INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Integration of Frame- and Label-synchronous Beam Search for Streaming Encoder–decoder Speech Recognition

Emiru Tsunoo[1], *Hayato Futami*[1], *Yosuke Kashiwagi*[1], *Siddhant Arora*[2], *Shinji Watanabe*[2]

[1]Sony Group Corporation, Japan
[2]Carnegie Mellon University, U.S.A.

emiru.tsunoo@sony.com

Abstract

Although frame-based models, such as CTC and transducers, have an affinity for streaming automatic speech recognition, their decoding uses no future knowledge, which could lead to incorrect pruning. Conversely, label-based attention encoder–decoder mitigates this issue using soft attention to the input, while it tends to overestimate labels biased towards its training domain, unlike CTC. We exploit these complementary attributes and propose to integrate the frame- and label-synchronous (F-/L-Sync) decoding alternately performed within a single beam-search scheme. F-Sync decoding leads the decoding for block-wise processing, while L-Sync decoding provides the prioritized hypotheses using look-ahead future frames within a block. We maintain the hypotheses from both decoding methods to perform effective pruning. Experiments demonstrate that the proposed search algorithm achieves lower error rates compared to the other search methods, while being robust against out-of-domain situations.

Index Terms: speech recognition, beam search, attention-based encoder–decoder, CTC

1. Introduction

Streaming style automatic speech recognition (ASR) is essential for better user experiences. Among end-to-end ASR models, connectionist temporal classification (CTC) [1–4] and transducers [5–7] successfully model the temporal phenomenon of speech in frame-by-frame computation. These models are referred to as frame-synchronous (F-Sync) models and they have an affinity for streaming processing [8–10]. Conversely, the output of ASR is a label-base, which is suitable for modeling with label-synchronous (L-Sync) models, such as attention-based decoders (AttDecs) [11, 12] and language models (LMs) [13, 14]. L-Sync models estimate labels in an autoregressive manner with the given context of the previously estimated output. Generally, L-Sync models have a strong ability to model label sequences, and they are used to improve the ASR performance through fusion or rescoring [15–18].

During decoding, F-Sync models expand the hypotheses at each time frame. F-Sync decoding can be efficiently computed in a beam search by maintaining scores of hypotheses ending with a blank token and non-blank tokens, respectively [1]. However, the hypotheses are pruned at each time frame by using only partial information from the input speech. This limitation sometimes causes unreliable pruning. Conversely, during L-Sync decoding, the hypotheses are extended token-by-token by making soft attention to all the input and the previous output sequence. Thus, this approach has an advantage in pruning over F-Sync decoding without future information. However, it has a label bias problem [19–21], where once the model overestimates a label, it is difficult to downgrade it by the suffix distribution. Furthermore, it is difficult to use the L-Sync models in streaming ASR, where block processing is used [10, 22] and their decoding is generally complicated [23–25].

Both F-Sync and L-Sync decoding are complementary. Several studies have attempted to combine F-Sync and L-Sync decoding. Watanabe *et al.* proposed the joint model, where AttDec and CTC are jointly trained, and CTC rescores the hypotheses generated by AttDec during inference. Sainath *et al.* proposed two-pass decoding [17], in which the AttDec rescores N-best list of the transducer's hypotheses. Li *et al.* combined separate F-Sync and L-Sync models where the former produces a lattice and the latter rescores it [26]. Yan *et al.* introduce AttDec score to F-Sync decoding of CTC in machine translation and speech translation tasks [27]. Additionally, [28, 29] experimentally compared F-Sync and L-Sync decoding. However, in those studies, one is merely used for rescoring the hypotheses generated by the other beforehand, and none simultaneously considers the hypotheses from the individual decoding methods.

This study exploits complementary attributes of the F-Sync and L-Sync decoding, and proposes integrating both in a single beam search scheme. To take advantage of F-Sync decoding, which is easy to incorporate with block-wise streaming ASR, the proposed beam search primarily runs in an F-Sync manner. To mitigate the issue of unreliable partial F-Sync score comparison, we employ the L-Sync beam search based on the shortest token-length prefixes of the hypotheses. The selected hypotheses by L-Sync are then preserved in the subsequent F-Sync pruning steps and adjusted as the hypotheses expand, ensuring that the most promising ones are maintained in the proposed beam search. Experiments demonstrate that the proposed search algorithm performs effectively pruning in the frame–label grid search, and achieve lower error rates compared to the other search methods in English and Japanese datasets. Evaluation using various domains in each language indicates that the proposed method is robust against out-of-domain situations.

2. F-Sync Decoding and L-Sync Decoding for Streaming ASR

2.1. F-Sync decoding

F-Sync models include CTC [1–4] and transducer variants [5–7], which are suitable for streaming ASR. In streaming ASR, blockwise processing is widely used [10, 22]. Let T_b be the last frame of b-th block. An acoustic representation sequence $\mathcal{H}^{T_b} = \{\mathbf{h}_t | 1 \leq t \leq T_b\}$ is extracted by an encoder from speech input. CTC and transducers introduce a blank token, ϕ, to align $\mathcal{H}^{T_b}$ with a different L-length of label sequence

$\mathcal{Y}^L = \{y_l | 1 \leq l \leq L\}$. The F-Sync model estimates ϕ-augmented tokens $z_t \in \mathcal{V} \cup \{\phi\}$ at time frame t, where $\mathcal{V}$ is the vocabulary. The estimated sequence $\mathcal{Z}^{T'} = \{z_t | 1 \leq t \leq T'\}$ can be mapped to a label sequence using mapping function $\mathcal{F} : \mathcal{Z}^{T'} \rightarrow \mathcal{Y}^L$. In the case of CTC, $T' = T_b$, and $T' = T_b + L$ for the transducers, which can emit tokens without consuming frames. Based on $\mathbf{h}_t$ and previous output label y_{l-1}, the F-sync model, $\mathrm{FSM}(\cdot)$, calculates a posterior of z_t.

$$q_{\mathrm{F}}(z_t | \mathbf{h}_t, y_{l-1}) = \mathrm{FSM}(\mathbf{h}_t, y_{l-1}) \tag{1}$$

For CTC, q_{F} does not depend on the previous output. Probability computation of $\mathcal{Y}^l$ at t is efficiently performed as

$$p_{\mathrm{F}}(\mathcal{Y}^l | \mathcal{H}^t) = \gamma(\mathcal{Y}^l_{(\mathrm{b})}, t) + \gamma(\mathcal{Y}^l_{(\mathrm{n})}, t) \tag{2}$$

$$\gamma(\mathcal{Y}^l_{(\mathrm{b})}, t) = \sum_{\mathcal{F}_{\mathrm{F}}(\mathcal{Z}^t) = \mathcal{Y}^l : z_t = \phi} p_{\mathrm{F}}(\mathcal{Y}^l | \mathcal{H}^{t-1}) q(z_t | \mathbf{h}_t, y_l)$$

$$\gamma(\mathcal{Y}^l_{(\mathrm{n})}, t) = \sum_{\mathcal{F}_{\mathrm{F}}(\mathcal{Z}^t) = \mathcal{Y}^l : z_t = y_l} p_{\mathrm{F}}(\mathcal{Y}^{l-1} | \mathcal{H}^{t-1}) q(z_t | \mathbf{h}_t, y_{l-1}),$$

where $\mathcal{Y}_{(\mathrm{b})}$ is the label sequence ending with ϕ, and $\mathcal{Y}_{(\mathrm{n})}$ is the other [1].

F-Sync decoding is performed frame-by-frame. At each time t, probability (2) is used on a logarithmic scale as F-Sync score α_{F}.

$$\alpha_{\mathrm{F}}(\mathcal{Y}^l, t) = \log p_{\mathrm{F}}(\mathcal{Y}^l | \mathcal{H}^t) \tag{3}$$

The hypotheses are copied by blank transition or expanded at each time step with $\mathrm{FSync} : \mathcal{Y}^l \rightarrow \mathcal{Y}^l, \mathcal{Y}^{l+1}$. In the beam search with a fixed beam size B, the $\mathrm{top}B(\cdot)$ function returns a set of the top B hypotheses at step t, denoted as $\Omega_{\mathrm{F},t}$:

$$\Omega_{\mathrm{F},t} = \mathrm{top}B(\alpha_{\mathrm{F}}(\mathcal{Y}, t) | \mathcal{Y} \in \mathrm{FSync}(\mathcal{Y})) \tag{4}$$

Since the F-Sync decoding has no knowledge of future inputs even when the block processing has certain look-ahead frames [10, 25], i.e., the score is conditioned only on $\mathcal{H}^t$ at step t, it sometimes incorrectly prune the correct hypothesis.

2.2. L-Sync models with blockwise streaming processing

We categorize AttDecs [11, 12] and LMs [13, 14] as L-Sync models. At each step i, the L-Sync models predict the probability distribution of the next label y_i using previous output $\mathcal{Y}^{i-1}$ and input $\mathcal{H}^{T_b}$. L-Sync score β_{L} is defined as

$$\beta_{\mathrm{L}}(\mathcal{Y}^i) = \sum_{j=1}^{i} \log p_{\mathrm{L}}(y_j | \mathcal{Y}^{j-1}, \mathcal{H}^{T_b}). \tag{5}$$

For LM, $p_{\mathrm{L}}(y_j)$ does not depend on input $\mathcal{H}^{T_b}$. All the hypotheses are simultaneously extended with $\mathrm{LSync} : \mathcal{Y}^{i-1} \rightarrow \mathcal{Y}^i$. Beam search at step i is performed to maintain top B hypotheses as in F-Sync decoding.

$$\Omega_{\mathrm{L},i} = \mathrm{top}B(\beta_{\mathrm{L}}(\mathcal{Y}^i) | \mathcal{Y}^i \in \mathrm{LSync}(\mathcal{Y}^{i-1})) \tag{6}$$

This contextual dependency enables the L-Sync models to provide a richer representation of label sequences. However, L-Sync models have the following two drawbacks:

- Label bias problem: Once the model overestimates a label biased toward its training domain, the following expansion cannot easily recover it [19–21].
- Endpoint problem: For streaming systems, the AttDec needs to detect an endpoint, which is the point to stop decoding with limited $\mathcal{H}^{T_b}$. This requires additional mechanisms to detect [23–25]. Once the decoding step i passes the endpoint, the AttDec tends to emit unreliable tokens; thus it is better for the AttDec to proceed decoding conservatively in the streaming systems [30].

2.3. F-Sync and L-Sync score fusion

Because F-Sync and L-Sync models are complementary, the aforementioned problems in F-Sync and L-Sync decoding can be alleviated by fusing both scores in pruning.

2.3.1. Score fusion in F-Sync decoding

Typically, LMs are fused in the F-Sync decoding. As in [31, 32], the scores (e.g., Eqs. (3) and (5)) are simply combined with weights λ. In the case of CTC, the combined score is defined as

$$\begin{aligned} s_{\mathrm{F}}(\mathcal{Y}^l, t) =& \lambda_{\mathrm{ctc}} \alpha_{\mathrm{ctc}}(\mathcal{Y}^l, t) + \lambda_{\mathrm{lm}} \beta_{\mathrm{lm}}(\mathcal{Y}^l) \\ &+ \lambda_{\mathrm{att}} \beta_{\mathrm{att}}(\mathcal{Y}^l) + \lambda_{\mathrm{len}} |\mathcal{Y}^l| \end{aligned} \tag{7}$$

The last term in (7) is a label reword, which uses the length of hypothesis $\mathcal{Y}^l$. This mitigates unfair score comparison of different length in the beam, which, however, requires careful tuning.

2.3.2. Score fusion in L-Sync decoding

The joint model [33] has both CTC and AttDec modules, which are trained in a multi-task learning framework. Its decoding is based on L-Sync search led by the AttDec. During the decoding, the score is a combination of the CTC, LM, and AttDec, as follows:

$$\begin{aligned} s_{\mathrm{L}}(\mathcal{Y}^i, i) =& \lambda_{\mathrm{ctc}} \alpha_{\mathrm{ctc}}(\mathcal{Y}^i \ldots, T_b) + \lambda_{\mathrm{lm}} \beta_{\mathrm{lm}}(\mathcal{Y}^i) \\ &+ \lambda_{\mathrm{att}} \beta_{\mathrm{att}}(\mathcal{Y}^i) + \lambda_{\mathrm{len}} |\mathcal{Y}^i| \end{aligned} \tag{8}$$

$$\alpha_{\mathrm{ctc}}(\mathcal{Y}^i \ldots, T_b) = \sum_{y_{i+1} \in \mathcal{V}} \alpha_{\mathrm{ctc}}(\mathcal{Y}^{i+1}, T_b) \tag{9}$$

(9) is a CTC prefix score that accumulates probabilities of all the suffixes of $\mathcal{Y}^i$. The prefix scores are calculated over only the hypotheses generated by the AttDec; thus, it is sometimes difficult to recover incorrect estimation of the AttDec by itself. Note that when $t = T_b$ and $i = l = L$, F-Sync score (3) and the prefix score (9) become equivalent; thus F-Sync decoding and L-Sync decoding eventually evaluate the same score, $s_{\mathrm{F}}(Y^L, T_b) = s_{\mathrm{L}}(Y^L, L)$.

3. Integrated Beam Search of the F-Sync and L-Sync Decoding

As discussed in Sec. 2.1, the F-Sync beam search is likely to incorrectly prune the hypothesis without considering future look-ahead input, which can be prevented by the AttDec that uses all T_b frames. Conversely, the label bias problems in the AttDec (Sec. 2.2) can be mitigated by using the F-Sync models. Therefore, we propose maintaining both the F-Sync and L-Sync hypotheses in extended beam $B' >= B$. Those hypotheses are generated individually from each decoding, and the step increment of t of the F-Sync decoding and i of the L-Sync decoding are performed alternately in an integrated single search algorithm, namely, FL-Sync beam search.

3.1. Prefix score fusion

In the FL-Sync beam search, We choose F-Sync to lead decoding because it is suitable for streaming ASR. Thus, at each step t, the hypotheses are copied and expanded based on F-Sync decoding as in Sec. 2.1. The score for hypothesis $\mathcal{Y}^l$ is extended from (7) and (8) to include both i and t as

$$\begin{aligned} s_{\mathrm{FL}}(\mathcal{Y}^l, i, t) =& \lambda_{\mathrm{ctc}} \alpha_{\mathrm{ctc}}(\mathcal{Y}^l, t) + \lambda_{\mathrm{lm}} \beta_{\mathrm{lm}}(\mathcal{Y}^i) \\ &+ \lambda_{\mathrm{att}} \beta_{\mathrm{att}}(\mathcal{Y}^i) + \lambda_{\mathrm{len}} |\mathcal{Y}^i|. \end{aligned} \tag{10}$$

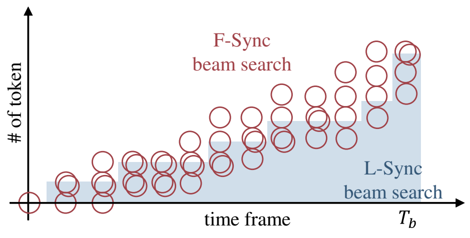

Figure 1: *F-Sync and L-Sync decoding process.*

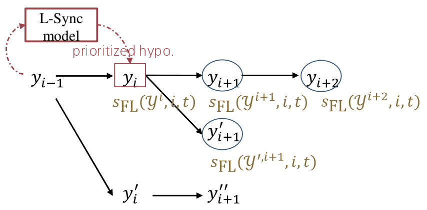
L-Sync model
prioritized hypo.
y_{i-1}
y_i
y_{i+1}
y_{i+2}
$s_{\mathrm{FL}}(\mathcal{Y}^i, i, t)$
$s_{\mathrm{FL}}(\mathcal{Y}^{i+1}, i, t)$
$s_{\mathrm{FL}}(\mathcal{Y}^{i+2}, i, t)$
y'_{i+1}
$s_{\mathrm{FL}}(\mathcal{Y}'^{,i+1}, i, t)$
y'_i
y''_{i+1}

Figure 2: *Prioritized L-Sync hypothesis (red box) and its successors (blue circles). The score of the prioritized hypothesis is compared to all scores of the successors to perform ancestor-pruning.*

While the fist term α_{ctc} defined in (3) is computed for $\mathcal{Y}^l$, β_{lm} and β_{att} are computed only for its prefix $\mathcal{Y}^i$, where $i \le l$ is the current L-Sync decoding step. The reason for using i instead of l is that the token prediction of the AttDec becomes unreliable when it exceeds the endpoint in streaming processing, as discussed in Sec. 2.2. Therefore, we avoid aggressively computing the scores for the entire l labels for the score fusion and conservatively use i instead. Thus, the integrated beam search aims to perform efficient pruning on this 2D grid space of (i, t).

Conversely, the L-Sync decoding expand all the i-length prefix of the hypotheses in hypothesis set $\Omega_{\mathrm{FL},t}$, i.e., $\mathcal{Y}^i \in \mathrm{Pfx}_i(\Omega_{\mathrm{FL},t})$, where $\mathrm{Pfx}_i(\cdot)$ is an $i-$length prefix extraction function. Therefore, to perform L-Sync decoding, the minimum length of the hypotheses needs to be $\min|\mathcal{Y}^l \in \Omega_{\mathrm{FL},t}| \ge i$. The L-Sync decoding advances its step $i \to i+1$ once the minimum length becomes larger than i. A graphical explanation of this is provided in Fig. 1. The red circles indicate that the F-Sync beam search expands the hypotheses in each time step. The L-Sync models add partial scores for the common length to these hypotheses (blue bars). In the last step of F-Sync decoding, T_b, the remaining steps of L-Sync for each hypothesis are consumed, and $\beta_{\mathrm{L}}(\mathcal{Y}^l)$ is applied to each hypothesis to determine the best candidate.

3.2. Prioritized hypotheses from L-Sync decoding

The L-Sync decoding generates hypotheses $\mathcal{Y}^i_{\mathrm{L}}$ based on (8) using all the input, including future look-ahead, which are useful to complement the weakness of F-Sync decoding. However, the hypotheses are pruned based on (10), whose first term, $\alpha_{\mathrm{ctc}}(Y^l, t)$, uses only partial t-frame input unlike (8), and the hypotheses can still be incorrectly pruned. To prevent from dropping L-Sync hypotheses $\mathcal{Y}^i_{\mathrm{L}}$ in the early stage, we prioritize them for survival. At step t, we first preserve those prioritized hypotheses of L-Sync decoding, then F-Sync decoding is performed to fill the remaining hypotheses in the beam B', as

$$\begin{aligned}\Omega_{\mathrm{FL},t} =& \mathrm{top}B(s_{\mathrm{L}}(\mathcal{Y}^i, i) | \mathcal{Y}^i \in \mathrm{LSync}(\mathrm{Pfx}_{i-1}(\Omega_{\mathrm{FL},t-1}))) \\ & \cup \mathrm{top}(B' - B)(s_{\mathrm{FL}}(\mathcal{Y}^l, i, t) | \mathcal{Y}^l \in \mathrm{FSync}(\Omega_{\mathrm{FL},t-1}))\end{aligned} \tag{11}$$

Algorithm 1 FL-Sync beam search algorithm.

```
Input: last frame number of current block $T_b$, acoustic features $\mathcal{H}^{T_b}$,
    beam width for AttDec $B$, total beam width $B'$
Output: $\Omega_{\mathrm{FL},T_b}$: hypotheses for current block $b$
 1: Initialize: $y_0 \leftarrow \langle \mathrm{sos} \rangle$, $\Omega_{\mathrm{FL},0} \leftarrow \{y_0\}$, $t \leftarrow 1$, $i \leftarrow 1$
 2: while $t < T_b$ do
 3:     $\hat{\Omega} \leftarrow \{\}$
 4:     if $i < \min(|Y \in \Omega_{\mathrm{FL},t-1}|)$ then
 5:         $i \leftarrow \min(|Y \in \Omega_{\mathrm{FL},t-1}|)$
 6:         $\hat{\Omega} \leftarrow \mathrm{LSync}(\mathrm{Pfx}_{i-1}(\Omega_{\mathrm{FL},t-1}))$    ▷ Prefix L-Sync search
 7:         $\hat{\Omega} \leftarrow \mathrm{top}B(s_{\mathrm{L}}(\mathcal{Y}^i, i) | \mathcal{Y}^i \in \hat{\Omega})$    ▷ Keep top-$B$ hyps
 8:     end if
 9:     $\hat{\Omega} \leftarrow \hat{\Omega} \cup \mathrm{FSync}(\Omega_{\mathrm{FL},t-1})$    ▷ FSync search
10:     $\hat{\Omega} \leftarrow \mathrm{AncestorPruning}(\hat{\Omega})$    ▷ Sec. 3.3
11:     $\Omega_{\mathrm{FL},t} \leftarrow \mathrm{OnlyPriority}(\hat{\Omega})$    ▷ Sec. 3.1
12:     $\Omega_{\mathrm{FL},t} \leftarrow \Omega_{\mathrm{FL},t} \cup \mathrm{top}(B' - |\Omega_{\mathrm{FL},t}|)(s_{\mathrm{FL}}(\mathcal{Y}, i, t) | \mathcal{Y} \in \hat{\Omega})$
13:     $t \leftarrow t + 1$
14: end while
15: return $\Omega_{\mathrm{FL},T_b}$
```

3.3. Ancestor-pruning for the prioritized hypothesis

The priority of the hypothesis $\mathcal{Y}^i_{\mathrm{L}}$ is no longer valid once the L-Sync decoding increase its step $i \to i+1$, as it generate new set of hypotheses $\mathcal{Y}^{i+1}_{\mathrm{L}}$. Further, to prune unnecessary hypotheses of L-Sync decoding, we propose to perform ancestor-pruning, which is similar to the depth-pruning in [32]. As the F-Sync decoding proceed, it expands the hypothesis $\mathcal{Y}^i \to \mathcal{Y}^{i+1}$, as shown in Fig. 2. The prioritized root hypothesis $\mathcal{Y}^i$ is indicated by the red box and the expanded successors $\mathcal{Y}^{i+*}$ are indicated by the blue circles. We compare the scores of all the successors and the root hypothesis, denoted as s_{FL} in Fig. 2. When all the scores of successors become greater than the score of the root, $s_{\mathrm{FL}}(\mathcal{Y}^i_{\mathrm{L}}, i, t) < \min(s_{\mathrm{FL}}(\mathcal{Y}^{i+*}, i, t))$, we prune away prioritized root hypothesis $\mathcal{Y}^i_{\mathrm{L}}$, because any new successor from the root $\mathcal{Y}^i_{\mathrm{L}}$ can no longer achieve higher score than the current successors due to the $0 \le q(z_t|\mathbf{h}_t, y_i) \le 1$ restriction.

3.4. FL-Sync beam search algorithm

The proposed beam search is summarized in Algorithm 1. As the FL-Sync decoding is performed based on the F-Sync frame-wise pruning, t is increased at each step in the loop. First, L-Sync decoding is performed if step $i < \min(|Y \in \Omega_{\mathrm{FL},t-1}|)$ as descried in Sec. 3.1. The L-Sync decoding expands the prefixes of the hypotheses (Line 6) and provides prioritized hypotheses $\mathcal{Y}^i_{\mathrm{L}}$, which are maintained in temporary hypothesis set $\hat{\Omega}$ (Line 7). Then the F-Sync decoding is performed based on (10), and merged with the prioritized hypotheses (Line 9). Subsequently, in Line 10, the ancestor-pruning described in Sec. 3.3 is performed, which removes unnecessary hypotheses. The hypotheses from L-Sync decoding have priority for survival (Line 11), then lastly the hypothesis set is filled from the temporary set up to total beam size, B' (Line 12). Note that, even the total beam size increases from B to B', computational impact is marginal from B-beam L-Sync decoding, because the F-Sync model, e.g., CTC, generally requires much less computation.

4. Experiments

To evaluate the effectiveness of the proposed FL-Sync beam search, we evaluated cross-domain and intra-domain scenarios in English and Japanese datasets.

Table 1: *Comparison of searching algorithms of streaming ASR in English datasets in WER. All the models were trained with Librispeech. ID refers to in-domain. In all the decoding methods, the LM of the target domain is fused.*

	total	L-Sync	LS→TEDLIUM3		LS→Switchboard		LS→STOP	Librispeech (ID)	
	beam size	beam size	dev	test	SW	CH		test-clean	test-other
Streaming L-Sync decoding [25]		5	13.7	13.1	30.2	35.7	14.5	3.0	8.1
CTC F-Sync decoding [32]	10		15.0	14.4	31.0	37.2	19.2	3.3	9.0
CTC+AttDec F-Sync decoding	10		14.6	14.0	31.4	36.6	19.4	3.1	8.6
Integ. FL-Sync decoding (proposed)	10	5	**13.4**	**12.7**	**29.9**	**35.0**	**14.1**	**2.9**	**7.9**

Table 2: *Comparison of searching algorithms of streaming ASR in Japanese datasets in CER. All the models were trained with CSJ+LaboroTV (LTV). ID refers to in-domain. In all the decoding methods, the general LM is fused.*

	total	L-Sync	CSJ+LTV	CSJ+LTV	CSJ+LTV	CSJ+LTV	CSJ (ID)		
	beam size	beam size	→TEDxJP	→News	→CMD1	→CMD2	eval1	eval2	eval3
Streaming L-Sync decoding [25]		5	12.1	3.5	4.1	12.8	8.0	7.5	**7.0**
CTC F-Sync decoding [32]	10		12.4	4.9	5.1	10.1	8.7	7.8	7.3
CTC+AttDec F-Sync decoding	10		12.1	3.5	3.5	10.7	8.0	7.4	7.4
Integ. FL-Sync decoding (proposed)	10	5	**11.9**	**3.0**	**3.2**	**9.6**	**7.9**	**7.3**	7.1

4.1. Experimental setup

For the English evaluation, we trained a streaming encoder–decoder ASR model following [25] with the Librispeech dataset [34], a read speech corpus. It was applied to three various domains: The TED-LIUM 3 [35] dev/test set (a spontaneous lecture style), Hub5'00 (telephony-style conversation) having Switchboard (SWB) and CallHome (CHM) subsets, and voice-command-style STOP dataset [36].

For Japanese, we trained a streaming ASR model with a merged set of the lecture style CSJ [37] and LaboroTV corpus of TV program [38]. We used three evaluation sets of the CSJ dataset for intra-domain setup, and TEDxJP [38] for cross-domain setup. To cover various range of domains, We also used in-house evaluation data; 720 read news utterances (News) spoken by four males and four females, 2,620 voice commands (CMD1) and far-field 1,768 commands (CMD2), each spoken by 10 hired speakers.

The input acoustic features were 80-dimensional filter bank features. The input features were applied mean normalization with a sliding window. The encoder consisted of 12 blocks of conformer [6]. The AttDec had six decoder blocks. Each block consisted of four-head 256-unit attention layers and 2048-unit feed-forward layers. Contextual block encoding [22] was applied to the encoder with a block size of 40, a shift size of 16, and a look-ahead size of 16. The models were trained using multitask learning with CTC loss [33], with a weight of 0.3.

For English evaluation, external LMs were trained using each target dataset for adaptation. LMs were four-layer unidirectional LSTM with 2048 units, using the byte-pair encoding (BPE) subword tokenization with 5000 token classes. For Japanese, we collected over 27 million sentences to train a general LM with 10,000 BPE tokens, applied to all the tasks.

We compared our proposed FL-Sync decoding with streaming L-Sync decoding [25], and CTC F-Sync decoding [32]. We also evaluated AttDec score fusion for CTC F-Sync decoding for comparison. For the CTC F-Sync decoding methods, $\lambda_{\text{lm}} = 0.4$ was used for English datasets, and $\lambda_{\text{lm}} = 0.1$ for Japanese, instead. For the baseline L-Sync decoding and the proposed FL-Sync decoding, we adopted $(\lambda_{\text{lm}}, \lambda_{\text{ctc}}) = (0.4, 0.4)$ for English intra-domain, $(0.6, 0.4)$ for English cross-domain, and $(0.3, 0.5)$ for Japanese experiments. λ_{att} was set as $(1 - \lambda_{\text{ctc}})$. We adopted $\lambda_{\text{len}} = 1$ for all the evaluation. We used $B' = 10$ and $B = 5$ as a beam size for total FL-Sync decoding and L-Sync decoding in it, respectively. For fair comparison, we used the same beam sizes for the baseline approaches.

4.2. Results of English evaluation

The word error rate (WER) results are summarized in Table 1. Streaming L-Sync decoding was generally better than CTC F-Sync decoding, particularly in the voice-command STOP dataset. This indicated that it is more effective in pruning to use input information including future look-ahead frames. Although the AttDec score fusion improved accuracy on CTC F-Sync decoding, the proposed FL-Sync decoding showed large improvement from those F-Sync decoding methods because it also preserved hypotheses from the L-Sync decoding. When we compare FL-Sync decoding with the L-Sync baseline decoding, our proposed method performed robustly in the cross-domain scenarios; for instance, WER was improved from 13.1% to 12.7% in TEDLIUM3 test set. Furthermore, we confirmed that it did not degrade the intra-domain performance, or even observed slight improvement from 8.1% to 7.9% in test-other, for instance. Since the scores, (7), (8), and (10), eventually become the same in all the search methods, the results show that our method is effective in pruning because of maintaining hypotheses from both L-Sync and F-Sync decoding.

4.3. Results of Japanese evaluation

We evaluated character error rates (CERs), which are listed in Table 2. The results followed similar tendency to the English evaluation. As we used the general LM, it did not exactly match to some of the target domains. As a result, we observed that even in the source domain CSJ eval sets, CERs were relatively higher than the other literature [25, 39]. Furthermore, the baseline L-Sync decoding had difficulty in adapting to the target domains, in particular in CMD2 data, which tend not to be grammatically structured. Since our proposed FL-Sync decoding maintained both L-Sync and F-Sync decoding, it recovered the domain bias and was robust to the cross-domain situation.

5. Conclusion

We have proposed a new FL-Sync beam search, which integrates complementary F-Sync and Sync decoding to mitigate the problems in both the decoding. The proposed beam search primarily runs in an F-Sync manner to incorporate it with blockwise streaming ASR. To overcome the unreliable partial F-Sync score comparison in pruning, L-Sync decoding provide the prioritized hypotheses with future look-ahead input frames, which are also maintained in the beam to perform effective pruning.

6. References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006, pp. 369–376.

[2] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, 2014, pp. 1764–1772.

[3] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. of ASRU Workshop*, 2015, pp. 167–174.

[4] D. Amodei *et al.*, "Deep Speech 2: End-to-end speech recognition in English and Mandarin," in *Proc. ICML*, vol. 48, 2016, pp. 173–182.

[5] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

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

[7] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. ICASSP*, 2020, pp. 7829–7833.

[8] 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. ICASSP*, 2019, pp. 5656–5660.

[9] 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. ICASSP*, 2021, pp. 6004–6008.

[10] Y. Shi, Y. Wang, C. Wu, C.-F. Yeh, J. Chan, F. Zhang, D. Le, and M. Seltzer, "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *Proc. ICASSP*, 2021, pp. 6783–6787.

[11] 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.

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

[13] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. Interspeech*, 2010, pp. 1045–1048.

[14] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, "Language modeling with deep transformers," in *Proc. Interspeech*, 2019, pp. 3905–3909.

[15] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *Proc. Interspeech*, 2017, pp. 523–527.

[16] A. Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *Proc. ICASSP*, 2018, pp. 1–5828.

[17] T. N. Sainath, R. Pang, D. Rybach, Y. He, R. Prabhavalkar, W. Li, M. Visontai, Q. Liang, T. Strohman, Y. Wu, *et al.*, "Two-pass end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 2773–2777.

[18] W. Zhou, S. Berger, R. Schlüter, and H. Ney, "Phoneme based neural transducer for large vocabulary speech recognition," in *Proc. ICASSP*, 2021, pp. 5644–5648.

[19] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, 2001, pp. 282–289.

[20] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Proc. NeurIPS*, vol. 28, 2015.

[21] K. Murray and D. Chiang, "Correcting length bias in neural machine translation," *arXiv preprint arXiv:1808.10006*, 2018.

[22] E. Tsunoo, Y. Kashiwagi, T. Kumakura, and S. Watanabe, "Transformer ASR with contextual block processing," in *Proc. of ASRU Workshop*, 2019, pp. 427–433.

[23] N. Moritz, T. Hori, and J. Le Roux, "Triggered attention for end-to-end speech recognition," in *Proc. ICASSP*, 2019, pp. 5666–5670.

[24] M. Li, C. Zorilă, and R. Doddipatla, "Head-synchronous decoding for transformer-based streaming ASR," in *Proc. ICASSP*, 2021, pp. 5909–5913.

[25] E. Tsunoo, C. Narisetty, M. Hentschel, Y. Kashiwagi, and S. Watanabe, "Run-and-back stitch search: Novel block synchronous decoding for streaming encoder-decoder ASR," in *Proc. ICASSP*, 2022, pp. 8287–8291.

[26] Q. Li, C. Zhang, and P. C. Woodland, "Combining frame-synchronous and label-synchronous systems for speech recognition," *arXiv preprint arXiv:2107.00764*, 2021.

[27] B. Yan, S. Dalmia, Y. Higuchi, G. Neubig, F. Metze, A. W. Black, and S. Watanabe, "CTC alignments improve autoregressive translation," *arXiv preprint arXiv:2210.05200*, 2022.

[28] 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.

[29] W. Zhou, A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "Equivalence of segmental and neural transducer modeling: A proof of concept," in *Proc. Interspeech*, 2021, pp. 2891–2895.

[30] E. Tsunoo, Y. Kashiwagi, and S. Watanabe, "Streaming transformer ASR with blockwise synchronous beam search," in *Proc. SLT*, 2021, pp. 22–29.

[31] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[32] K. Hwang and W. Sung, "Character-level incremental speech recognition with recurrent neural networks," in *Proc. ICASSP*, 2016, pp. 5335–5339.

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

[34] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[35] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Esteve, "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *International conference on speech and computer*, 2018, pp. 198–208.

[36] P. Tomasello, A. Shrivastava, D. Lazar, P.-C. Hsu, D. Le, A. Sagar, A. Elkahky, J. Copet, W.-N. Hsu, Y. Adi, *et al.*, "STOP: A dataset for spoken task oriented semantic parsing," in *Proc. SLT*, 2023, pp. 991–998.

[37] K. Maekawa, H. Koiso, S. Furui, and H. Isahara, "Spontaneous speech corpus of Japanese," in *Proc. of the International Conference on Language Resources and Evaluation (LREC)*, 2000, pp. 947–9520.

[38] S. Ando and H. Fujihara, "Construction of a large-scale Japanese ASR corpus on TV recordings," in *Proc. ICASSP*, 2021, pp. 6948–6952.

[39] S. Karita, Y. Kubo, M. A. U. Bacchiani, and L. Jones, "A comparative study on neural architectures and training methods for Japanese speech recognition," in *Proc. Interspeech*, 2021, pp. 2092–2096.