CS 224S / LINGUIST 285
Spoken Language Processing

Andrew Maas
Stanford University
Spring 2022

Lecture 11: Recent end-to-end ASR approaches. Building responsible systems.

Outline

- Advanced end-to-end ASR models
  - RNN-Transducer loss
  - On-device Google voice search with RNN-T models
  - Convolution-augmented transformer
- Building responsible ML systems

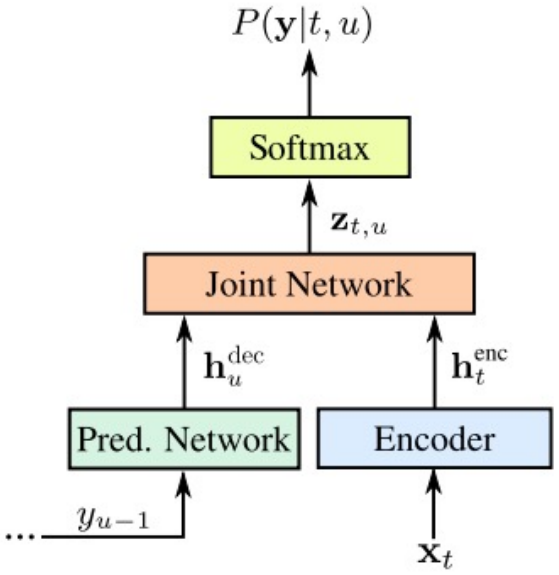
Reminders

- Project proposals due tonight by 11:59pm
  - Assume your proposal is overall good. Start work!
  - Look for specific feedback/suggestions via Gradescope.

- This Thursday 5/5 our first guest lecture!
  - **Arlo Faria**. Years of experience building ASR systems in research and industry settings
  - Attend live or synchronously via Zoom.
    - 1% of your final grade for the course for just showing up!
    - If you can't attend synchronously, ask a question in advance on Ed

RNN-Transducer loss

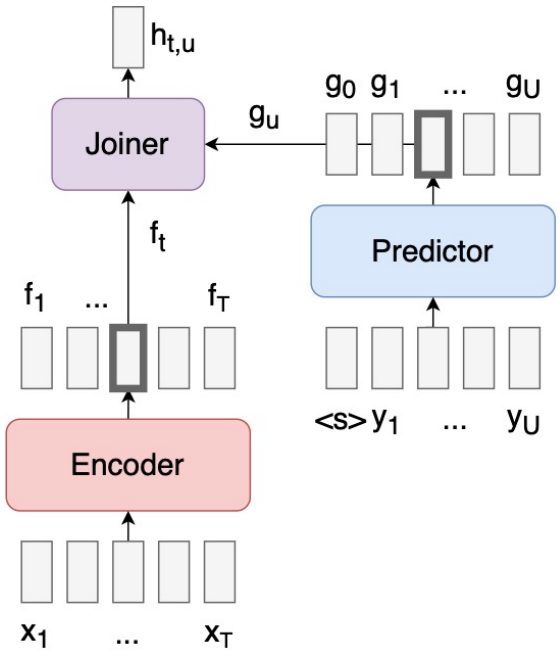
- Directly optimizes target word sequence as correct label
  - Graphemes (letters) or word parts (10k-50k) used in practice
- Learned combination of acoustic + language model pieces
- Conditions on sequence output so far (y_{t-1})
- Misnomer: Does not require RNNs!

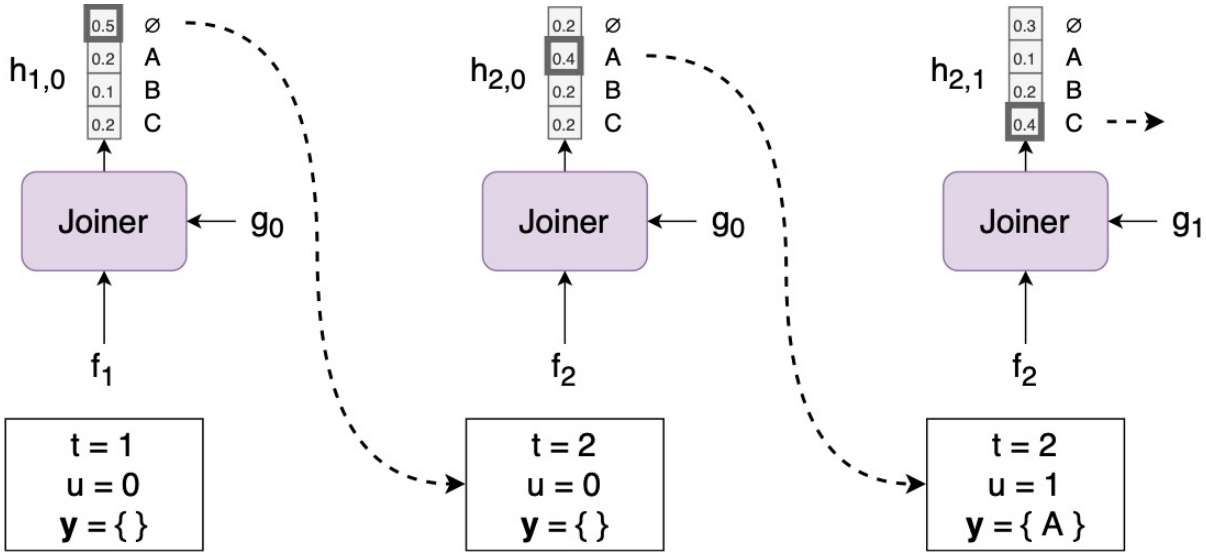


RNN-T loss: (Graves, 2012) Figure: Rao, Sak, & Prabhavalkar. 2017)

RNN-Transducer loss computation

- Causal inference structure. Works for online decoding
- Predictor inputs are only non-blank tokens (*y*)
- Do not increment t when non-blank token is output

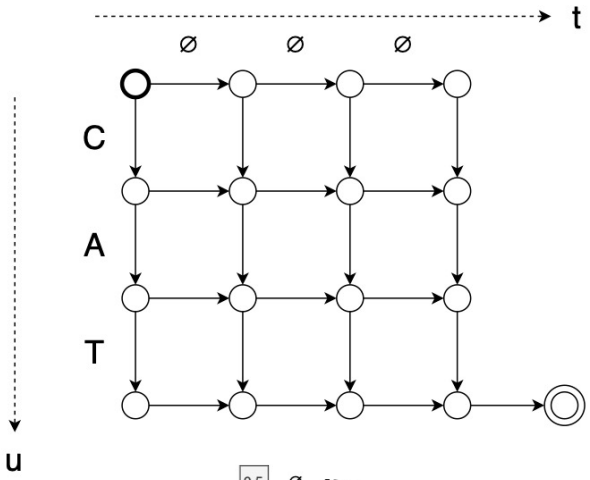


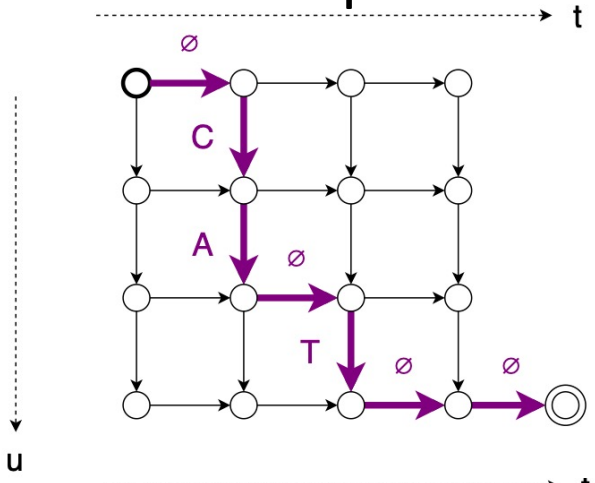


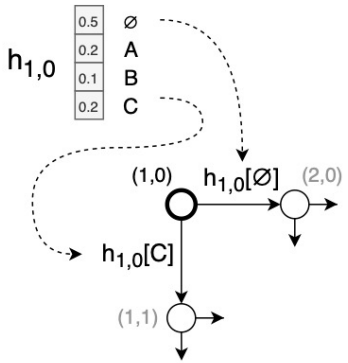
Figures from Loren Lugosch

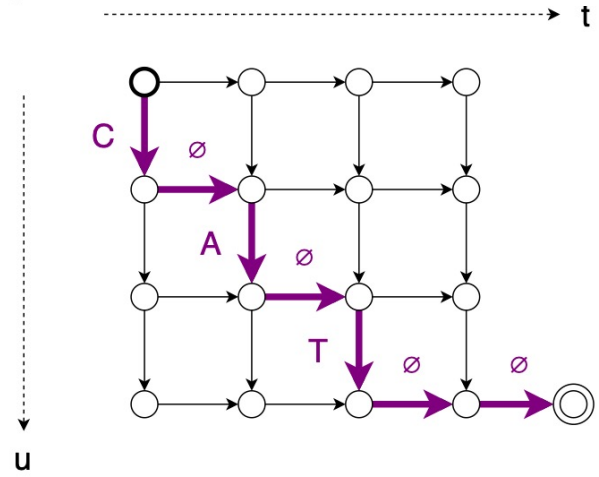
RNN-Transducer loss computation

- Many alignments consistent with true transcript
- Dynamic programming like CTC. No repeated output collapse









Figures from Loren Lugosch

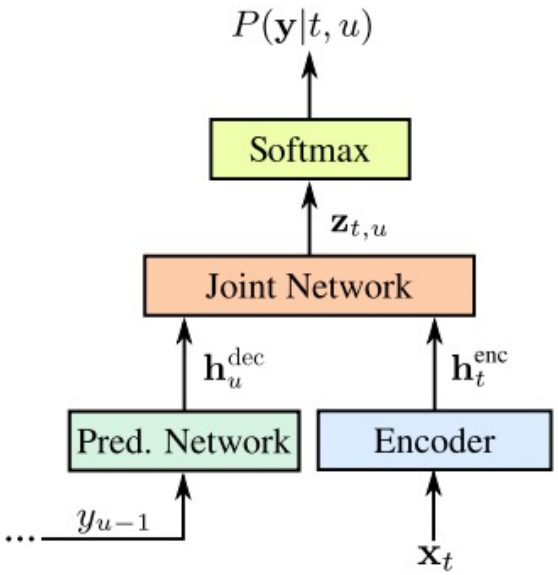
RNN-Transducer loss

- Maximize P(y|x) by summing over all consistent alignments
  - Multiple alignments due to *blank.*
  - No repeated output collapsing

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{z} \in \mathcal{Z}(\mathbf{y}, T)} P(\mathbf{z}|\mathbf{x})$$

- Single alignment:

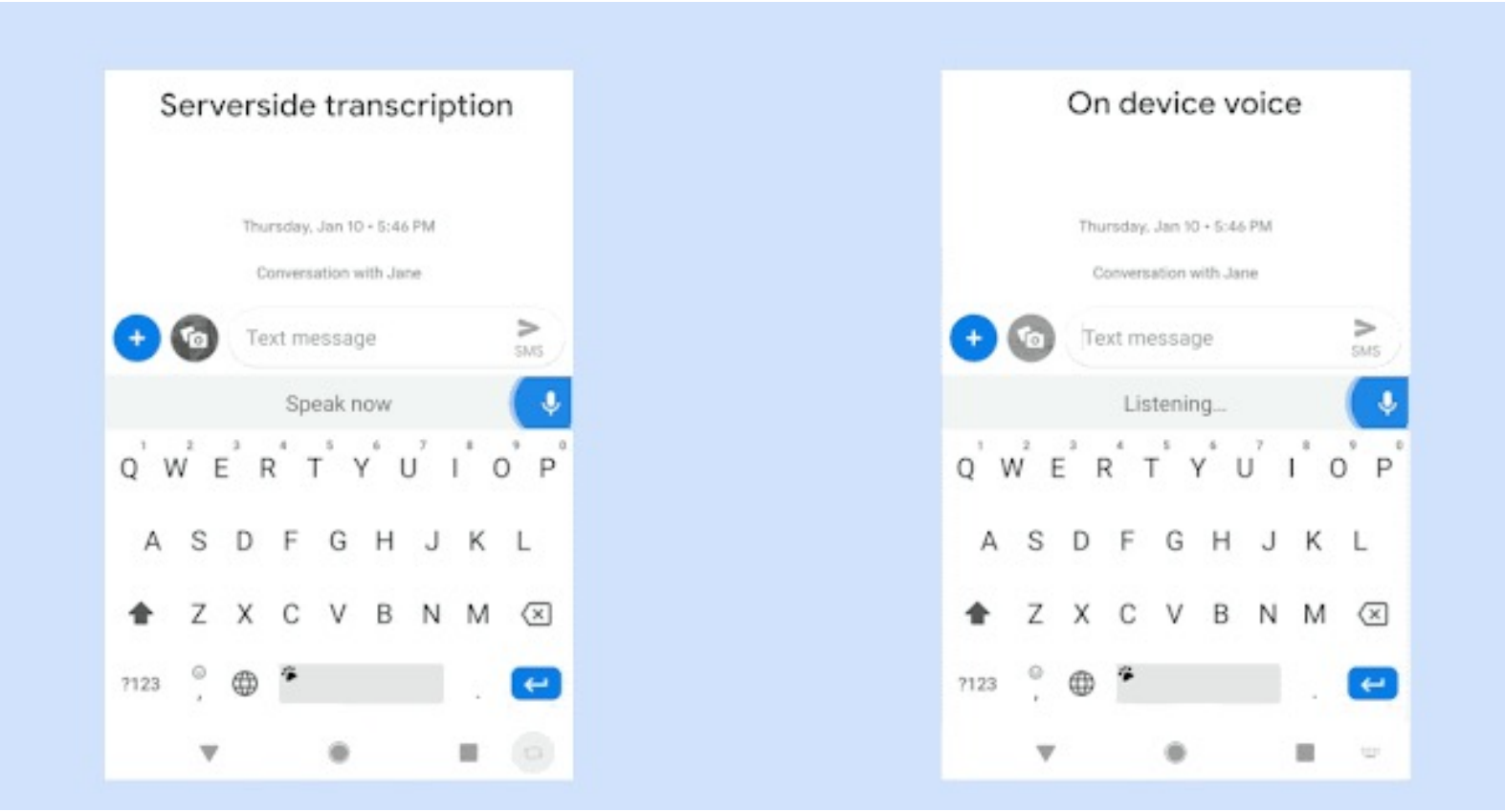
$$P(\mathbf{z}|\mathbf{x}) = \prod_i P(z_i|\mathbf{x}, t_i, \text{Labels}(z_{1:(i-1)}))$$



RNN-T loss: (Graves, 2012) Figure: Rao, Sak, & Prabhavalkar. 2017)

Google on-device ASR enabled by RNN-T

- It works! All neural, large vocabulary, high quality ASR running just on a phone (no cloud server required)



(Google blog post, 2019) (He, Sainath, et al. arXiv, 2018)

Google on-device ASR enabled by RNN-T: Enabling innovations

- What innovations made it possible to go from 2GB models and cloud computation to compact, on-device models?

- Decoding: Beam search with a single NN instead of weighted finite state transducer decoding machinery
- NN parameter quantization.
  - 4x model size compression. 4x runtime speed improvement
- LM contextual biasing. User-specialized LM to upweight common requests / inputs
- Scaling up training with parallel RNN-T.
- Improved text normalization + sub-word output units

(Google blog post, 2019) (He, Sainath, et al. arXiv, 2018)

Conformer: Convolution-augmented Transformer for Speech Recognition

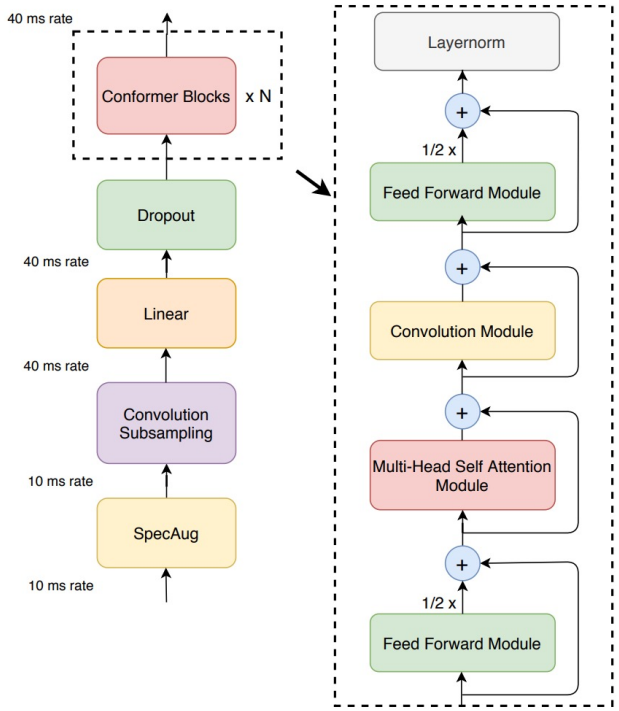


Figure 1: ***Conformer encoder model architecture.*** *Conformer comprises of two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed self-attention and convolution modules. This is followed by a post layernorm.*

- Sequence-to-sequence transformer with multi-headed self attention.
- Transformer encoder combines attention (global context) with convolution (local invariance)
- RNN-T loss architecture

(Gulati *et al.* 2020)

Conformer: Convolutional transformer encoder

- Sequence-to-sequence transformer with multi-headed self attention. Directly optimizes target word sequence
- Combines attention (global context) with convolution (local invariance)

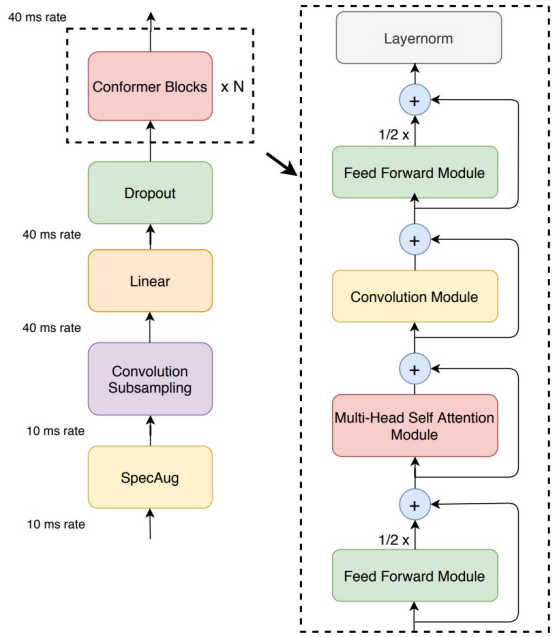


Figure 1: ***Conformer encoder model architecture.*** *Conformer comprises of two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed self-attention and convolution modules. This is followed by a post layernorm.*

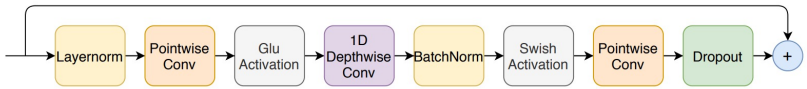


Figure 2: ***Convolution module.*** *The convolution module contains a pointwise convolution with an expansion factor of 2 projecting the number of channels with a GLU activation layer, followed by a 1-D Depthwise convolution. The 1-D depthwise conv is followed by a Batchnorm and then a swish activation layer.*

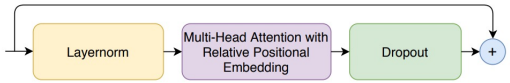


Figure 3: ***Multi-Headed self-attention module.*** *We use multi-headed self-attention with relative positional embedding in a pre-norm residual unit.*

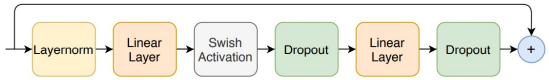


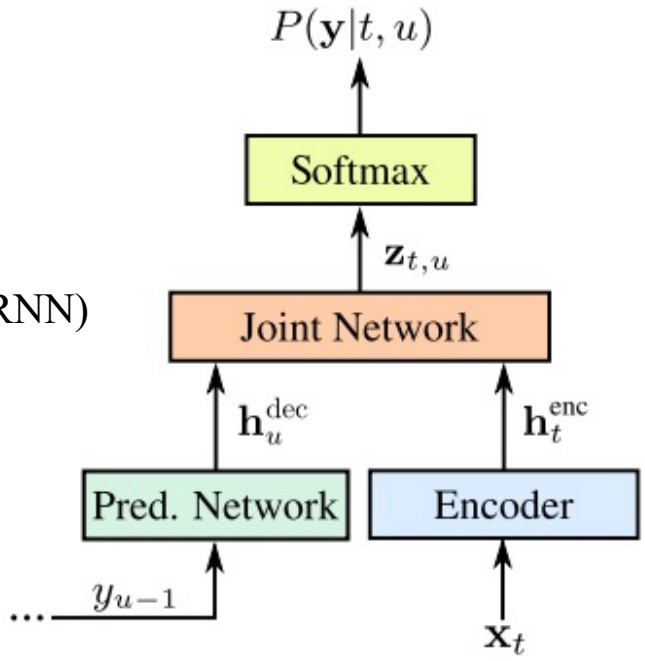
Figure 4: ***Feed forward module.*** *The first linear layer uses an expansion factor of 4 and the second linear layer projects it back to the model dimension. We use swish activation and a pre-norm residual units in feed forward module.*

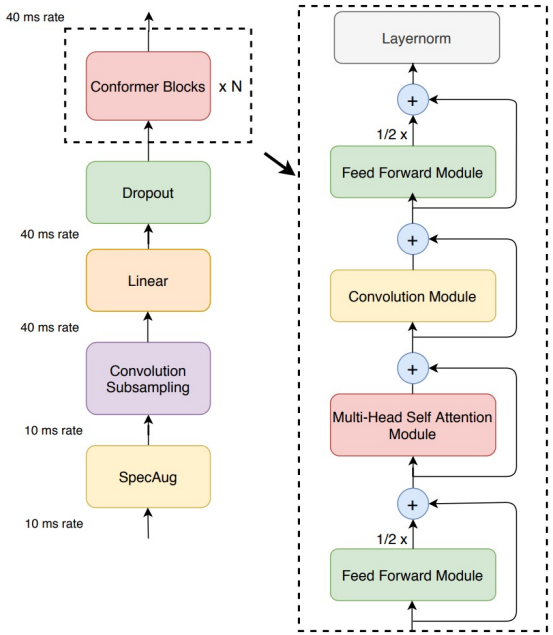
(Gulati *et al.* 2020)

Conformer: Putting it all together

1 hidden layer LSTM (RNN)

3 hidden layer LSTM (RNN). Pre-trained as language model





(Gulati *et al.* 2020)

Conformer: Convolution-augmented Transformer for Speech Recognition

Table 2: *Comparison of Conformer with recent published models. Our model shows improvements consistently over various model parameter size constraints. At 10.3M parameters, our model is 0.7% better on testother when compared to contemporary work, ContextNet(S) [10]. At 30.7M model parameters our model already significantly outperforms the previous published state of the art results of Transformer Transducer [7] with 139M parameters.*

Method	#Params (M)	WER Without LM		WER With LM	
		testclean	testother	testclean	testother
Hybrid					
Transformer [33]	-	-	-	2.26	4.85
CTC					
QuartzNet [9]	19	3.90	11.28	2.69	7.25
LAS					
Transformer [34]	270	2.89	6.98	2.33	5.17
Transformer [19]	-	2.2	5.6	2.6	5.7
LSTM	360	2.6	6.0	2.2	5.2
Transducer					
Transformer [7]	139	2.4	5.6	2.0	4.6
ContextNet(S) [10]	10.8	2.9	7.0	2.3	5.5
ContextNet(M) [10]	31.4	2.4	5.4	**2.0**	4.5
ContextNet(L) [10]	112.7	**2.1**	4.6	**1.9**	4.1
Conformer (Ours)					
Conformer(S)	10.3	**2.7**	**6.3**	**2.1**	**5.0**
Conformer(M)	30.7	**2.3**	**5.0**	**2.0**	**4.3**
Conformer(L)	118.8	**2.1**	**4.3**	**1.9**	**3.9**

(Gulati *et al.* 2020)

Dual Mode ASR: Joint encoder + training for streaming & full context models

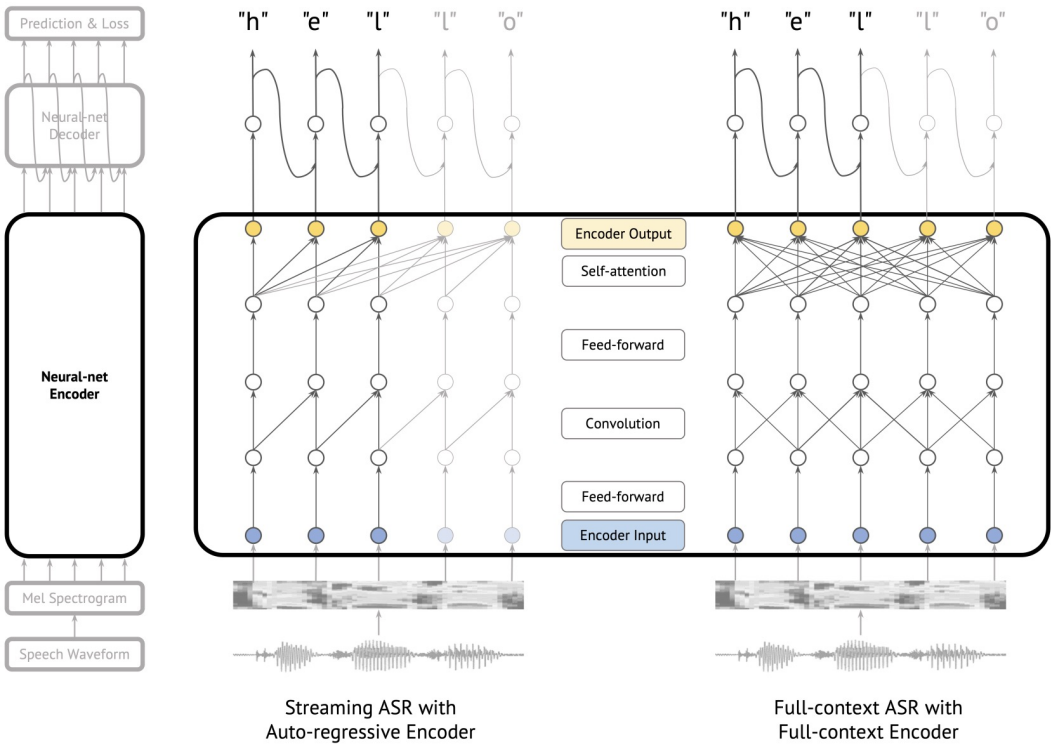


Figure 1: A simplified illustration of the similarity and difference between Streaming ASR and Full-context ASR networks. Modern end-to-end streaming and full-context ASR models share most of the neural architectures and training recipes in common, with the most significant difference in the **ASR encoder (highlighted)**. Streaming ASR encoders are auto-regressive models, with each prediction of the current timestep conditioned on previous ones (no future context). We show examples of feed-forward layer, convolution layer and self-attention layer in the encoder of streaming and full-context ASR respectively. With Dual-mode ASR, we unify them without parameters overhead.

(Yu *et al.* 2021)

Dual Mode ASR: Joint encoder + training for streaming & full context models

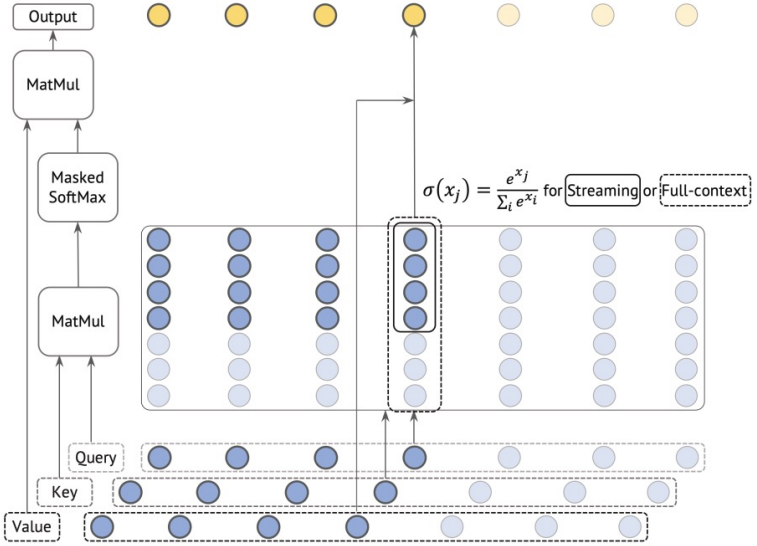
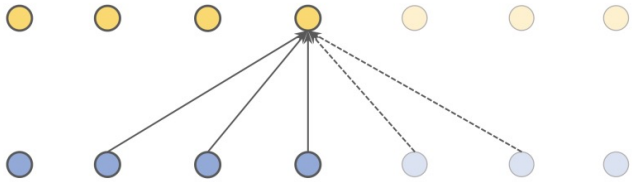


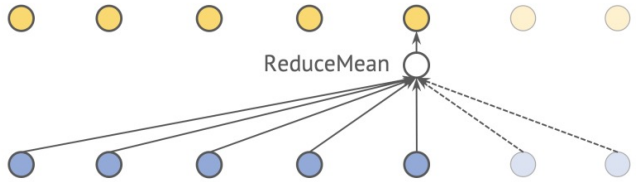
Figure 3: Dual-mode self-attention layer.

: Connected in both streaming and full-context mode.

: Connected only in full-context mode.



Dual-mode Temporal-wise Convolution
(kernel size: 3 or 5)



Dual-mode Temporal-wise Average Pooling
(left-context pool vs. full-context pool)

(Yu *et al.* 2021)

Dual Mode ASR: Joint encoder + training for streaming & full context models

Table 3: Summary of our results on Librispeech dataset (Panayotov et al., 2015). We report WER on TestClean and TestOther (noisy) set. Compared with standalone ContextNet and Conformer models, Dual-mode ASR models have both higher accuracy in average and better streaming latency.

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)

Table 4: Ablation studies of weight sharing, joint training and inplace distillation. We report WER on TestOther (noisy) set (Panayotov et al., 2015) using ContextNet with same training settings.

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

(Yu *et al.* 2021)

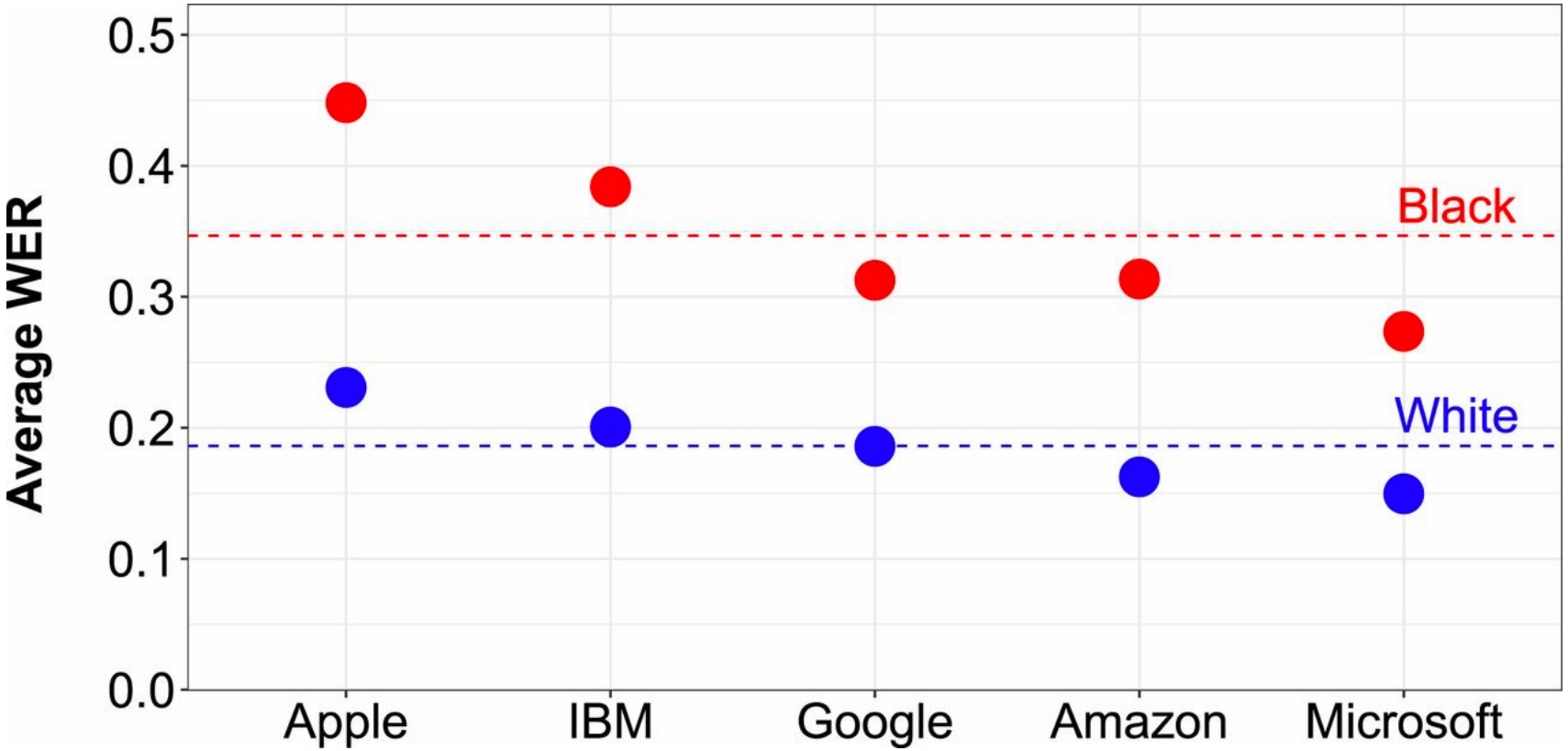
Outline

- Advanced end-to-end ASR models
  - RNN-Transducer loss
  - On-device Google voice search with RNN-T models
  - Convolution-augmented transformer
- **Building responsible ML systems**

Negative impacts of ML systems

- Broad categorization for this discussion:
  - Harmful system: negative impact on people or the world
  - Biased system: performs differently, or not at all, for some subpopulation of inputs

ML group bias in ASR: Racial disparities



(Koenecke *et al.* 2020)

ML group bias in ASR: Racial disparities

- 23% of audio snippets from black speakers have WER > 0.5. Compared to 1.6% of audio snippets from white speakers

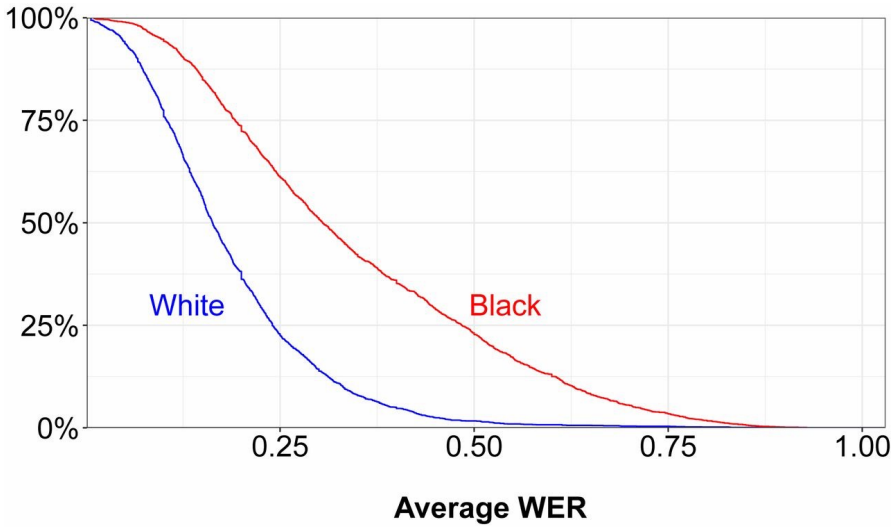


Table 2.
Error rates on a matched subset of identical short phrases spoken by white and black individuals in our sample

	Average WER for black speakers	Average WER for white speakers
Apple	0.28	0.12
IBM	0.21	0.10
Google	0.17	0.11
Amazon	0.18	0.08
Microsoft	0.13	0.07

(Koenecke *et al.* 2020)

ML group bias in ASR: Racial disparities

- Causes? Differences attributed to acoustic model quality (rather than language model)
  - Training data distribution bias?
  - Pronunciation lexicon assumptions a potential factor

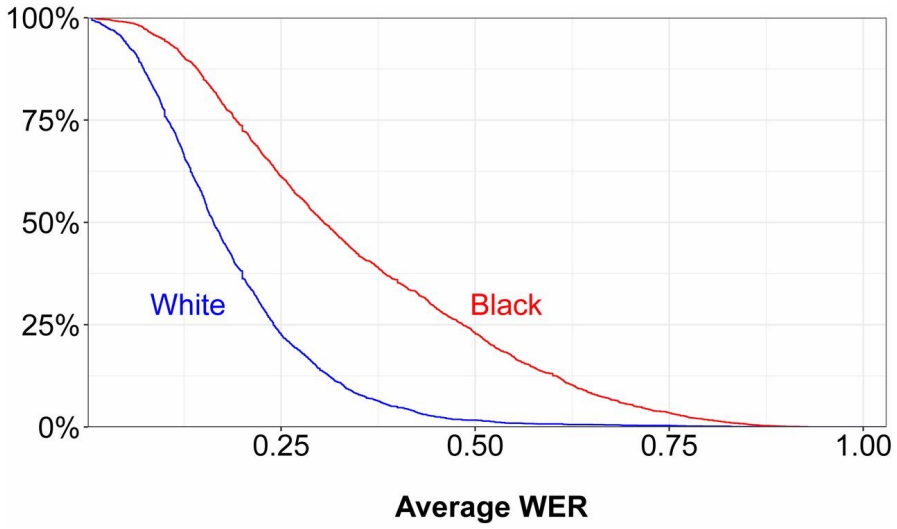


Table 2.
Error rates on a matched subset of identical short phrases spoken by white and black individuals in our sample

	Average WER for black speakers	Average WER for white speakers
Apple	0.28	0.12
IBM	0.21	0.10
Google	0.17	0.11
Amazon	0.18	0.08
Microsoft	0.13	0.07

(Koenecke *et al.* 2020)

Negative impacts of ML systems

- What causes negative impacts of ML?
  (think beyond just spoken language systems here)

Negative impacts of ML systems: Common causes

- System design issues
- Insufficiently trained / low performing ML models
- Distribution shift and mis-application

System design issues

- Problem formulation / ML task should be ethical
- Depends little on particulars of an ML model
- Example:
  - Predict future criminality from face image
  - In speech, emotion recognition is sometimes controversial
    - Difficult to categorize the true range of emotions, and how individuals express them in speech
    - Need to be careful of how recognized emotions are used in a broader system.

Insufficiently trained systems

- ML is never perfect. Potential negative impacts of mistakes
  - Lower quality experience for some inputs/users
- Often ML mistakes are not uniform random
  - Performance issues might disproportionately affect certain subgroups of input examples
  - Spoken language inputs are often correlated with demographic or medical factors affecting a person's voice

Slides from co:rise Applied ML course

System design + data distributions

- Define and announce the expectations for a system
- Fundamentally comes back to (1) training data distribution and (2) generalization performance of models
  - Ensure training data is representative
  - Develop “unit test” evaluation sets to ensure sub-group performance is adequate
- Current research on detecting and fixing group bias

Distribution shift and misapplication of models

- We assume test/production data and training data are IID
  - == training set is representative of all future inputs
- Distribution shift: When test/production distribution is different or drifts from IID relative to training data
- Misapplication: Using a model on a production/test distribution for which it was not trained / evaluated

Slides from co:rise Applied ML course

Model bias and multiple languages

- Models can have assumptions that work well in some, but not all languages
  - E.g. Turkish has complex morphology (→ many possible words) and loose word ordering (→ n-gram models perform poorly)
  - Performance is worse in systematic ways.
  - Possible to adapt systems. Important to test for this when working with multi-lingual systems
  - Deep learning approaches can encode fewer assumptions
- *Low resource languages* might create problems for deep learning approaches. Often large training sets required.

Data collection

- Be aware of local laws if collecting audio data
  - *California's wiretapping law is a "two-party consent" law. California makes it a crime to record or eavesdrop on any confidential communication, including a private conversation or telephone call, without the consent of all parties to the conversation. (source)*
- Generally data should be collected for a purpose
- Participants should be aware of any secondary uses of the data
  - GDPR makes this secondary use requirement more explicit

Summary: Building responsible ML

- Focus: Build quality ML models that do what they claim!
- Thoughtful system design
- Test for and document subgroup performance issues
- Data provenance, transparency (and privacy) is central

- Realize that YOU may be the only informed person in critical decisions about where/how to apply ML

Appendix