FASTEMIT: LOW-LATENCY STREAMING ASR WITH SEQUENCE-LEVEL EMISSION REGULARIZATION

Jiahui Yu   Chung-Cheng Chiu   Bo Li   Shuo-yiin Chang   Tara N. Sainath   Yanzhang He
Arun Narayanan   Wei Han   Anmol Gulati   Yonghui Wu   Ruoming Pang

Google LLC, USA

{jiahuiyu,rpang}@google.com

ABSTRACT

Streaming automatic speech recognition (ASR) aims to emit each hypothesized word as quickly and accurately as possible. However, emitting fast without degrading quality, as measured by word error rate (WER), is highly challenging. Existing approaches including Early and Late Penalties [1] and Constrained Alignments [2, 3] penalize emission delay by manipulating *per-token* or *per-frame* probability prediction in sequence transducer models [4]. While being successful in reducing delay, these approaches suffer from significant accuracy regression and also require additional word alignment information from an existing model. In this work, we propose a sequence-level emission regularization method, named *FastEmit*, that applies latency regularization directly on *per-sequence* probability in training transducer models, and does not require any alignment. We demonstrate that *FastEmit* is more suitable to the sequence-level optimization of transducer models [4] for streaming ASR by applying it on various end-to-end streaming ASR networks including RNN-Transducer [5], Transformer-Transducer [6, 7], ConvNet-Transducer [8] and Conformer-Transducer [9]. We achieve $150 \sim 300\text{ms}$ latency reduction with significantly better accuracy over previous techniques on a Voice Search test set. *FastEmit* also improves streaming ASR accuracy from $4.4\%/8.9\%$ to $\mathbf{3.1\%}/\mathbf{7.5\%}$ WER, meanwhile reduces 90th percentile latency from 210ms to only $\mathbf{30ms}$ on LibriSpeech.

1. INTRODUCTION

End-to-end (E2E) recurrent neural network transducer (RNN-T) [4] models have gained enormous popularity for streaming ASR applications, as they are naturally streamable [1, 5, 6, 7, 10, 11, 12, 13]. However, naive training with a sequence transduction objective [4] to maximize the log-probability of target sequence is **unregularized** and these streaming models learn to predict better by using more context, causing significant emission delay (*i.e.*, the delay between the user speaking and the text appearing). Recently there are some approaches trying to regularize or penalize the emission delay. For example, Li *et al*. [1] proposed Early and Late Penalties to enforce the prediction of </s> (end of sentence) within a reasonable time window given by a voice activity detector (VAD). Constrained Alignments [2, 3] were also proposed by extending the penalty terms to each word, based on speech-text alignment information [14] generated from an existing speech model.

While being successful in terms of reducing latency of streaming RNN-T models, these two regularization approaches suffer from accuracy regression [1, 3]. One important reason is because both regularization techniques penalize the *per-token* or *per-frame* prediction probability independently, which is inconsistent with the sequence-level transducer optimization of *per-sequence* probability

calculated by the transducer forward-backward algorithm [4]. Although some remedies like second-pass Listen, Attend and Spell (LAS) [15] rescorer [16, 17] and minimum word error rate (MWER) training technique [18] have been used to reduce the accuracy regression, these approaches come at a non-negligible compute cost in both training and serving.

In this work, we propose a novel sequence-level emission regularization method for streaming models based on transducers, which we call *FastEmit*. *FastEmit* is designed to be directly applied on the transducer forward-backward per-sequence probability, rather than individual per-token or per-frame prediction of probability independently. In breif, in RNN-T [4] it first extends the output vocabulary space $\mathcal{Y}$ with a 'blank token' $\varnothing$, meaning 'output nothing'. Then the transducer forward-backward algorithm calculates the probability of each lattice (speech-text alignment) in the $T \times U$ matrix, where T and U is the length of input and output sequence respectively. Finally the optimal lattice in this matrix can be automatically learned by maximizing log-probability of the target sequence. It is noteworthy that in this transducer optimization, emitting a vocabulary token $y \in \mathcal{Y}$ and the blank token $\varnothing$ are *treated equally*, as long as the log-probability of the target sequence can be maximized. However, in streaming ASR systems the blank token $\varnothing$ 'output nothing' should be discouraged as it leads to higher emission latency. We will show in detail that *FastEmit*, as a sequence-level regularization method, encourages emitting vocabulary tokens $y \in \mathcal{Y}$ and suppresses blank tokens $\varnothing$ across the entire sequence based on transducer forward-backward probabilities, leading to significantly lower emission latency while retaining recognition accuracy.

FastEmit has many advantages over other regularization methods to reduce emission latency in end-to-end streaming ASR models: (1) *FastEmit* is a sequence-level regularization based on transducer forward-backward probabilities, thus is more suitable when applied jointly with the sequence-level transducer objective. (2) *FastEmit* does not require any speech-word alignment information [3] either by labeling or generated from an existing speech model. Thus it is easy to 'plug and play' in any transducer model on any dataset without any extra effort. (3) *FastEmit* has minimal hyper-parameters to tune. It only introduces one hyper-parameter λ to balance the transducer loss and regularization loss. (4) There is no additional training or serving cost to apply *FastEmit*.

We apply *FastEmit* on various end-to-end streaming ASR networks including RNN-Transducer [5], Transformer-Transducer [6, 7], ConvNet-Transducer [8] and Conformer-Transducer [9]. We achieve $\mathbf{150 \sim 300ms}$ latency reduction with significantly better accuracy over previous methods [2, 3, 10] on a Voice Search test set. *FastEmit* also improves streaming ASR accuracy from $4.4\%/8.9\%$ to $\mathbf{3.1\%}/\mathbf{7.5\%}$ WER, meanwhile reduces 90th percentile latency from 210ms to only $\mathbf{30ms}$ on LibriSpeech.
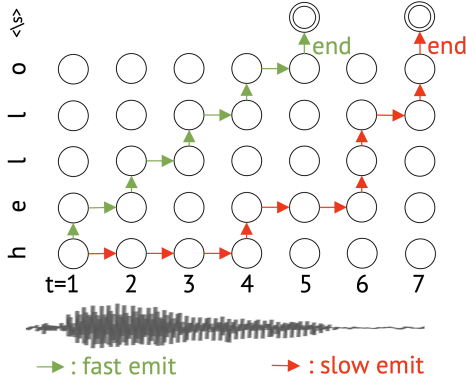
Fig. 1. Examples of fast and slow transducer emission lattices (speech-text alignments). Transducer aims to maximize the log-probability of any lattice, regardless of its emission latency.
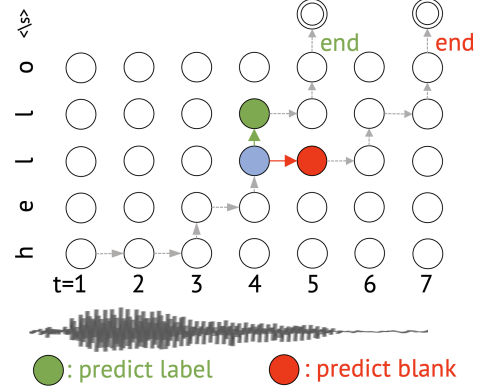

Fig. 2. Illustration of *FastEmit* regularization. Consider any node (*e.g.*, **blue** node), *FastEmit* encourages predicting label $y \in \mathcal{Y}$ (**green** node) instead of predicting blank $\varnothing$ (**red** node).

2. TRANSDUCER WITH FASTEMIT

In this section, we first delve into transducer [4] and show why naively optimizing the transducer objective is **unregularized** thus unsuitable for low-latency streaming ASR models. We then propose *FastEmit* as a sequence-level emission regularization method to regularize the emission latency.

2.1. Transducer

Transducer optimization [4] automatically learns probabilistic alignments between an *input sequence* $\boldsymbol{x} = (x_1, x_2, \ldots, x_T)$ and an *output sequence* $\boldsymbol{y} = (y_1, y_2, \ldots, y_U)$, where T and U denote the length of input and output sequences respectively. To learn the probabilistic alignments, it first extends the output space $\mathcal{Y}$ with a 'blank token' $\varnothing$ (meaning 'output nothing', visually denoted as right arrows in Figure 1 and 2): $\bar{\mathcal{Y}} = \mathcal{Y} \cup \varnothing$. The allocation of these blank tokens then determines an alignment between the input and output sequences. Given an input sequence $\boldsymbol{x}$, the transducer aims to maximize the log-probability of a conditional distribution:

$$\mathcal{L} = -\log P(\hat{\boldsymbol{y}}|\boldsymbol{x}) = -\log \sum_{\boldsymbol{a} \in \mathcal{B}^{-1}(\hat{\boldsymbol{y}})} P(\boldsymbol{a}|\boldsymbol{x}) \qquad (1)$$

where $\mathcal{B} : \bar{\mathcal{Y}} \to \mathcal{Y}$ is a function that removes the $\varnothing$ tokens from each alignment lattice $\boldsymbol{a}$, and $\hat{\boldsymbol{y}}$ is the ground truth output sequence tokenized from text label.

As shown in Figure 1, we denote each *node* (t, u) as the probability of emitting the first u elements of the output sequence by the first t frames of the input sequence. We further denote the prediction from a neural network $\hat{y}(t, u)$ and $b(t, u)$ as the probability of label token (up arrows in figures) and blank token (right arrows in figures) at *node* (t, u). To optimize the transducer objective, an efficient forward-backward algorithm [4] is used to calculate the probability of each alignment and aggregate all possible alignments before propagating gradients back to $\hat{y}(t, u)$ and $b(t, u)$. It is achieved by defining *forward variable* $\alpha(t, u)$ as the probability of emitting $\hat{y}[1{:}u]$ during $x[1{:}t]$, and *backward variable* $\beta(t, u)$ as the probability of emitting $\hat{y}[u+1{:}U]$ during $x[t{:}T]$, using an efficient forward-

backward propagation algorithm:

$$\alpha(t, u) = \hat{y}(t, u{-}1)\alpha(t, u{-}1) + b(t{-}1, u)\alpha(t{-}1, u), \quad (2)$$
$$\beta(t, u) = \hat{y}(t, u)\beta(t, u{+}1) + b(t, u)\beta(t{+}1, u), \qquad (3)$$

where the initial conditions are $\alpha(1, 0) = 1$, $\beta(T, U) = b(T, U)$. It is noteworthy that $\alpha(t, u)\beta(t, u)$ defines the probability of all complete alignments in $\mathcal{A}_{t,u}$: {complete alignment through node(t, u)}:

$$P(\mathcal{A}_{t,u}|\boldsymbol{x}) = \sum_{\boldsymbol{a} \in \mathcal{A}_{t,u}} P(\boldsymbol{a}|\boldsymbol{x}) = \alpha(t, u)\beta(t, u). \qquad (4)$$

By diffusion analysis of the probability of all alignments, we know that $P(\hat{\boldsymbol{y}}|\boldsymbol{x})$ is equal to the sum of $P(\mathcal{A}_{t,u}|\boldsymbol{x})$ over any top-left to bottom-right diagonal nodes (*i.e.*, all complete alignments will pass through any diagonal cut in the $T \times U$ matrix in Figure 1) [4]:

$$P(\hat{\boldsymbol{y}}|\boldsymbol{x}) = \sum_{(t,u):t+u=n} P(\mathcal{A}_{t,u}|\boldsymbol{x}), \forall n : 1 \le n \le U + T. \quad (5)$$

Finally, gradients of transducer loss function $\mathcal{L} = -\log P(\hat{\boldsymbol{y}}|\boldsymbol{x})$ *w.r.t.* neural network prediction of probability $\hat{y}(t, u)$ and $b(t, u)$ can be calculated according to Equations 1, 2, 3, 4 and 5.

2.2. *FastEmit*

Now let us consider any node in the $T \times U$ matrix, for example, the blue node at (t, u), as shown in Figure 2. First we know that the probability of emitting $\hat{y}[1{:}u]$ during $x[1{:}t]$ is $\alpha(t, u)$. At the next step, the alignment can either 'go up' by predicting label $u{+}1$ to the green node with probability $\hat{y}(t, u)$, or 'turn right' by predicting blank $\varnothing$ to the red node with probability $b(t, u)$. Finally together with backward probability β of the new node, the probability of all complete alignments $\mathcal{A}_{t,u}$ passing through node (t, u) in Equation 4 can be **decomposed** into two parts:

$$P(\mathcal{A}_{t,u}|\boldsymbol{x}) = \alpha(t, u)\beta(t, u) = \qquad (6)$$
$$\underbrace{\alpha(t, u)b(t, u)\beta(t{+}1, u)}_{\text{predict blank}} + \underbrace{\alpha(t, u)\hat{y}(t, u)\beta(t, u{+}1)}_{\text{predict label}},$$

which is equivalent as replacing $\beta(t, u)$ in Equation 4 with Equation 3. From Equation 6 we know that gradients of transducer loss

$\mathcal{L}$ *w.r.t.* the probability prediction of any node (t, u) have following properties (closed-form gradients can be found in [4] Equation 20):

$$\frac{\partial \mathcal{L}}{\partial \hat{y}(t, u)} \propto \alpha(t, u)\beta(t, u+1) \tag{7}$$

$$\frac{\partial \mathcal{L}}{\partial b(t, u)} \propto \alpha(t, u)\beta(t+1, u). \tag{8}$$

However, this transducer loss $\mathcal{L}$ aims to maximize log-probability of all possible alignments, regardless of their emission latency. In other words, as shown in Figure 2, emitting a vocabulary token $y \in \mathcal{Y}$ and the blank token $\varnothing$ are *treated equally*, as long as the log-probability is maximized. It inevitably leads to emission delay because streaming ASR models learn to predict better by using more future context, causing significant emission delay.

By the decomposition in Equation 6, we propose a simple and effective transducer regularization method, *FastEmit*, which encourages predicting label instead of blank by additionally maximizing the probability of 'predict label' based on Equation 1, 5 and 6:

$$\tilde{P}(\mathcal{A}_{t,u}|\boldsymbol{x}) = \underbrace{\alpha(t, u)\hat{y}(t, u)\beta(t, u+1)}_{\text{predict label}}, \tag{9}$$

$$\tilde{\mathcal{L}} = -\log \sum_{(t,u):t+u=n} (P(\mathcal{A}_{t,u}|\boldsymbol{x}) + \boldsymbol{\lambda}\tilde{P}(\mathcal{A}_{t,u}|\boldsymbol{x})), \tag{10}$$

$\forall n : 1 \leq n \leq U + T$. $\tilde{\mathcal{L}}$ is the new transducer loss with *FastEmit* regularization and $\boldsymbol{\lambda}$ is a hyper-parameter to balance the transducer loss and regularization loss. *FastEmit* is easy to implement based on an existing transducer implementation, because the gradients calculation of this new regularized transducer loss $\tilde{\mathcal{L}}$ follows:

$$\frac{\partial \tilde{\mathcal{L}}}{\partial \hat{y}(t, u)} = (1+\boldsymbol{\lambda})\frac{\partial \mathcal{L}}{\partial \hat{y}(t, u)}, \tag{11}$$

$$\frac{\partial \tilde{\mathcal{L}}}{\partial b(t, u)} = \frac{\partial \mathcal{L}}{\partial b(t, u)}, \tag{12}$$

To interpret the gradients of *FastEmit*, intuitively it simply means that the gradients of emitting label tokens has a 'higher learning rate' back-propagating into the streaming ASR network, while emitting blank token remains the same. We also note that the proposed *FastEmit* regularization method is based on alignment probabilities instead of per-token or per-frame prediction of probability, thus we refer it as *sequence-level emission regularization*.

3. EXPERIMENTAL DETAILS

3.1. Latency Metrics

Our latency metrics of streaming ASR are motivated by real-world applications like Voice Search and Smart Home Assistants. In this work we mainly measure two types of latency metrics described below: (1) partial recognition latency on both LibriSpeech and MultiDomain datasets, and (2) endpointer latency [19] on MultiDomain dataset. A visual example of two latency metrics is illustrated in Figure 3. For both metrics, we report both 50-th (medium) and 90-th percentile values of all utterances in the test set to better characterize latency by excluding outlier utterances.

Partial Recognition (PR) Latency is defined as the timestamps difference of two events as illustrated in Figure 3: (1) when the last token is emitted in the finalized recognition result, (2) the end of the speech when a user finishes speaking estimated by forced alignment. PR latency is especially descriptive of user experience in real-world
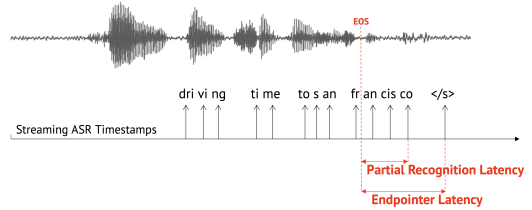

Fig. 3. A visual illustration of PR latency and EP latency metrics.

streaming ASR applications like Voice Search and Assistants. Moreover, PR latency is the lower bound for applying other techniques like Prefetching [11], by which streaming application can send early server requests based on partial/incomplete recognition hypotheses to retrieve relevant information and necessary resources for future actions. Finally, unlike other latency metrics that may depend on hardware, environment or system optimization, PR latency is inherented to streaming ASR models and thus can better characterize the emission latency of streaming ASR. It is also noteworthy that models that capture stronger contexts can emit a hypothesis even before they are spoken, leading to a **negative PR latency**.

Endpointer (EP) Latency is different from PR latency and it measures the timestamps difference between: (1) when the streaming ASR system predicts the end of the query (EOQ), (2) the end of the speech when a user finishes speaking estimated by forced alignment. As illustrated in Figure 3, EOQ can be implied by jointly predicting the $</s>$ token with end-to-end Endpointing introduced in [19]. The endpointer can be used to close the microphone as soon as the user finishes speaking, but it is also important to avoid cutting off users while they are still speaking. Thus, the prediction of the $</s>$ token has a higher latency compared with PR latency, as shown in Figure 3. Note that PR latency is also a lower bound of EP latency, thus reducing the PR latency is the main focus of this work.

3.2. Dataset and Training Details

We report our results on two datasets, a public dataset LibriSpeech [20] and an internal large-scale dataset MultiDomain [21].

Our main results and ablation studies will be presented on a widely used public dataset LibriSpeech [20], which consists of about 1000 hours of English reading speech. For data processing, we extract 80-channel filterbanks feature computed from a 25ms window with a stride of 10ms, use SpecAugment [22] for data augmentation, and train with the Adam optimizer. We use a single layer LSTM as the decoder. All of these training settings follow the previous work [8, 9] for fair comparison. We train our LibriSpeech models on 960 hours of LibriSpeech training set with labels tokenized using a 1,024 word-piece model (WPM), and report our test results on LibriSpeech TestClean and TestOther (noisy).

We also report our results a production dataset MultiDomain [21], which consists of 413,000 hours speech, 287 million utterances across multiple domains including Voice Search, YouTube, and Meetings. Multistyle training (MTR) [23] is used for noise robustness. These training and testing utterances are anonymized and hand-transcribed, and are representatives of Google's speech recognition traffic. All models are trained to predict labels tokenized using a 4,096 word-piece model (WPM). We report our results on a test set of 14K Voice Search utterances with duration less than 5.5 seconds long.

3.3. Model Architectures

FastEmit can be applied to any transducer model on any dataset without any extra effort. To demonstrate the effectiveness of our proposed method, we apply *FastEmit* on a wide range of transducer models including RNN-Transducer [5], Transformer-Transducer [6], ConvNet-Transducer [8] and Conformer-Transducer [9]. We refer the reader to the individual papers for more details of each model architecture. For each of our experiment, we keep the exact same training and testing settings including model size, model regularization (weight decay, variational noise, *etc.*), optimizer, learning rate schedule, input noise and augmentation, *etc*. All models are implemented, trained and benchmarked based on Lingvo toolkit [24].

All these model architectures are based on encoder-decoder transducers. The encoders are based on autoregressive models using uni-directional LSTMs, causal convolution and/or left-context attention layers (no future context is permitted). The decoders are based on prediction network and joint network similar to previous RNN-T models [1, 4, 10]. For all experiments on LibriSpeech, we report results directly after training with the transducer objective. For all our experiments on MultiDomain, results are reported with minimum word error rate (MWER) finetuning [18] for fair comparison.

4. RESULTS

In this section, we first report our results on LibriSpeech dataset and compare with other streaming ASR networks. We next study the hyper-parameter λ in *FastEmit* to balance transducer loss and regularization loss. Finally, we conduct large-scale experiments on the MultiDomain production dataset and compare *FastEmit* with other methods [1, 2, 3] on a Voice Search test set.

4.1. Main Results on LibriSpeech

Table 1. Streaming ASR results on LibriSpeech dataset. We apply *FastEmit* to Large and Medium size streaming ContextNet [8] and Conformer [9].

Method	WER TestClean	WER TestOther	PR50 (ms)	PR90 (ms)
LSTM	4.7	11.1	80	180
Transformer	4.5	10.9	70	190
Conformer-M	4.6	9.9	140	280
+FastEmit	3.7 (-0.9)	9.5 (-0.4)	-40 (-180)	80 (-200)
Conformer-L	4.5	9.5	110	230
+FastEmit	3.5 (-1.0)	9.1 (-0.4)	-60 (-170)	70 (-160)
ContextNet-M	4.5	10.0	70	270
+FastEmit	3.5 (-1.0)	8.6 (-1.4)	-110 (-180)	40 (-230)
ContextNet-L	4.4	8.9	50	210
+FastEmit	**3.1** (-1.3)	**7.5** (-1.4)	**-120** (-170)	**30** (-180)

We first present results of *FastEmit* on both Medium and Large size streaming ContextNet [8] and Conformer [9] in Table 1. We did a small hyper-parameter sweep of λ and set 0.01 for ContextNet and 0.004 for Conformer. *FastEmit* significantly reduces PR latency by $\sim$ **200ms**. It is noteworthy that streaming ASR models that capture stronger contexts can emit the full hypothesis even before they are spoken, leading to a **negative PR latency**. We also find *FastEmit* even improves the recognition accuracy on LibriSpeech. By error analysis, the deletion errors have been significantly reduced. As

LibriSpeech is long-form spoken-domain read speech, *FastEmit* encourages early emission of labels thus helps with vanishing gradients problem in long-form RNN-T [25], leading to less deletion errors.

4.2. Hyper-parameter λ in *FastEmit*

Table 2. Study of loss balancing hyper-parameter λ in *FastEmit* on LibriSpeech dataset, based on M-size streaming ContextNet [8].

FastEmit H-Param λ	WER TestClean	WER TestOther	PR50 (ms)	PR90 (ms)
0 (No FastEmit)	4.5	10.0	70	270
0.001	4.1 (-0.4)	8.7 (-1.3)	60 (-10)	190 (-80)
0.004	3.5 (-1.0)	8.4 (-1.6)	-30 (-100)	100 (-170)
0.008	3.6 (-0.9)	8.5 (-1.5)	-80 (-150)	50 (-220)
0.01	3.5 (-1.0)	8.6 (-1.4)	-110 (-180)	40 (-230)
0.02	3.8 (-0.7)	9.1 (-0.9)	-170 (-240)	-30 (-300)
0.04	4.4 (-0.1)	10.0 (0.0)	-230 (-300)	-90 (-360)

Next we study the hyper-parameter λ of *FastEmit* regularization by applying different values on M-size streaming ContextNet [8]. As shown in Table 2, larger λ leads to lower PR latency of streaming models. But when the λ is larger than a certain threshold, the WER starts to degrade due to the regularization being too strong. Moreover, λ also offers flexibility of WER-latency trade-offs.

4.3. Large-scale Experiments on MultiDomain

Table 3. Streaming ASR results of *FastEmit* RNN-T, Transformer-T and Conformer-T on a Voice Search test set compared with [2, 3, 10].

Method	WER	EP50 (ms)	EP90 (ms)	PR50 (ms)	PR90 (ms)
RNN-T	6.0	360	750	190	330
+CA [2, 3]	6.7 (+0.7)	450	860	-50 (-260)	60 (-250)
+MaskFrame	6.5 (+0.5)	250	730	100 (-90)	250 (-80)
+FastEmit	6.2 (+0.2)	330	650	-10 (-200)	180 (-150)
Transformer-T	6.1	400	780	220	370
+FastEmit	6.3 (+0.2)	390	740	60 (-160)	220 (-150)
Conformer-T	5.6	260	590	150	290
+FastEmit	5.8 (+0.2)	290	660	-110 (-260)	90 (-200)

Finally we show that *FastEmit* regularization method is also effective on the large scale production dataset MultiDomain. In Table 3, we apply *FastEmit* on RNN-Transducer [5], Transformer-Transducer [6] and Conformer-Transducer [9]. For RNN-T, we also compare *FastEmit* with other methods [2, 3, 10]. All results are finetuned with minimum word error rate (MWER) training technique [18] for fair comparison. In Table 3, *CA* denotes constrained alignment [2, 3], *MaskFrame* denotes the idea of training RNN-T models with incomplete speech by masking trailing n frames to encourage a stronger decoder thus can emit faster. We perform a small hyper-parameter search for both baselines *CA* and *MaskFrame* and report their WER, EP and PR latency on a Voice Search test set. *FastEmit* achieves **150 $\sim$ 300ms** latency reduction with significantly better accuracy over baseline methods in RNN-T [5], and generalizes further to Transformer-T [6] and Conformer-T [9]. By error analysis, as Voice Seach is short-query written-domain conversational speech, emitting faster leads to more errors. Nevertheless, among all techniques in Table 3, *FastEmit* achieves best WER-latency trade-off.

5. REFERENCES

[1] Bo Li, Shuo-yiin Chang, Tara N Sainath, Ruoming Pang, Yanzhang He, Trevor Strohman, and Yonghui Wu, "Towards fast and accurate streaming end-to-end asr," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6069–6073.

[2] Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.

[3] Tara N. Sainath, Ruoming Pang, David Rybach, Basi García, and Trevor Strohman, "Emitting Word Timings with End-to-End Models," *Proc. Interspeech*, 2020.

[4] Alex Graves, "Sequence Transduction with Recurrent Neural Networks," *CoRR*, vol. abs/1211.3711, 2012.

[5] Yanzhang He, Tara N. Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, Qiao Liang, Deepti Bhatia, Yuan Shangguan, Bo Li, Golan Pundak, Khe Chai Sim, Tom Bagby, Shuo-Yiin Chang, Kanishka Rao, and Alexander Gruenstein, "Streaming End-to-end Speech Recognition For Mobile Devices," in *Proc. ICASSP*, 2019.

[6] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7829–7833.

[7] Ching-Feng Yeh, Jay Mahadeokar, Kaustubh Kalgaonkar, Yongqiang Wang, Duc Le, Mahaveer Jain, Kjell Schubert, Christian Fuegen, and Michael L Seltzer, "Transformer-transducer: End-to-end speech recognition with self-attention," *arXiv preprint arXiv:1910.12977*, 2019.

[8] Wei Han, Zhengdong Zhang, Yu Zhang, Jiahui Yu, Chung-Cheng Chiu, James Qin, Anmol Gulati, Ruoming Pang, and Yonghui Wu, "Contextnet: Improving convolutional neural networks for automatic speech recognition with global context," *arXiv preprint arXiv:2005.03191*, 2020.

[9] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[10] Tara N Sainath, Yanzhang He, Bo Li, Arun Narayanan, Ruoming Pang, Antoine Bruguier, Shuo-yiin Chang, Wei Li, Raziel Alvarez, Zhifeng Chen, et al., "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6059–6063.

[11] Shuo-Yiin Chang, Bo Li, David Rybach, Yanzhang He, Wei Li, Tara Sainath, and Trevor Strohman, "Low latency speech recognition using end-to-end prefetching," in *Interspeech*. ISCA, 2020.

[12] Jiahui Yu, Wei Han, Anmol Gulati, Chung-Cheng Chiu, Bo Li, Tara N. Sainath, Yonghui Wu, and Ruoming Pang, "Universal asr: Unify and improve streaming asr with full-context modeling," *arXiv preprint arXiv:2010.06030*, 2020.

[13] Chengyi Wang, Yu Wu, Shujie Liu, Jinyu Li, Liang Lu, Guoli Ye, and Ming Zhou, "Low latency end-to-end streaming speech recognition with a scout network," *arXiv preprint arXiv:2003.10369*, 2020.

[14] Ehsan Variani, Tom Bagby, Kamel Lahouel, Erik McDermott, and Michiel Bacchiani, "Sampled connectionist temporal classification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4959–4963.

[15] William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals, "Listen, Attend and Spell," *CoRR*, vol. abs/1508.01211, 2015.

[16] Tara N. Sainath, Ruoming Pang, David Rybach, Yanzhang He, Rohit Prabhavalkar, Wei Li, Mirko Visontai, Qiao Liang, Trevor Strohman, Yonghui Wu, Ian McGraw, and Chung-Cheng Chiu, "Two-Pass End-to-End Speech Recognition," *Proc. Interspeech*, 2019.

[17] Wei Li, James Qin, Chung-Cheng Chiu, Ruoming Pang, and Yanzhang He, "Parallel rescoring with transformer for streaming on-device speech recognition," *arXiv preprint arXiv:2008.13093*, 2020.

[18] Rohit Prabhavalkar, Tara N. Sainath, Yonghui Wu, Patrick Nguyen, Zhifeng Chen, Chung-Cheng Chiu, and Anjuli Kannan, "Minimum Word Error Rate Training for Attention-based Sequence-to-Sequence Models," in *Proc. ICASSP*, 2018.

[19] Shuo-Yiin Chang, Bo Li, Tara N. Sainath, Gabor Simko, and Carolina Parada, "Endpoint Detection Using Grid Long Short-Term Memory Networks for Streaming Speech Recognition," in *Proc. Interspeech*, 2017.

[20] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[21] Arun Narayanan, Ananya Misra, Khe Chai Sim, Golan Pundak, Anshuman Tripathi, Mohamed Elfeky, Parisa Haghani, Trevor Strohman, and Michiel Bacchiani, "Toward domain-invariant speech recognition via large scale training," in *Proc. SLT*. IEEE, 2018, pp. 441–447.

[22] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[23] Bo Li, Tara N Sainath, Khe Chai Sim, Michiel Bacchiani, Eugene Weinstein, Patrick Nguyen, Zhifeng Chen, Yanghui Wu, and Kanishka Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," in *Proc. ICASSP*. IEEE, 2018, pp. 4749–4753.

[24] Jonathan Shen, Patrick Nguyen, Yonghui Wu, Zhifeng Chen, et al., "Lingvo: a modular and scalable framework for sequence-to-sequence modeling," *arXiv preprint arXiv:1902.08295*, 2019.

[25] Chung-Cheng Chiu, Wei Han, Yu Zhang, Ruoming Pang, Sergey Kishchenko, Patrick Nguyen, Arun Narayanan, Hank Liao, Shuyuan Zhang, Anjuli Kannan, et al., "A comparison of end-to-end models for long-form speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 889–896.