A Better and Faster End-to-End Model for Streaming ASR

Presenter: Bo Li (boboli@google.com)

Authors: Bo Li, Anmol Gulati, Jiahui Yu, Tara N. Sainath, Chung-Cheng Chiu, Arun Narayanan, Shuo-Yiin Chang, Ruoming Pang, Yanzhang He, James Qin, Wei Han, Qiao Liang, Yu Zhang, Trevor Strohman, Yonghui Wu

IEEE ICASSP 2021

Google

previously,

E2E models outperform conventional models in:

- quality (i.e. WER) [1]
- endpointer latency [2]
- but suffer from high partial latency

this work,

we present **a better quality and latency tradeoff** for streaming ASR by introducing:

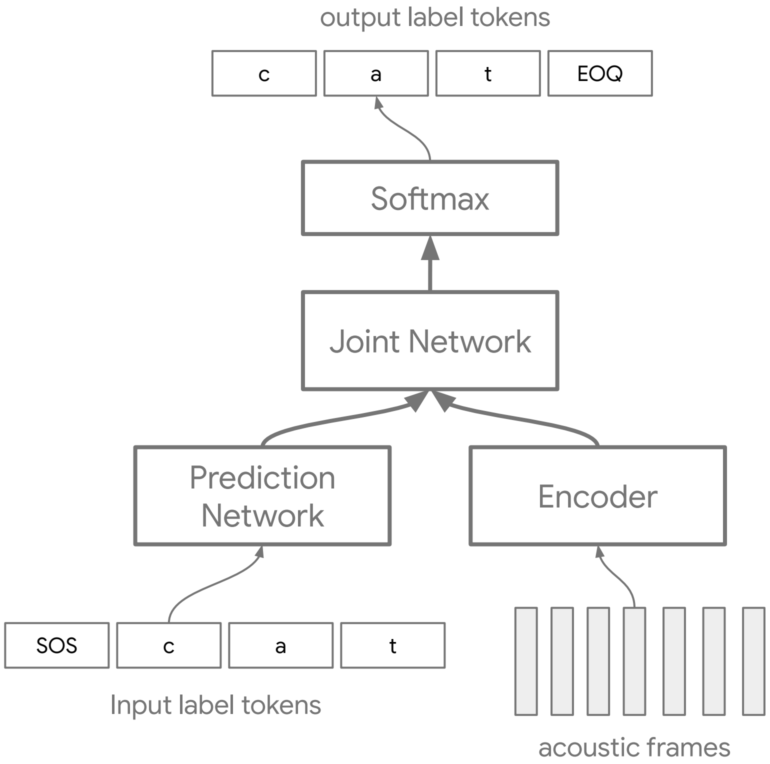
- Conformer Encoder[4]: for better quality
- Cascaded Encoders[5]: for better quality
- FastEmit[2]: for lower latency

Agenda

- Baseline System Architecture
- Quality
  - Conformer Encoder
  - Two-pass using Cascaded Encoders
- Latency
  - Metrics
  - Techniques
- Experiments
- Conclusions

System Architecture

RNN-T EP

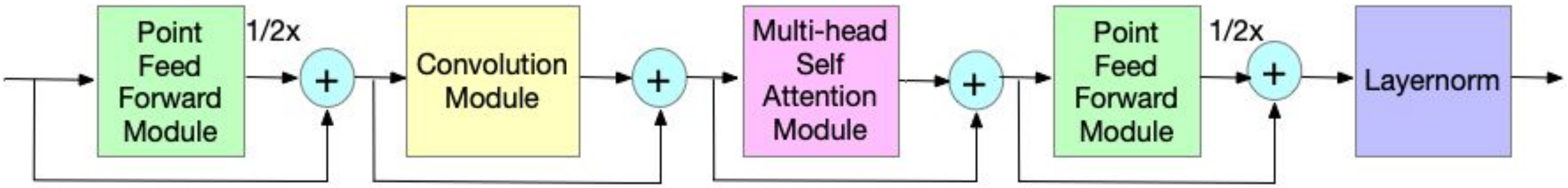


- LSTM Encoder
- LSTM Prediction Network
- Outputs: 4K WPM + EOQ

Quality Improvements

Conformer Encoder

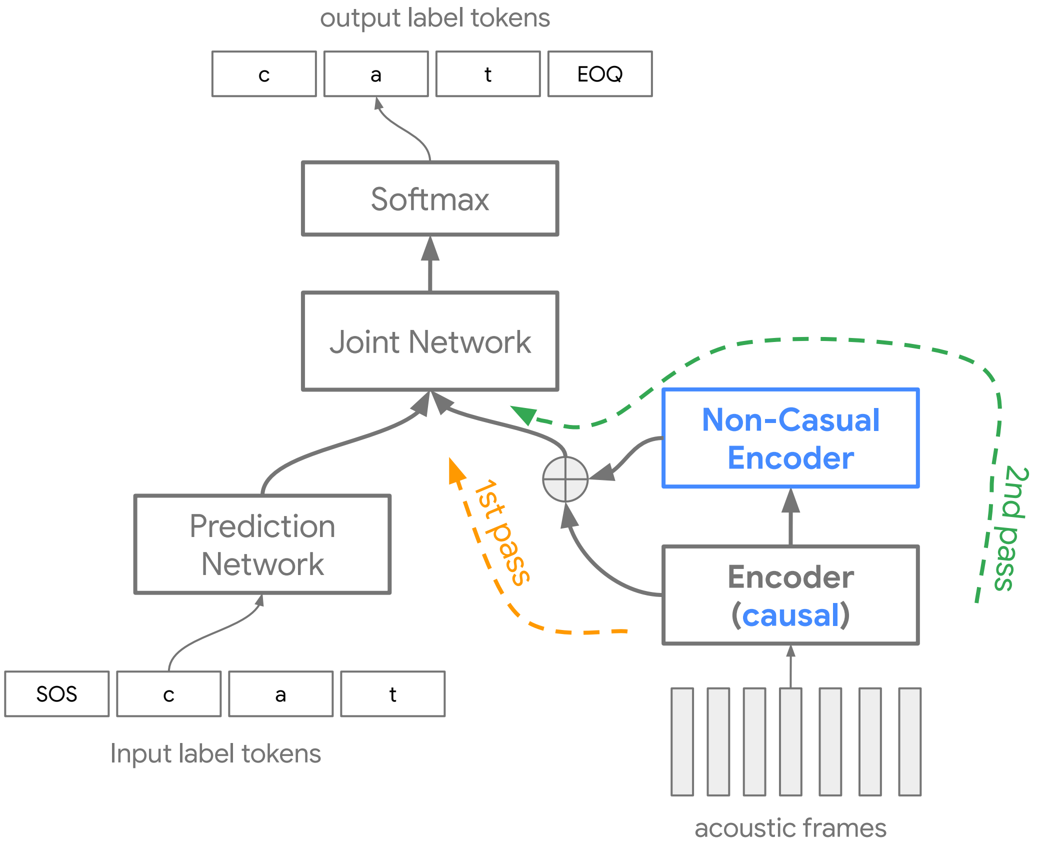
- We replace the encoder LSTM layers with Conformer [4]
- Changes to the existing Conformer:
  - Self-attention, convolution and normalization layers from full context → left context only for streaming applications;
  - Full context self-attentnion → local self-attention for better long-form generalization;
  - Batch normalization → group normalization[23] for multi-domain training data;
  - Relative positional encoding → reusing convolution for implicit positional information



Two-pass using Cascaded Encoder

- Two-pass models:
  - Fast 1st pass: sacrifice quality for better latency;
  - High quality 2nd pass: make up for the quality degradation in 1st pass.
- Conventional RNN-T + LAS[18]:
  - Rescoring limits the 2nd pass capability;
  - Attention models do poorly on long-form data [22].
- Cascaded Encoders Two-pass model:
  - Non-causal encoder layers ⟶ bringing in the full-context aspects of LAS for better quality;
  - RNN-T decoder with beam search for 2nd pass;
  - Sharing the RNN-T decoder between the two passes ⟶ smaller model size to fit on devices.

Cascaded Encoders

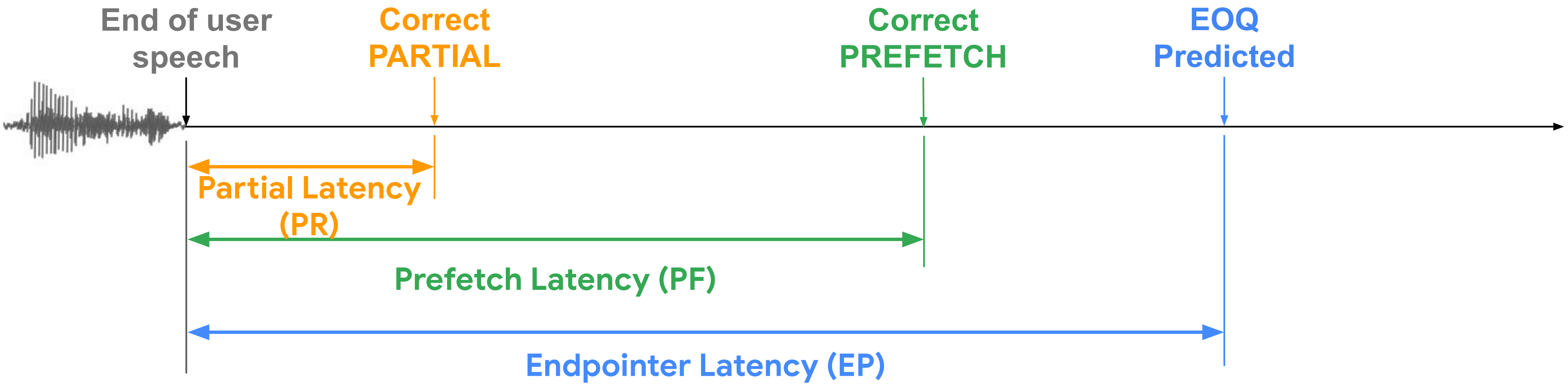


Latency Improvements

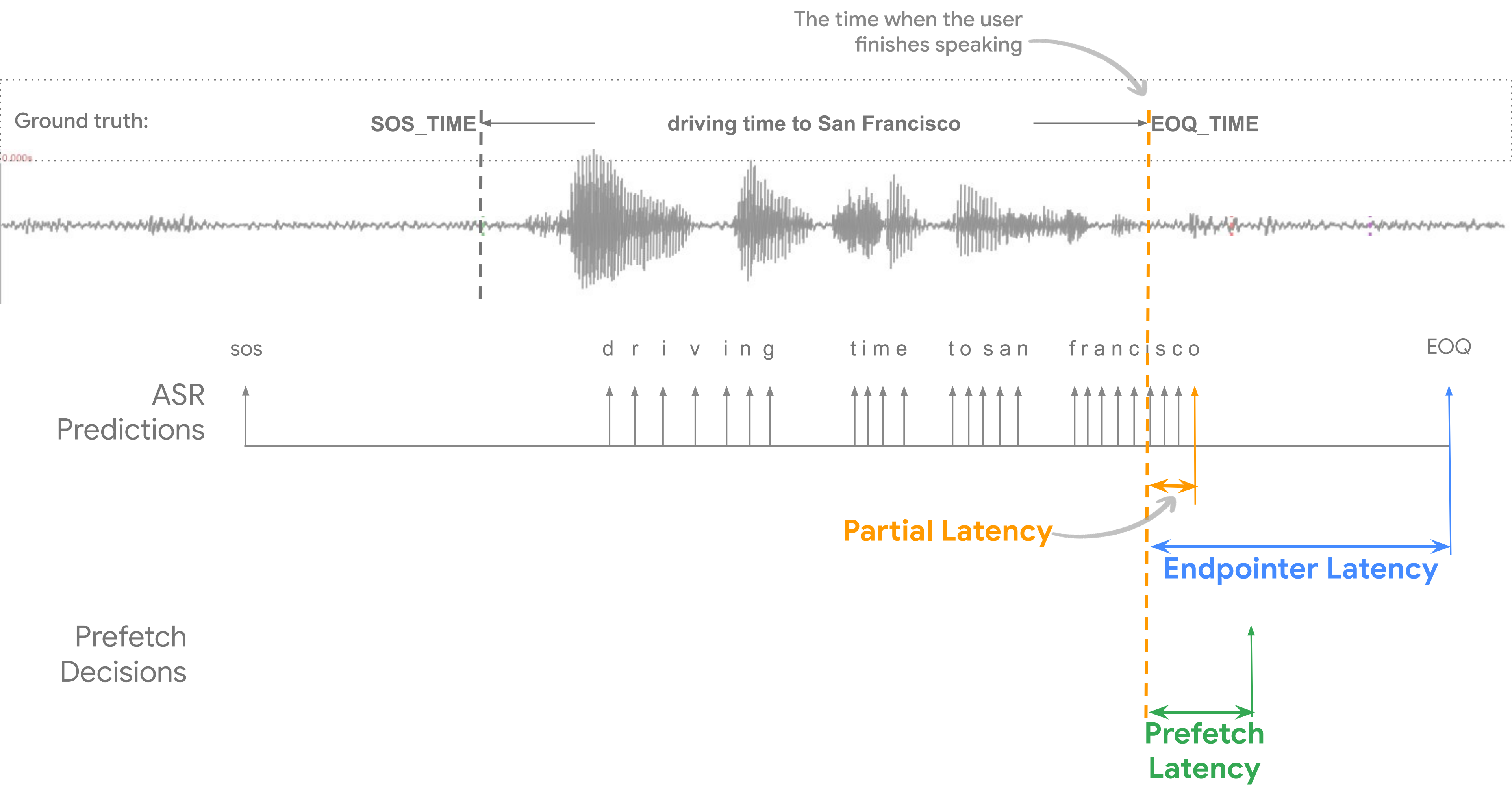
Latency Metrics

- Endpointer Latency (EP)
  - **Definition**: the time difference between when the user finishes speaking and when the system predicts the end of query (EOQ).
  - **Measures**: median (EP50) and 90th percentile (EP90) latency.
- Prefetch Latency (PF)
  - **Definition**: the time difference between when the first correct prefetch is trigged and when the user finishes speaking.
  - **Measures**: PF50 and PF90, together with the prefetching rate (PFR).
- Partial Latency (PR)
  - **Definition**: the time difference between when the first correct partial hypothesis is generated by the model and when the user finishes speaking.
  - **Measures**: PR50 and PR90.

Latency Metrics



- Partial latency is **inherent** to the model, while prefetch and endpointer latency depends on additional decision logic.
- Partial latency is the **lower bound** for prefetch latency.



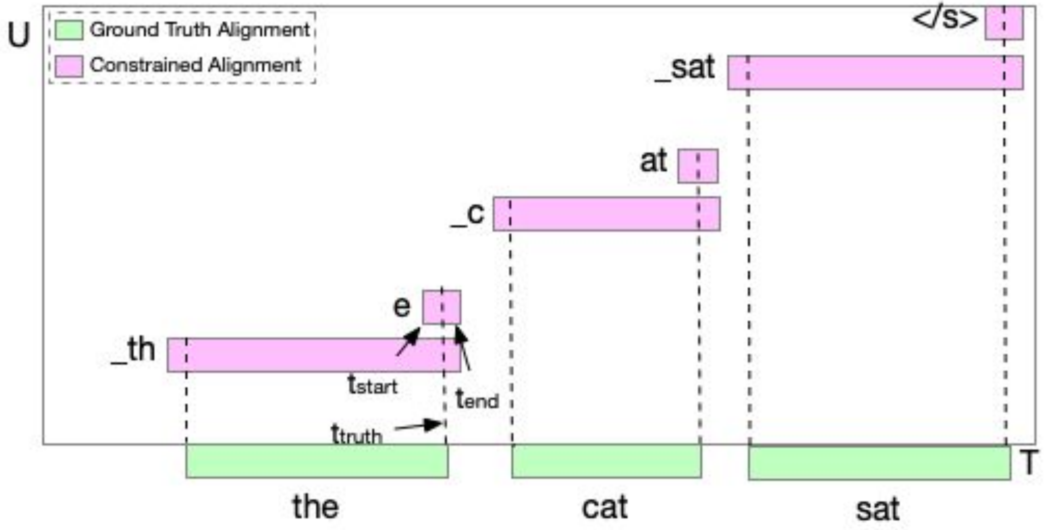
The time when the user finishes speaking
Ground truth:
SOS_TIME
driving time to San Francisco
EOQ_TIME
0.000s
sos
d r i v i n g
t i m e
t o s a n
f r a n c i s c o
EOQ
ASR Predictions
Partial Latency
Endpointer Latency
Prefetch Decisions
Prefetch Latency

Techniques

- Endpointer Latency (EP)
  - External EP
  - RNN-T EP: predicting EOQ jointly with ASR[2]
- Prefetch Latency (PF)
  - Silence based prefetching:
    - using voice activity detector (VAD);
    - triggers a prefetch after observing a fixed interval of silence.
  - E2E Prefetching[29]:
    - utilizing the EOQ prediction of the RNN-T EP model;
    - when EOQ probability is above a certain threshold, it declares a prefetch decision.
- Partial Latency (PR)
  - Constrained Aligments[20]
  - FastEmit[3]

Constrained Alignment

- Adds time constraints to RNN-T predictions.
  - It penalizes token predictions that are early or late.
  - In practice, we only constrain the start and end tokens of each word.



FastEmit

- Normally, emitting a **blank** or **non-blank** token are treated **equally** in RNN-T.
- In streaming ASR, however, emitting the blank token (i.e. delaying outputs) can lead to **higher latency**.
- We hence modify the RNN-T loss to suppress blank tokens.
- It is implemented by adding a **regularization term** in the original RNN-T formulations:

$$\frac{\partial \mathcal{L}}{\partial \mathrm{Pr}(k|t,u)} = -\frac{\alpha(t,u)}{\mathrm{Pr}(\mathbf{y}^*|\mathbf{x})} \begin{cases} (1+\lambda_{\text{FastEmit}})\beta(t,u+1) \text{ if } k = y_{u+1} \\ \beta(t+1,u) \text{ if } k = \varnothing \\ 0 \text{ otherwise.} \end{cases}$$

- Intuitively, it applies a "**higher learning rate**" to the prediction of non-blank token during back-propagation.

Experiments

Experiment Setup

- Dataset:
    - Human transcribed audio-text pairs from a variety of domains: Search, Farfield, Telephony, YouTube[1]
- Features:
    - 128D Log-mel Filterbanks together with a 1-hot vector of the domain-id to help with modeling domain variations.
- Models:
    - Causal Encoder: 17 causal (left-context only) Conformer layers;
    - Prediction Network: 2 layer LSTM.
    - Joint Network: a single feed-forward layer.
    - Non-causal Encoder: 2 layer Conformer layers with additional 5.04s right context.
- Metrics:
    - Quality: Word error rate (WER)
    - Latency: EP50, EP90, PR50, PR90, PF50, PF90, PFR

Quality Exps

- **B1**: LSTM encoder baseline system.
- **C0**: Simply limit Conformer[4] to use only-left contexts:
  - Different domains tend to have different length distribution, leading to biased batch normalization stats.
  - Removing bucketing resolves the quality degradation.
  - However, no bucketing slows down training.

Exp.	Model Size (M)	Training Speed (examples/sec.)	WER (%)
B1 LSTM w MWER [1]	122	3100	6.0
C0 w/o MWER [4]	141	3970	6.8
C0 No bucketing, w/o MWER	141	2450	5.8
C1 w/o MWER	141	3550	5.9
C2 w/o MWER	137	4200	5.8
w MWER			5.6

Quality Exps

- **C1**: With group normalization, we maintain similar WER but less speed regression.
- **C2**: Swapping the order of convolution and self-attention :
  - further improves the training speed
  - with MWER, it yields a 7% relative WER reduction and 35% speedup over LSTM.

Exp.	**Model Size** (M)	**Training Speed** (examples/sec.)	**WER** (%)
B1 LSTM w MWER [1]	122	3100	6.0
C0 w/o MWER [4]	141	3970	6.8
C0 No bucketing, w/o MWER	141	2450	5.8
C1 w/o MWER	141	3550	5.9
C2 w/o MWER	137	4200	5.8
w MWER			5.6

Latency Exps

Exp.	WER (%)	Endpointer Latency		Partial Latency		Prefetch Latency		
		EP50 (ms)	EP90 (ms)	PR50 (ms)	PR90 (ms)	PF50 (ms)	PF90 (ms)	PFR
B0 Conventional [1]	6.6	460	870	-150	60	90	190	1.48
B1 LSTM RNN-T [1]	6.0	310	710	170	310	170	320	1.80

- **B0**: Hybrid AM + LM Conventional baseline system.
- **B1**: Existing LSTM RNN-T
  - Good quality and EP latency
  - much worse PR and PF latencies.

Latency Exps

Exp.	WER (%)	Endpointer Latency EP50 (ms)	EP90 (ms)	Partial Latency PR50 (ms)	PR90 (ms)	Prefetch Latency PF50 (ms)	PF90 (ms)	PFR
B0 Conventional [1]	6.6	460	870	-150	60	90	190	1.48
B1 LSTM RNN-T [1]	6.0	310	710	170	310	170	320	1.80
B2 B1 + Constrained Alignment	6.9	230	560	-40	80	100	200	1.29
B3 B1 + FastEmit	6.2	330	650	-10	180	80	210	1.47

- **B2**: Constrained alignment reduces latency but hurts quality
- **B3**: FastEmit reduces latency with less quality regression

Latency Exps

Exp.	WER	Endpointer Latency		Partial Latency		Prefetch Latency		
	(%)	EP50 (ms)	EP90 (ms)	PR50 (ms)	PR90 (ms)	PF50 (ms)	PF90 (ms)	PFR
B0 Conventional [1]	6.6	460	870	-150	60	90	190	1.48
B1 LSTM RNN-T [1]	6.0	310	710	170	310	170	320	1.80
C2 Conformer RNN-T	5.6	260	590	150	290	220	350	1.65
C3 C2 + FastEmit	5.8	290	660	-110	90	70	210	1.29
C4 C3 + E2E Prefetch	6.0	290	660	-110	90	-50	110	1.86

- **C2**: Switching to Conformer encoder improves quality.
- **C3**: FastEmit improves partial latency.
- **C4**: E2E Prefetch reduces the gap between partial latency and prefetch latency.

C4 gives an E2E system with the **same quality** as the LSTM RNN-T but **much better latencies**.

Latency Exps

Exp.	WER (%)	Endpointer Latency EP50 (ms)	EP90 (ms)	Partial Latency PR50 (ms)	PR90 (ms)	Prefetch Latency PF50 (ms)	PF90 (ms)	PFR
B0 Conventional [1]	6.6	460	870	-150	60	90	190	1.48
B1 LSTM RNN-T [1]	6.0	310	710	170	310	170	320	1.80
C4 C3 + E2E Prefetch	6.0	290	660	-110	90	-50	110	1.86
T1 Two-pass LAS Rescoring	5.3	290	660	-100	140	80	210	1.30
T2 Single-pass Causal	6.0	290	660	-90	120	-20	130	1.90
T2 Two-pass	5.4	290	660	-80	140	0	140	1.84
T2 Single-pass Non-causal	4.8	-	-	-	-	-	-	-

- **T1**: Two-pass with LAS rescoring further improves quality.
- **T2**: Two-pass with Cascaded encoders:
  - maintains 1st pass latency gains;
  - reaches similar quality as T1;
  - even better quality for non-streaming applications with the same model.

Conclusions

- Confomer encoder brings further quality gains.
- FastEmit, a simple yet effective latency technique, brings E2E latency close to classical models.
- Two-pass model using Cascaded Encoders maintains 1st pass latency while further reducing WERs.

With these improvements, we can build a system that is better and faster than the previous best E2E system and surpassing the conventional model in quality and all latency metrics.

Thank you!

Contacts: boboli@google.com