SpeechWave

ICLR 2021

Dual-Mode ASR: Unify and Improve Streaming ASR with full-context Modelling

Erfan Loweimi

Centre for Speech Technology Research (CSTR)
University of Edinburgh

Listen!, 20 July 2021

SpeechWave

ICLR 2021

Dual-Mode ASR: Unify and Improve Streaming ASR with full-context Modelling

Erfan Loweimi

Centre for Speech Technology Research (CSTR)
University of Edinburgh

Listen!, 20 July 2021

Dual-mode ASR: Unify and Improve Streaming ASR with Full-context Modeling

Jiahui Yu[1] **Wei Han**[1†] **Anmol Gulati**[1†] **Chung-Cheng Chiu**[1] **Bo Li**[2]
Tara N. Sainath[2] **Yonghui Wu**[1] **Ruoming Pang**[1]

[1]Google Brain [2]Google LLC

{jiahuiyu, rpang}@google.com

Google Brain

ICLR 2021

Outline

- Detour: Two-pass E2E ASR
- Dual-mode E2E ASR: Streaming & Full-context
- Building Dual-mode Layers & Blocks
- Experimental Results
- Conclusion

Two-Pass End-to-End Speech Recognition

Tara N. Sainath, Ruoming Pang*, David Rybach, Yanzhang He, Rohit Prabhavalkar, Wei Li, Mirkó Visontai, Qiao Liang, Trevor Strohman, Yonghui Wu, Ian McGraw, Chung-Cheng Chiu*

Google, Inc., USA

{tsainath,rpang}@google.com

INTERSPEECH 2019

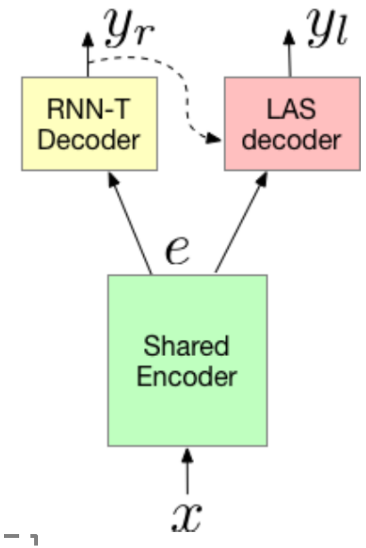
Joint training from scratch in unstable!

Training:

1) Train RNN-T (encoder & Decoder)
2) Freeze Enc, train LAS decoder
3) Fine-tune all jointly, using sum of losses

Possible decoding ways:

1) Beam search, LAS decode uses only RNN-T's $\mathbf{e}_{1:T}$ [not y_r]
2) Rescore RNN-T's Top-K hypothesises using LAS dec & **e**.



Related to *deliberation network* (NIPS 2017): refine first-pass decoding results in the second pass using global info.

Two-Pass End-to-End Speech Recognition

Tara N. Sainath, Ruoming Pang*, David Rybach, Yanzhang He, Rohit Prabhavalkar, Wei Li, Mirkó Visontai, Qiao Liang, Trevor Strohman, Yonghui Wu, Ian McGraw, Chung-Cheng Chiu*

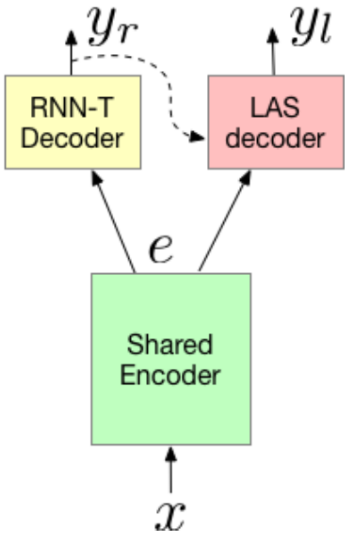
Google, Inc., USA

{tsainath,rpang}@google.com

INTERSPEECH 2019

Exp-ID	Decoding	SU	LU
B0	RNN-T	6.9	4.5
B1	LAS-only	5.4	4.5
E1	Beam Search	**6.1**	4.8
E2	Rescoring	6.2	**4.1**

Better WER, worse latency for streaming RNN-T (trade-off)



SU: short utterance (< 5.5 sec)
LU: long utterance (> 5.5 sec)
Latency increase: < 200*ms

* 200ms → limit of acceptable interactive latency

Streaming (on-line) ASR

- Emit each word hypothesis, on the fly
- Performance measures:
  - Recognition <u>Accuracy</u>
  - hypothesis emission <u>Latency</u>
- Challenge: No future context info (causality)
  - May be a limited look-ahead (e.g., 60 ms)
- Examples: CTC, RNN-T, ...

Full-context (off-line) ASR

- Await completion of an utterance before emitting complete hypothesis (decoding)
- Speech measure: Accuracy & Real time factor
- Example: (Attention) Encoder Decoder [RNN]
- Better performance than streaming
  - Access to future context

Streaming vs Full-context [E2E]

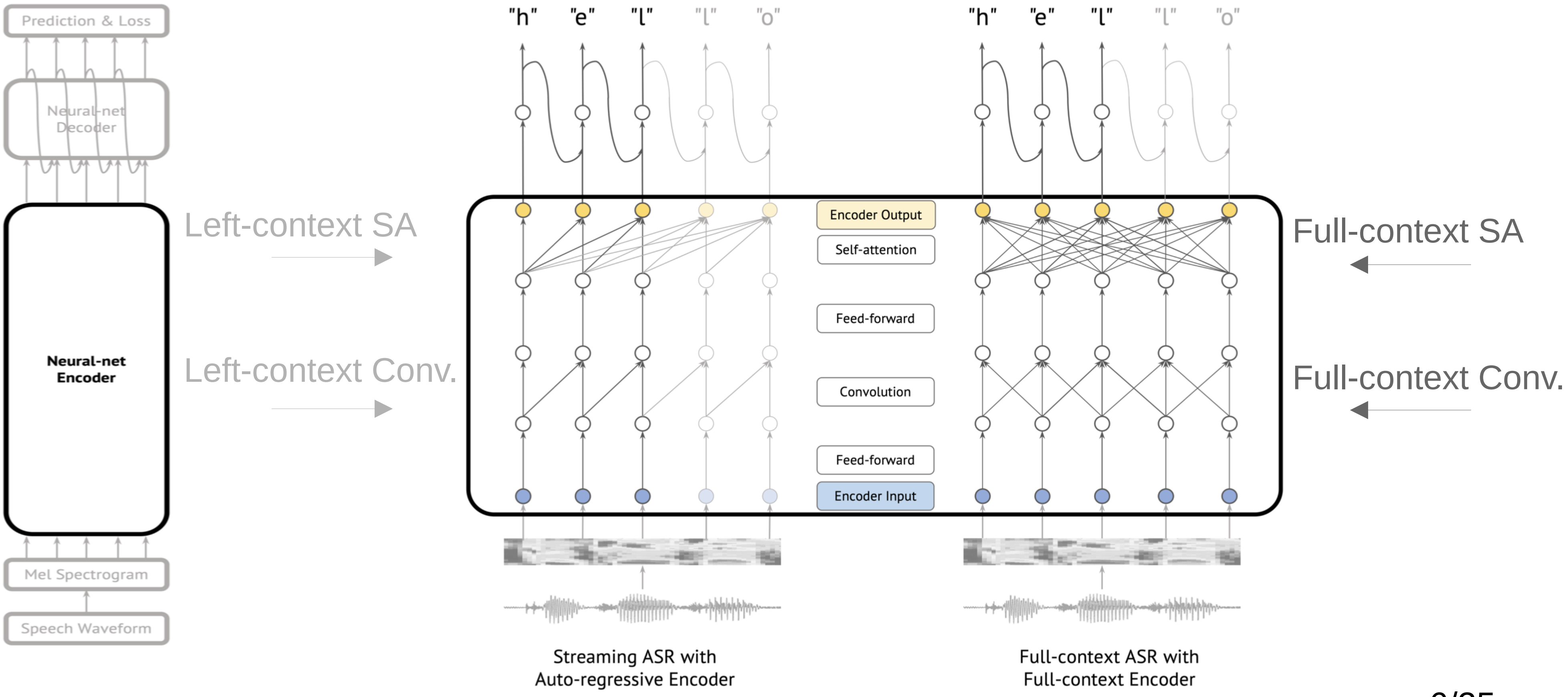
- Developed & deployed separately, although SIMILAR in many aspects ...
  - feature, data aug, Arch./reg./norm., objective function, training recipes, decoding method (AR*), ...

- Key DIFFERENCE ... Encoder ...
  - Full-context → $h_t = f_{enc}(x_{\mathbf{1:T}}, y_{history})$
  - Streaming → $h_t = f_{enc}(x_{\mathbf{1:t}}, y_{history})$ ← Causality [App. 1]

* h_t: encoder output @ t

* AR: Autoregressive

Streaming vs Full-context Encoder



E. Loweimi

This Paper … Dual-Mode ASR

- **What**: Unify streaming [RNN-T] and full-context ASR
- **How**: Weight sharing (WS), Joint training (JT), IPKD*
- **Why**: Improve latency & accuracy of streaming ASR
- **Core** element/challenge: Dual-mode Encoder
- Encoder Architectures:
  - Conformer [App. 2]
  - ContextNet [App. 3]

Dual-mode Encoder

- Need to (re)design some modules/layers
  - Conv. Layer, Pooling, Self-attention, Norm. layer, …
- Design principles for dual-mode modules/layers
  - … should be runnable in two modes [switch]
  - … with minimum additional parameter overhead
    - No separate modules! → Weight sharing

Design Dual-mode Layers (1)

- **Point-wise** operators are naturally dual-mode
  - No info propagation/processing across time
- Examples …
  - Point-wise FFNN in Transformer/conformer
  - Point-wise convolution (a.k.a. 1x1 convolution)
  - Skip/residual connection (x_t + f(x_t))
  - Dropout, Activation function, element-wise multiplication, ...

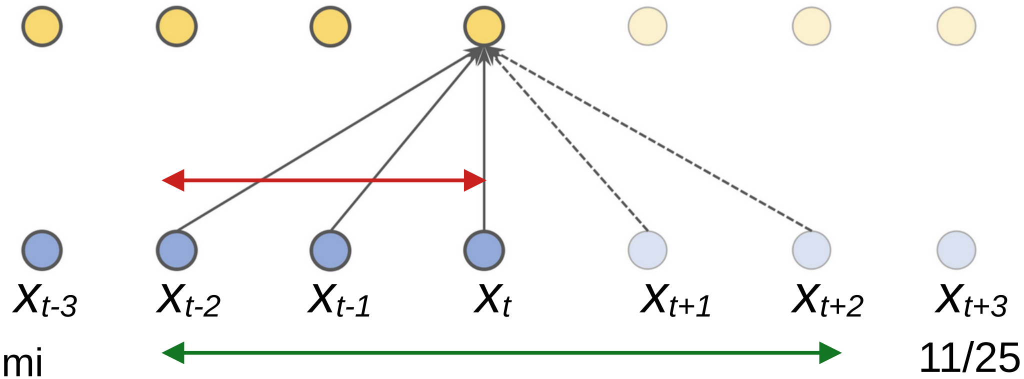
Design Dual-mode Layers (2)

- We need to redesign the following ...
  - Dual-mode (time-wise) convolution
  - Dual-mode (time-wise) pooling
  - Dual-mode self-attention
  - Dual-mode batch/layer normalisation

Dual-mode Conv. Layer

- Symmetric convolution (kernel size k) for full-context
- Causal convolution ((k+1)/2) for streaming
  - Kernel biased/skewed to left
- Additional params (rel. to streaming): ((k-1)/2)

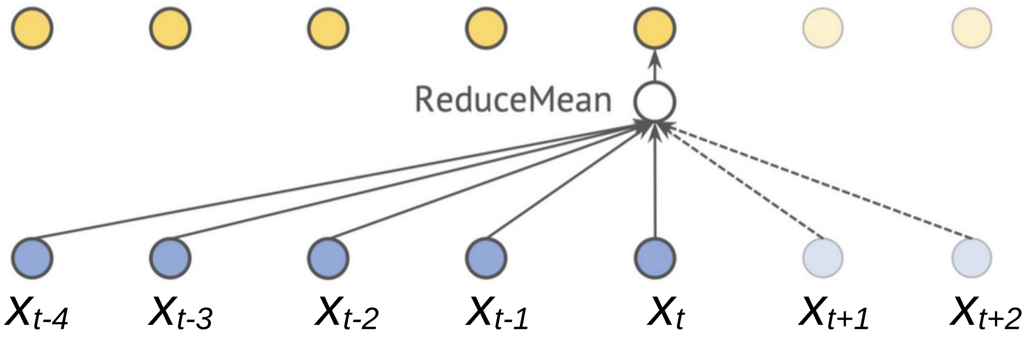
– Solid lines: Connected in both modes
-- Dash lines: Connected in full-context



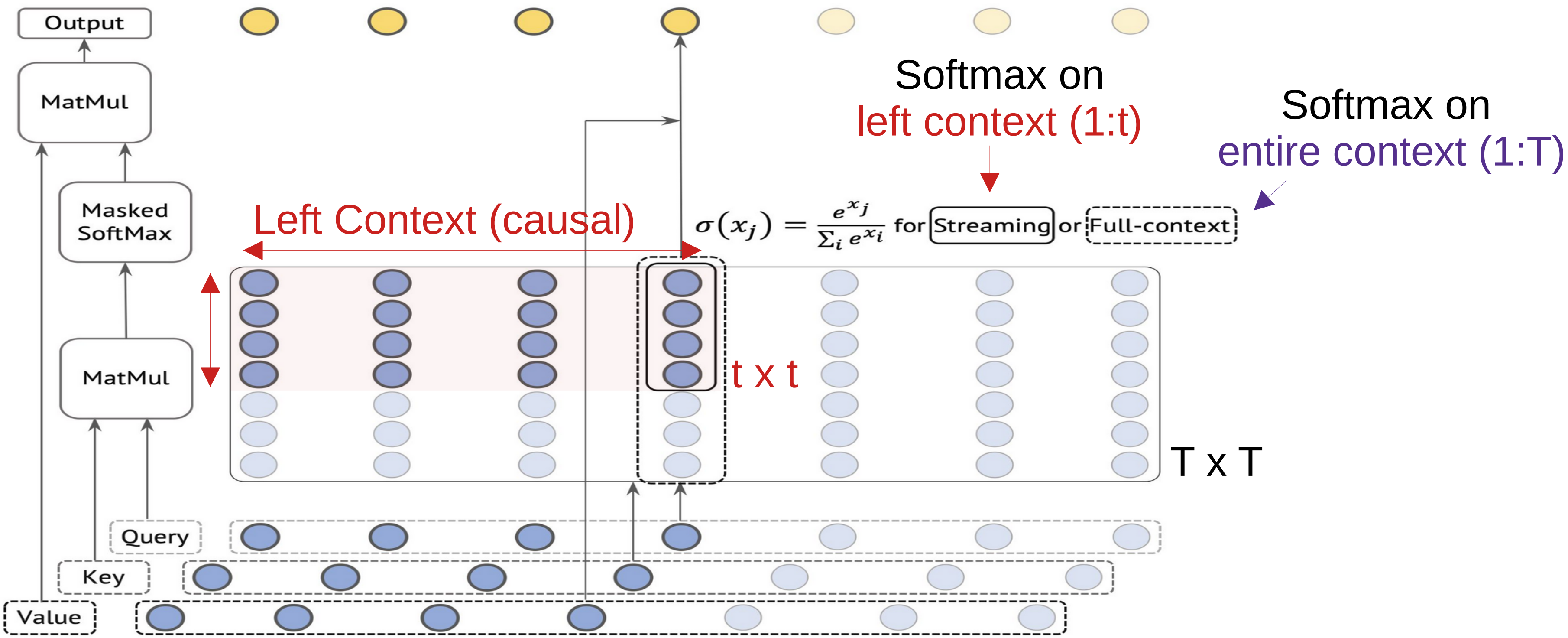
Dual-mode Pooling

- Squeeze-and-Excite layer in ContextNet
  - Use cumsum(1:t) instead of avg. over all T frames
- No additional parameter (pooling is param-free!)
- How about freq-wise Conv?

– Solid lines: Connected in both modes
-- Dash lines: Connected in full-context



Dual-mode Self-Attention



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}}V)$$

Dual-mode Batch/Layer Norm

- Stats of streaming & full-context are different
- For each mode a separate norm layer is instantiated
- No parameter sharing between modes

Training Modes

- Randomly sampled Training
  - Randomly choose the mode, update parameters
  - Control importance BY sampling probability
- Joint Training … aggregate losses ...
  - $\text{Loss} = w_1 \, \text{Loss}_{\text{Full-Context}} + w_2 \, \text{Loss}_{\text{Streaming}}$
  - Control importance BY weights
- Empirically ... Joint-training is better

Training Modes with IPKD

- In-place Knowledge Distillation (IPKD)
  - Teacher ≡ Full-context; student ≡ Streaming
  - “Inplace”: share weight + trained jointly, on the fly
  - Encourage consistency of predicted token probabilities

- Note: ONLY applicable with joint training regime
  - Loss = w_1 $\text{Loss}_{\text{Full-Context}}$ + w_2 $\text{Loss}_{\text{Streaming}}$ + w_3 $\text{Loss}_{\text{IPKD}}$
  - Here, $w_1 = w_2 = w_3$

Algorithm 1 Pseudocode of training Dual-mode ASR network (Joint training)

```
# Requires: data_loader; context manager with support of mode switching by network.mode();
    dual_mode_network with support of running both modes under context manager;

for x, y in data_loader: # Load a minibatch of speech input x and text label y.
   with dual_mode_network.mode('fullcontext'): # Switch context to 'fullcontext' mode.
      # Compute full-context prediction given speech input x and text label y.
      fullcontext_pred = dual_mode_network.forward_encoder_decoder(x, y)
      # Compute RNN-T loss of full-context mode.
      fullcontext_loss = rnnt_loss(fullcontext_pred, y)

   with dual_mode_network.mode('streaming'): # Switch context to 'streaming' mode.
      # Compute streaming prediction given speech input x and text label y.
      streaming_pred = dual_mode_network.forward_encoder_decoder(x, y)
      # Compute RNN-T loss of streaming mode.
      streaming_loss = rnnt_loss(streaming_pred, y)

   # Add inplace knowledge distillation loss (full-context prediction as teacher).
   distill_loss = inplace_distill_loss(streaming_pred, stop_gradient(fullcontext_pred))

   # Compute total loss as a sum of full-context, streaming and distillation losses.
   loss = fullcontext_loss + streaming_loss + distill_loss
   loss.backward() # Update weights.
```

.detach() in PyT

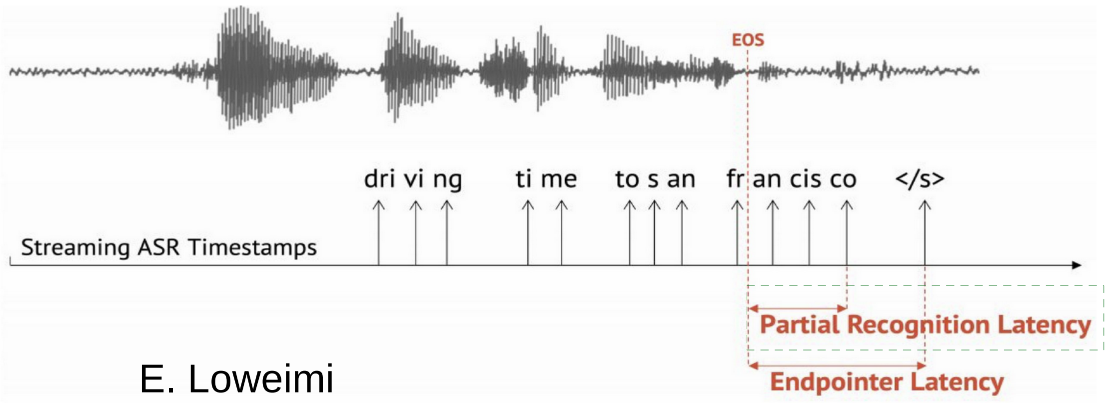
Stop_gradient: Loss$_{IPKD}$ does not backpropagate through computation graph of the full_context model (only affects streaming mode parameters).

Experimental Setup

- Exactly following the baseline models settings
  - SpecAug, Adam, learning rate scheduling/warm-up
- Instead of mean, Latency@X% Percentile reported (robust)
  - X% Percentile = InverseCDF(X/100); Median = InverseCDF(0.5)
- Data:
  - LibriSpeech (970h, #u: 281k) [Reading]
  - MultiDomain (413kh, #u: 287M) [Voice Search, Far-field, YouTube, Meeting]

Measuring Latency

- Latency Measure (ms): $t_2 - t_1$
  - t_1: when speaker stop speaking (EOS)
  - t_2: when last token is emitted in finalised results
- Note: Negative latency … Emit full hypothesis before speaker finishes is possible! ← strong context modelling or too large EOS(?!)



Experimental Results – MultiDomain

Method	Mode	# Params (M)	VS Test WER(%)	Latency@50 (ms)	Latency@90 (ms)
ContextNet	Full-context	133	5.1	——	——
Conformer	Full-context	142	5.2	——	——
LSTM (Sainath et al., 2020)	Streaming	179	6.4	190	350
ContextNet (Han et al., 2020)	Streaming	133	6.1	160	310
Conformer (Gulati et al., 2020)	Streaming	142	6.1	160	300
Dual-mode ContextNet	Full-context Streaming	133	4.9 6.0 (-0.1)	—— 10 (-150)	—— 220 (-90)
Dual-mode Conformer	Full-context Streaming	142	5.0 6.0 (-0.1)	—— -50 (-210)	—— 130 (-170)

* Latency of Streaming mode improves remarkably
* … Negative latency with Conformer! What if t_1 is wrong (too large EOS)?!
* Accuracy improves marginally (for both modes)

Experimental Results – LibriSpeech

Method	Mode	# Params (M)	Test Clean/Other WER(%)	Latency@50 (ms)	Latency@90 (ms)
LSTM-LAS	Full-context	360	2.6 / 6.0	——	——
QuartzNet-CTC	Full-context	19	3.9 / 11.3	——	——
Transformer	Full-context	29	3.1 / 7.3	——	——
Transformer	Full-context	139	2.4 / 5.6	——	——
ContextNet	Full-context	31.4	2.4 / 5.4	——	——
Conformer	Full-context	30.7	2.3 / 5.0	——	——
Transformer	Streaming	18.9	5.0 / 11.6	80	190
ContextNet	Streaming	31.4	4.5 / 10.0	70	270
Conformer	Streaming	30.7	4.6 / 9.9	140	280
ContextNet Look-ahead	Streaming	31.4	4.1 / 9.0	150	420
Dual-mode Transformer	Full-context	29	3.1 / 7.9	——	——
	Streaming		4.4 (-0.6) / 11.5 (-0.1)	-50 (-130)	30 (-160)
Dual-mode ContextNet	Full-context	31.8	2.3 / 5.3	——	——
	Streaming		3.9 (-0.6) / 8.5 (-1.5)	40 (-30)	160 (-110)
Dual-mode Conformer	Full-context	30.7	2.5 / 5.9	——	——
	Streaming		3.7 (-0.9) / 9.2 (-0.7)	10 (-130)	90 (-190)

60 ms

Look-ahead
WER ↓
Latency ↑

Conformer has lower latency than ContextNet

ContextNet ... 60ms look-ahead

Ablation Study, Streaming Mode

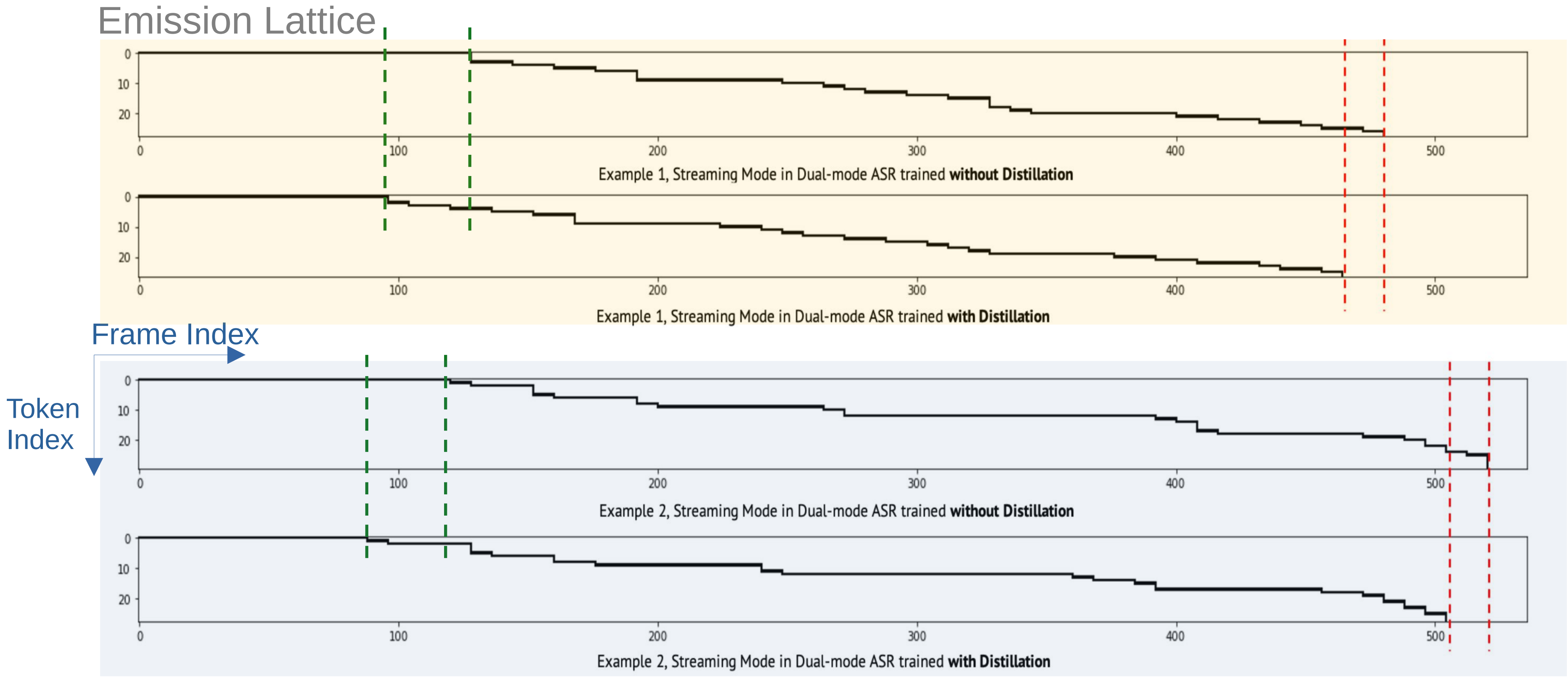
LibriSpeech

	Weight Sharing	Joint Training	Inplace Distillation	TestOther WER(%)	Latency@50 (ms)	Latency@90 (ms)
0)	✔	✔	✔	8.5	40	160
1)	✔	✔	✘	10.2 (+1.7)	120 (+80)	310 (+150)
2)	✔	✘	✘	10.6 (+2.1)	90 (+50)	290 (+130)
3)	✘	✔	✔	9.9 (+1.4)	50 (+10)	210 (+50)

* [0,1] IPKD improves both accuracy & latency
* [1,2] Randomly Sampled training (Joint-training = off): worse WER, better latency
* [0,3] W/0 weigh sharing (train 2 separate models): worse accuracy & latency

    ** Weight sharing effect → smaller, better and faster model

Dual-mode decreases latency ...



Reviewers Comments

- Initial title was “*Universal ASR*” ↔ over-stating
  - Universal could have many dimensions, e.g.,
    - close vs distant talking, single vs multi-lingual, clean vs noisy, narrow vs wideband, single vs multi-domain, ...
- Application to simultaneous machine translation (MT)?
  - Unlike ASR, behaviour of online & offline MT systems could be very different

Conclusion

- **Dual-mode E2E ASR**: Streaming (online) + Full-context (offline)
- **Goal**: improve emission latency and accuracy of streaming ASR
- **Tricks**: weight sharing, joint training, in-place knowledge distillation
- **Challenge**: Redesigning <u>encoder</u> layers to operate in dual-mode
- **Tasks**: LibriSpeech & MultiDomain
- **Architectures**: ContextNet & Conformer
- **Results**: SOTA emission latency and recognition accuracy
  - … up to some negative latency for Conformer (& Transformer)

Thanks for Your Attention!

- Q&A

- Appendices:
  - (A1) Causal vs Autoregressive
  - (A2) Conformer
  - (A3) ContextNet
  - (A4) Squeeze-and-Excite Module

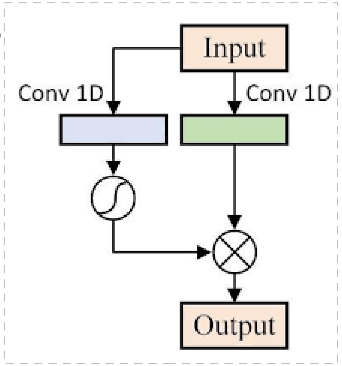
Causality vs Auto-regressive

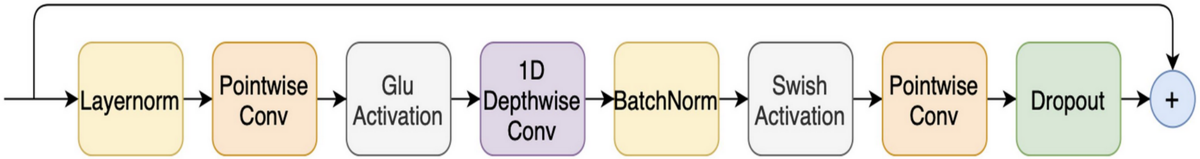
- Causality: $Y_{t0} = f(X_{1:t0})$; not any $t > t_0$
  - Characterisation based on input-output relationship
- Auto-regressive (AR): $Y_{t0} = f(Y_{t<t0}, X)$
  - Characterisation based on output-output relationship
  - Antonym: Moving Average (MA) $\rightarrow$ $Y_{t0} = f(X, [\text{NO Y HERE}])$
- Example of Causal layers:
  - Uni-dir. RNNs, Causal convolution, left-context Self-attention
- Example of Auto-regressive layers:
  - Uni-dir. RNNs, decoder

Conformer

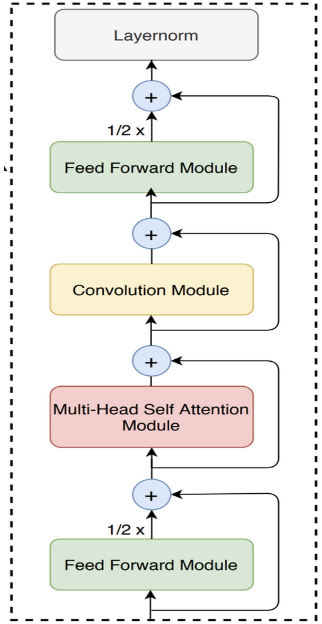
- Combines Convolution and Transformer
- To model both local (Convolution) and global [Self-attention] dependencies
- Separable Convolution

GLU



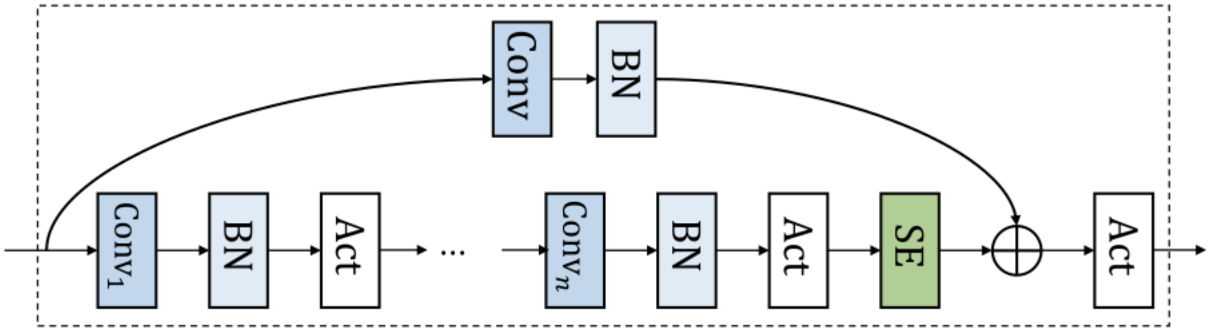


Convolution Module



ContextNet

- Depth-wise separable convolution + SE* layer + Swish activation
- Similar to QuartzNet [Jasper with separable conv]



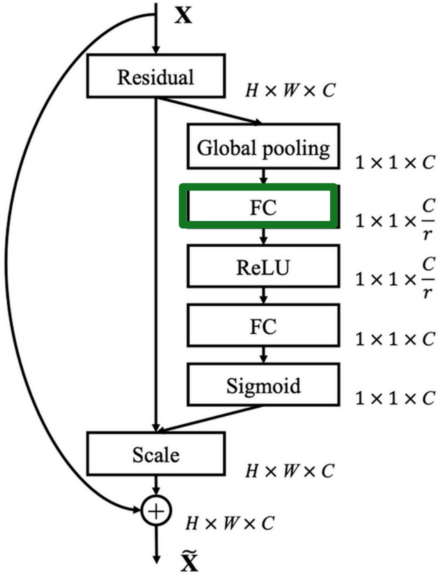
Convolution modules in ContextNet

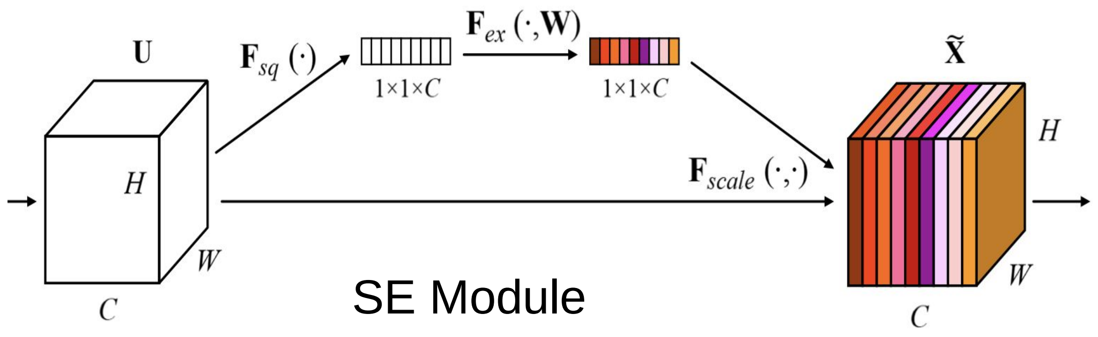
* SE: Squeeze-and-Excite

Squeeze-and-Excite (SE) Module

- Goal: Weight/Scale channels using channel interdependencies
- Module: AvgPool → Linear (r) → ReLU → Linear (1/r) →Sigmoid
  - r: compression ratio (bottleneck), e.g. 8 or 16

SE-ResNet Module





SE Module