech, resulting in a non-negligible recognition delay in the online streaming scenario. So our goal is to continue sequence generation as long as some of the active hypotheses detect the subsequent token boundaries over the current acoustic observation.

3.1. Proposed algorithm

To perform beam search with a minimal display latency given a partial acoustic observation, we propose an efficient *block-synchronous* beam search decoding for MoChA, which relaxes the constraint of the label-synchronous hypothesis pruning. The block-synchronous decoding combines the advantages of breadth- and best-first searches, efficient batched computation [36] and small display latency. The proposed algorithm is shown in Algorithm 1. Given the m-th input block x^m of a fixed length T_{block} [10ms], we perform the breadth-first search over the corresponding encoder outputs h^m of length T'_{block} ($< T_{\text{block}}$). Unlike label-synchronous decoding, however, active hypotheses in the current beam are forcibly pruned no matter whether all active hypotheses detect the next boundaries in h^m. The search in the m-th block continues until (1) none of the active hypotheses find any further boundary in h^m (*line:5*) or (2) the number of generated tokens in h^m surpasses $U_{\max}(= T'_{\text{block}} \times R_{\text{len}})$.[2]

Let Ω_+ be a set of hypotheses having a possibility to detect the next boundary in h^m. Hypotheses in Ω_+ are added to Ω_- without prefix expansion if any next boundary is not detected in h^m (*line:12*). In this case, we allow to generate $\langle\text{eos}\rangle$ only because of MoChA's behavior (see Algorithm 1 in [11]). Otherwise, it is expanded by the top-k tokens and is added to Ω_{next} (*line:18*). When $\langle\text{eos}\rangle$ is generated, the hypothesis is added to a complete hypothesis set Ω_{eos} instead (*line:16*). Pruning is conducted over Ω_{next}[3] with a beam width B at every output step (*line:22*). To avoid biasing to shorter hypotheses because of the monotonic decrease of sequence-level log probabilities, we normalize them by the current output sequence length (*length normalization* [37]).

Algorithm 1 Block-synchronous beam search decoding with MoChA at the m-th block

1: **function** BLOCKSYNC($h^m, \Omega_+, \Omega_{\text{eos}}, B, R_{\text{len}}$)
2: $U_{\max} \leftarrow |h^m| \times R_{\text{len}}, \Omega_- \leftarrow \{\}$
3: **for** $i = 1, \ldots, U_{\max}$ **do**
4: **if** $|\Omega_+| = 0$ **then**
5: **break** ▷ Move to the next block
6: **end if**
7: $p_{\text{mocha},i} = \text{Decoder}(\Omega_+, [h^{m-1}_{-(w-1):}; h^m])$
8: $score = \log p_{\text{mocha},i} + \lambda_{\text{lm}} \log p_{\text{lm},i}$ ▷ Normalize by length
9: $\Omega_{\text{next}} \leftarrow \{\}$
10: **for** y in Ω_+ **do**
11: **if** $\sum_j z_{i,j} = 0$ **then**
12: add y to Ω_- ▷ No boundary detected
13: **end if**
14: **for** $k \in \mathcal{V}$ **do**
15: **if** $k = \langle\text{eos}\rangle$ **then**
16: add y to Ω_{eos}
17: **else**
18: add $y + [k]$ to Ω_{next}
19: **end if**
20: **end for**
21: **end for**
22: $\Omega_+ \leftarrow$ top-B in Ω_{next} ▷ Pruning
23: **end for**
24: $\Omega_+ \leftarrow \Omega_+ \cup \Omega_-$
25: **return** $(\Omega_+, \Omega_{\text{eos}})$
26: **end function**

The entire search process for an utterance or a session is finalized when all pointers to h^m in Ω_+ reach the last encoder output, $h^m_{|h^m|-1}$. The details will be described in Section 4. The search is equivalent to frame-synchronous and label-synchronous decoding by setting T'_{block} to 1 and ∞, respectively. Therefore, the proposed algorithm is a generalized form of both search methods.

Concurrently to this work, streaming block-synchronous decoding with an offline Transformer decoder was proposed in [13]. They determined to move to the next block when generating $\langle\text{eos}\rangle$ at the current block and introduced complicated heuristics to avoid token repetition. The decoding complexity was quadratic of the input length because of incremental decoding. In contrast, our method can move to the next block without generating $\langle\text{eos}\rangle$ in each block because MoChA has a function of detecting a token boundary at *each frame*. This also saves a computation of the softmax normalization in the output layer. Moreover, the decoding complexity of our method is linear, and we do not perform joint CTC decoding.

4. VAD-free streaming inference

In the AED models, it is crucial to keep decoding contexts to improve the recognition performance because of the conditional dependency of output symbols. However, it is required to exclude long samples from the training data to fit the GPU/TPU memory and perform efficient training. Therefore, when recognizing long-form speech during inference, the models must generalize to unseen samples, but it is challenging in general [4].

[2] R_{len} is a hyperparameter to control the maximum output length in each block, but we did not observe token repetition.

[3] This is more effective than pruning over $\Omega_{\text{next}} \cup \Omega_-$.

4.1. Proposed algorithm

To balance the limitation of the generalization capability to long-form speech and the effective context management, we determine a suitable timing to reset model states based on CTC probabilities. We regard consecutive blank tokens ($\varnothing$) generated from the CTC branch as a silence region and use them to find a *reset point*. Unlike a CTC-based pre-segmentation in [20], our method does not perform the segmentation explicitly, i.e., we do not detect the onset. Instead, we recognize all frames including long silence. Therefore, we do not have to wait for the pre-segmentation to be completed to start recognition, leading to latency reduction.

The proposed algorithm is shown in Algorithm 2. We adopt the block-synchronous decoding in Section 3. We count the number of consecutive blank tokens $n_{\varnothing}$ from the previous reset point and detect the next reset point when $n_{\varnothing}$ surpasses a threshold $N_{\varnothing}$ (**condition 1**, *line:18*). We also regard a weak non-blank spike whose probability is less than P_{spike} as a blank token. Note that the recognition continues until the end of the current block regardless of the result of the reset point detection. Moreover, we also allow resetting the states when $\langle\text{eos}\rangle$ is generated (**condition 2**, *line:22*). This is important for determining the reset point when enough silence is not found for a while. Once a reset point is detected, we push the most probable hypothesis in Ω_+ to a session-level hypothesis set Ω_{session} and reset both decoder states and Ω_+ (*line:26-27*). When using LSTM encoders, we also reset the encoder states. To deal with speech frames around the block boundaries, we re-encode acoustic features in the previous block after the state reset and use the last states as the initial states in the current block (*back-off initialization*). When using Conformer encoders [38], however, we do not have to reset the encoder states because they are agnostic to input offsets thanks to relative positional encoding and time-restricted self-attention. To avoid frequent state resets, we introduce a safeguard in which the reset point detection is prohibited until the total number of input frames t from the previous reset point accumulates up to a threshold N_{sg} [10ms] (*line:9*). Moreover, LSTM LM states are carried over to the next block to provide useful contexts before the reset point.[4]

Algorithm 2 VAD-free streaming inference

1: **function** DECODE($x, N_{\text{sg}}, N_{\varnothing}, P_{\text{spike}}, R_{\text{len}}$)
2: $t \leftarrow 0, n_{\varnothing} \leftarrow 0$
3: $IsReset \leftarrow False$
4: $\Omega_{\text{session}} \leftarrow \{\}, \Omega_{+} \leftarrow \{\}, \Omega_{\text{eos}} \leftarrow \{\}$
5: **for** $m = 1, \cdots, M$ **do**
6: $h^m \leftarrow \text{Encode}(x^m, IsReset)$
7: $\Omega_{+}, \Omega_{\text{eos}} \leftarrow \text{BlockSync}(h^m, \Omega_{+}, \Omega_{\text{eos}}, B, R_{\text{len}})$
8: $t \leftarrow t + |x^m|$
9: **if** $t \geq N_{\text{sg}}$ **then** ▷ Safeguard
10: $p^{\text{ctc}} \leftarrow \text{CTC}(h^m)$
11: **for** $j = 1, \cdots, |h^m|$ **do**
12: **if** $\underset{k}{\text{argmax}}\, p^{\text{ctc}}_{j,k} = \varnothing$ or $\underset{k}{\max}\, p^{\text{ctc}}_{j,k} < P_{\text{spike}}$ **then**
13: $n_{\varnothing} \leftarrow n_{\varnothing} + 1$
14: **else**
15: $n_{\varnothing} \leftarrow 0$
16: **end if**
17: **if** $n_{\varnothing} \geq N_{\varnothing}$ **then**
18: $IsReset \leftarrow True$ ▷ Condition: 1
19: **end if**
20: **end for**
21: **if** $LastToken(\text{argmax}(\Omega_{+} \cup \Omega_{\text{eos}})) = \langle\text{eos}\rangle$ **then**
22: $IsReset \leftarrow True$ ▷ Condition: 2
23: **end if**
24: **end if**
25: **if** $IsReset$ **then**
26: $\Omega_{\text{session}}.push(\text{argmax}(\Omega_{+} \cup \Omega_{\text{eos}}))$
27: $t \leftarrow 0, n_{\varnothing} \leftarrow 0, \Omega_{+} \leftarrow \{\}$ ▷ Reset states
28: $IsReset \leftarrow False$
29: **end if**
30: **end for**
31: **return** Ω_{session}
32: **end function**

5. Experimental evaluations

5.1. Experimental setup

We used the TEDLIUM release v2 (TEDLIUM2) [39] and the Corpus of Spontaneous Japanese (CSJ) [40]. TEDLIUM2 consists of about 210-hour English lecture speech. CSJ consists of about 600-hour Japanese spontaneous academic lecture speech. We combined three official test sets in CSJ: *eval1*, *eval2*, and *eval3* to construct the *test* set. We extracted 80-channel log-mel filterbank coefficients computed with a 25-ms window shifted every 10ms with Kaldi [41].

We investigated three kinds of encoder architectures; unidirectional LSTM (UniLSTM), latency-controlled bidirectional LSTM (LC-BLSTM) [42], and latency-controlled Conformer (LC-Conformer) encoders [43]. The UniLSTM consisted of five layers of LSTM with 1024 units. The LC-BLSTM had 512 units in both directions. We set both the current and right block sizes to 40, i.e., 400ms. The LC-Conformer encoder had the same architecture as Conformer (M) [38] with a kernel size of 15, while the number of layers was reduced from 16 to 12. We used hierarchical downsampling [44] with the max-pooling having a stride of 2 at the last frontend CNN layer, 4th, and 8th Conformer blocks. We also replaced batch normalization in each convolution module with layer normalization [32]. We adopted the masking strategy in [43], where lookahead frames were truncated in the same block, including the frontend CNN layers. We set the left and current block sizes to 960ms and 320ms, respectively. Therefore, the average algorithmic latency (AAL) of the UniLSTM, LC-BLSTM, and LC-Conformer encoders was 60ms, 660ms, and 160ms, respectively. The decoder consisted of a single layer of LSTM with 1024 units. We set a chunk size w of MoChA to 4.

We applied CTC-ST [17] to all LSTM models with $\lambda_{\text{sync}} = 1.0$ in Eq. (2). Moreover, StableEmit [32] was applied to the UniLSTM and Conformer models on TEDLIUM2 with $\lambda_{\text{se}} = 0.1$ and $\lambda_{\text{qua}} = 2.0$. During inference, we used $B = 10$ with a four-layer LSTM LM. We set (N_{sg}, $N_{\varnothing}$, P_{spike}, R_{len}) to (1600, 40, 0.1, 1.0). The codes are publicly available.[5]

5.2. Results

5.2.1. Utterance-level evaluation

We first compare the type of beam search decoding with the ground-truth segmentation in Table 1. Note that we did not use CTC probabilities to reset the model states here. Using the UniLSTM encoder, we confirmed comparable WER with the block-synchronous decoding compared to the label-synchronous one on the TEDLIUIM2 *dev* set while it was slightly degraded on the *test* set. Minimum WER training [45] could mitigate the degradation. In contrast, we observed WER reduction down to the block size of 240ms and 400ms on the CSJ *dev* and *test* sets, respectively. A possible explanation is that the block-synchronous decoding introduced some effective

[4] We trained LSTM LM by carrying over the last state in a mini-batch to the initial state in the next mini-batch during LM training.

[5] https://github.com/hirofumi0810/neural_sp.

Table 1: *Comparison of beam search type for MoChA with the ground-truth segmentation*

Encoder	Output synchronization	T_{block}	WER [%] (↓)			
			TEDLIUM2		CSJ	
			dev	test	dev	test
UniLSTM	Label	∞	11.7	10.9	6.6	7.5
	Block	64	**11.6**	11.7	**6.4**	**7.3**
	Block	40	**11.6**	11.6	**6.4**	**7.4**
	Block	32	**11.6**	12.0	**6.5**	**7.5**
	Block	24	11.8	12.2	**6.5**	7.6
	Block	16	12.0	12.6	**6.6**	7.7
	Frame	4	13.2	13.6	8.0	9.7
LC-BLSTM	Label	∞	10.3	8.6	5.9	6.5
	Block	40	10.4	**8.6**	**5.6**	**6.3**
	Frame	4	10.4	**8.6**	**5.6**	**6.3**
LC-Conformer	Label	∞	9.3	9.0	–	–
	Block	32	9.4	**8.9**	–	–
	Frame	8	**9.3**	**9.0**	–	–

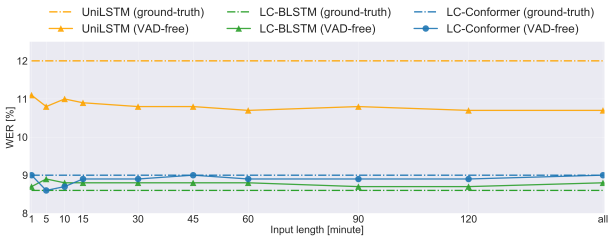

Figure 1: *WER on simulated long-form recordings on the TEDLIUM2 test set. All models used block-synchronous decoding. The UniLSTM model used* $T_{\text{block}} = 32$.

inductive biases to the search process.

On the other hand, regarding LC-BLSTM and LC-Conformer encoders, we observed similar WERs on TEDLIUM2 and better WERs on CSJ, even with the frame-synchronous decoding. This indicates that we can manage the display latency without WER degradation by leveraging future information in the encoder.

5.2.2. *Long-form simulation with utterance concatenation*

We next simulate variable lengths of long-form recordings to demonstrate the robustness of the proposed VAD-free inference algorithm. We first sorted utterances in the evaluation set by the timestamps and concatenated adjacent utterances in a greedy way until the total length reached a certain length. Therefore, multiple speakers could appear in a concatenated utterance. We used $T_{\text{block}} = 32$ for the UniLSTM encoder. We present the results on TEDLIUM2 in Figure 1. The horizontal dot lines denote WERs of the block-synchronous decoding with the ground-truth segmentation. We confirmed that the VAD-free inference was robust to long-form speech regardless of the encoder type and input lengths. Interestingly, the performance of the UniLSTM encoder was significantly improved from the ground-truth segmentation for all the lengths.

5.2.3. *Ablation study*

We conduct the ablation study of the VAD-free inference algorithm on TEDLIUM2 in Table 2. We concatenated all utterances in each evaluation set. We observed that the safeguard had the largest impact, indicating that MoChA is more likely to generate ⟨eos⟩ at the end of blocks with the block-synchronous decoding. Length normalization was also important for longer hypotheses to rank at the top. Back-off initialization was important for the LSTM encoder to deal with frames around a reset point better. Other techniques were slightly but consistently helpful.

Table 2: *Ablation study with UniLSTM MoChA on simulated long-form recordings of TEDLIUM2 (all concatenated)*

Decoding	WER [%] (↓)	
	dev (1.59h)	test (2.61h)
Block ($T_{\text{block}} = 32$)	**11.4**	**10.7**
w/o length normalization	13.3	12.9
w/o LM state carryover	11.6	**10.7**
w/o safeguard	19.7	19.7
w/o condition2	11.7	10.9
w/o back-off initialization	12.9	12.0

Table 3: *Session-level results with real long-form recordings*

Encoder	T_{block}	VAD	WER [%] (↓)		
			TED LIUM2 test	IWSLT tst20{13/14/15}	CSJ test
UniLSTM	32	✗	**10.9**	**18.8/16.7/31.9**	**8.3**
LC-BLSTM	40	✗	**8.8**	**16.9/15.1/29.7**	**8.7**
LC-Conformer	32	✗	**8.9**	**16.4/15.1/30.1**	–
UniLSTM	32	✓	26.1	32.2/29.9/43.9	13.1
LC-BLSTM	40	✓	18.3	25.5/23.0/37.1	11.9
LC-Conformer	32	✓	34.9	37.3/36.2/43.5	–

5.2.4. *Session-level evaluation on real long-form recordings*

Finally, we conduct experiments on the real session-level lecture recordings. Unlike the simulated experiments, there exist a lot of silence frames in the real recordings. We compared the proposed method with the pre-segmentation with WebRTC VAD[6]. Because both corpora do not necessarily contain all transcriptions in a session, we recognized all frames but removed tokens that did not match the ground-truth segments to calculate WER. We also evaluated the models trained on TEDLIUM2 with the IWSLT *tst2013*, *tst2014*, and *tst2015* sets [46]. Results in Table 3 showed that the proposed VAD-free inference achieved comparative WERs to those with the ground-truth segmentation on TEDLIUM2. Although the results on CSJ degraded slightly compared to when using the ground-truth segmentation, they were much better than cascading VAD and ASR models. Moreover, the degradation was much smaller than the CTC-based pre-segmentation [20,21], which had a large latency to start recognition. We observed that the VAD model was more likely to generate short segments that did not suit E2E models, especially for the LC-Conformer encoder. This was because the LC-Conformer encoder had a total history context of 11.52 seconds. Although there is room for improving VAD, the proposed unified framework eliminates the need for developing separate models independently.

6. Conclusions

In this work, we have proposed the block-synchronous beam search decoding and the VAD-free inference algorithm to recognize unsegmented long-form speech with the hybrid CTC/MoChA framework. Experimental evaluations on English and Japanese lecture corpora demonstrated that the proposed decoding method enabled stable recognition of long-form speech with a linear-time decoding complexity. It was more accurate than performing VAD with an external model.

[6] `https://github.com/wiseman/py-webrtcvad`

7. References

[1] A. Graves, S. Fernández, 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, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[3] A. Narayanan, R. Prabhavalkar, C.-C. Chiu, D. Rybach, T. N. Sainath, and T. Strohman, "Recognizing long-form speech using streaming end-to-end models," in *Proc. of ASRU*, 2019, pp. 920–927.

[4] C.-C. Chiu, W. Han, Y. Zhang, R. Pang, S. Kishchenko, P. Nguyen, A. Narayanan, H. Liao, S. Zhang, A. Kannan *et al.*, "A comparison of end-to-end models for long-form speech recognition," in *Proc. of ASRU*, 2019, pp. 889–896.

[5] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. of NIPS*, 2015, pp. 577–585.

[6] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. of ICASSP*, 2016, pp. 4960–4964.

[7] 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. of Interspeech*, 2017, pp. 939–943.

[8] E. Battenberg, J. Chen, R. Child, A. Coates, Y. G. Y. Li, H. Liu, S. Satheesh, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," in *Proc. of ASRU*, 2017, pp. 206–213.

[9] J. Li, Y. Wu, Y. Gaur, C. Wang, R. Zhao, and S. Liu, "On the comparison of popular end-to-end models for large scale speech recognition," in *Proc. of Interspeech*, 2020, pp. 1–5.

[10] C. Raffel, M.-T. Luong, P. J. Liu, R. J. Weiss, and D. Eck, "Online and linear-time attention by enforcing monotonic alignments," in *Proc. of ICML*, 2017, pp. 2837–2846.

[11] C.-C. Chiu and C. Raffel, "Monotonic chunkwise attention," in *Proc. of ICLR*, 2018.

[12] L. Dong and B. Xu, "CIF: Continuous integrate-and-fire for end-to-end speech recognition," in *Proc. of ICASSP*, 2020, pp. 6079–6083.

[13] E. Tsunoo, Y. Kashiwagi, and S. Watanabe, "Streaming Transformer ASR with blockwise synchronous beam search," in *Proc. of SLT*, 2020, pp. 22–29.

[14] H. Inaguma and T. Kawahara, "Alignment knowledge distillation for online streaming attention-based speech recognition," *arXiv preprint arXiv:2103.00422*, 2021.

[15] L. Dong, C. Yi, J. Wang, S. Zhou, S. Xu, X. Jia, and B. Xu, "A comparison of label-synchronous and frame-synchronous end-to-end models for speech recognition," *arXiv preprint arXiv:2005.10113*, 2020.

[16] H. Inaguma, Y. Gaur, L. Lu, J. Li, , and Y. Gong, "Minimum latency training strategies for streaming sequence-to-sequence ASR," in *Proc. of ICASSP*, 2020, pp. 6064–6068.

[17] H. Inaguma, M. Mimura, and T. Kawahara, "CTC-synchronous training for monotonic attention model," in *Proc. of Interspeech*, 2020, pp. 571–575.

[18] T. G. Kang, H.-G. Kim, M.-J. Lee, J. Lee, and H. Lee, "Partially overlapped inference for long-form speech recognition," in *Proc. of ICASSP*. IEEE, 2021, pp. 5989–5993.

[19] T. Hughes and K. Mierle, "Recurrent neural networks for voice activity detection," in *Proc. of ICASSP*, 2013, pp. 7378–7382.

[20] T. Yoshimura, T. Hayashi, K. Takeda, and S. Watanabe, "End-to-end automatic speech recognition integrated with CTC-based voice activity detection," in *Proc. of ICASSP*, 2020, pp. 6999–7003.

[21] Y. Fujita, S. Watanabe, and M. Omachi, "End-to-end ASR and audio segmentation with non-autoregressive insertion-based model," *arXiv preprint arXiv:2012.10128*, 2020.

[22] M. Li, S. Zhou, and B. Xu, "Long-running speech recognizer: An end-to-end multi-task learning framework for online ASR and VAD," *arXiv preprint arXiv:2103.01661*, 2021.

[23] S.-Y. Chang, R. Prabhavalkar, Y. He, T. N. Sainath, and G. Simko, "Joint endpointing and decoding with end-to-end models," in *Proc. of ICASSP*, 2019, pp. 5626–5630.

[24] 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. of ICASSP*, 2020, pp. 6069–6073.

[25] J. Mahadeokar, Y. Shangguan, D. Le, G. Keren, H. Su, T. Le, C.-F. Yeh, C. Fuegen, and M. L. Seltzer, "Alignment restricted streaming recurrent neural network transducer," in *Proc. of SLT*, 2021, pp. 52–59.

[26] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. of ICASSP*, 2017, pp. 4835–4839.

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

[28] H. Sak, F. de Chaumont Quitry, T. Sainath, K. Rao *et al.*, "Acoustic modelling with CD-CTC-sMBR LSTM RNNs," in *Proc. of ASRU*, 2015, pp. 604–609.

[29] C. Wang, Y. Wu, L. Lu, S. Liu, J. Li, G. Ye, and M. Zhou, "Low latency end-to-end streaming speech recognition with a scout network," in *Proc. of Interspeech*, 2020, pp. 2112–2116.

[30] J. Yu, C.-C. Chiu, B. Li, S.-y. Chang, T. N. Sainath, Y. He, A. Narayanan, W. Han, A. Gulati, Y. Wu *et al.*, "FastEmit: Low-latency streaming asr with sequence-level emission regularization," in *Proc. of ICASSP*, 2021, pp. 6004–6008.

[31] J. Yu, W. Han, A. Gulati, C.-C. Chiu, B. Li, T. N. Sainath, Y. Wu, and R. Pang, "Dual-mode ASR: Unify and improve streaming ASR with full-context modeling," in *Proc. of ICLR*, 2021.

[32] H. Inaguma and T. Kawahara, "StableEmit: Selection probability discount for reducing emission latency of streaming monotonic attention ASR," in *Proc. of Interspeech*, 2021.

[33] H. Miao, G. Cheng, P. Zhang, T. Li, and Y. Yan, "Online hybrid CTC/attention architecture for end-to-end speech recognition," in *Proc. of Interspeech*, 2019, pp. 2623–2627.

[34] ——, "Online hybrid CTC/attention end-to-end automatic speech recognition architecture," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1452–1465, 2020.

[35] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. of ACL*, 2016, pp. 1715–1725.

[36] H. Seki, T. Hori, S. Watanabe, N. Moritz, and J. Le Roux, "Vectorized beam search for CTC-attention-based speech recognition," in *Proc. of Interspeech*, 2019, pp. 3825–3829.

[37] K. Murray and D. Chiang, "Correcting length bias in neural machine translation," in *Proc. of the Third Conference on Machine Translation: Research Papers*, 2018, pp. 212–223.

[38] 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. of Interspeech*, 2020, pp. 5036–5040.

[39] A. Rousseau, P. Deléglise, and Y. Estève, "TED-LIUM: An automatic speech recognition dedicated corpus," in *Proc. of LREC*, 2012, pp. 125–129.

[40] K. Maekawa, "Corpus of Spontaneous Japanese: Its design and evaluation," in *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[41] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *Proc. of ASRU*, 2011.

[42] Y. Zhang, G. Chen, D. Yu, K. Yaco, S. Khudanpur, and J. Glass, "Highway long short-term memory RNNs for distant speech recognition," in *Proc. of ICASSP*, 2016, pp. 5755–5759.

[43] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming Transformer transducer for speech recognition on large-scale dataset," *arXiv preprint arXiv:2010.11395*, 2020.

[44] L. Dong, F. Wang, and B. Xu, "Self-attention aligner: A latency-control end-to-end model for ASR using self-attention network and chunk-hopping," in *Proc. of ICASSP*, 2019, pp. 5656–5660.

[45] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.-C. Chiu, and A. Kannan, "Minimum word error rate training for attention-based sequence-to-sequence models," in *Proc. of ICASSP*, 2018, pp. 4839–4843.

[46] E. Ansari, N. Bach, O. Bojar, R. Cattoni, F. Dalvi, N. Durrani, M. Federico, C. Federmann, J. Gu, F. Huang *et al.*, "Findings of the IWSLT 2020 evaluation campaign," in *Proc. of IWSLT*, 2020, pp. 1–34.