Google

Towards End-to-End Speech Recognition

Bo Li, Yanzhang He, Shuo-Yiin Chang

ISCSLP Tutorial 4

Nov. 26, 2018

Acknowledgement

Google Brain Team

- Raziel Alvarez
- William Chan
- Jan Chorowski
- Chung-Cheng Chiu
- Zhifeng Chen
- Navdeep Jaitly
- Anjuli Kannan
- Patrick Nguyen
- Ruoming Pang
- Colin Raffel
- Ron Weiss
- Yonghui Wu
- Yu Zhang

Google Speech Team

- Michiel Bacchiani
- Tom Bagby
- Deepti Bhatia
- Alexander Gruenstein
- Shankar Kumar
- Qiao Liang
- Ian McGraw
- Golan Pundak
- Rohit Prabhavalkar
- Kanishka Rao
- David Rybach
- Tara Sainath
- Johan Schalkwyk
- Vlad Schogol
- (June) Yuan Shangguan
- Khe Chai Sim
- Gabor Simko
- Trevor Strohman
- Zelin Wu
- Ding Zhao
- Kazuki Irie (intern)
- Shubham Toshniwal (intern)

Acknowledgement

The content of this tutorial is mostly based on the following tutorial with recent updates. Many slides are borrowed with authors' consent.

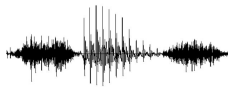
End-to-End Models for Automatic Speech Recognition

Rohit Prabhavalkar and Tara Sainath
Tutorial T2, Interspeech 2018
http://interspeech2018.org/program-tutorials.html

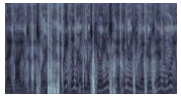
What is End-to-End ASR?

Conventional ASR

Pipeline



Input Speech



Feature Extraction

DNN/RNN Acoustic Model

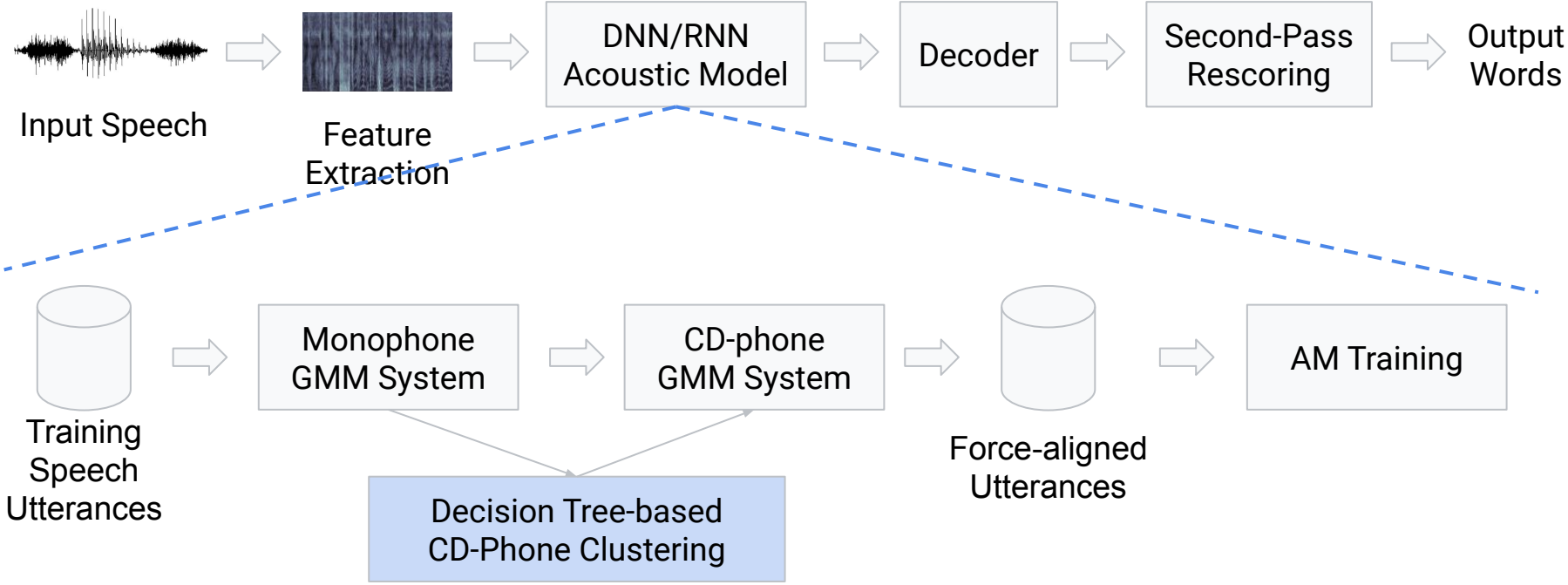
Decoder

Second-Pass Rescoring

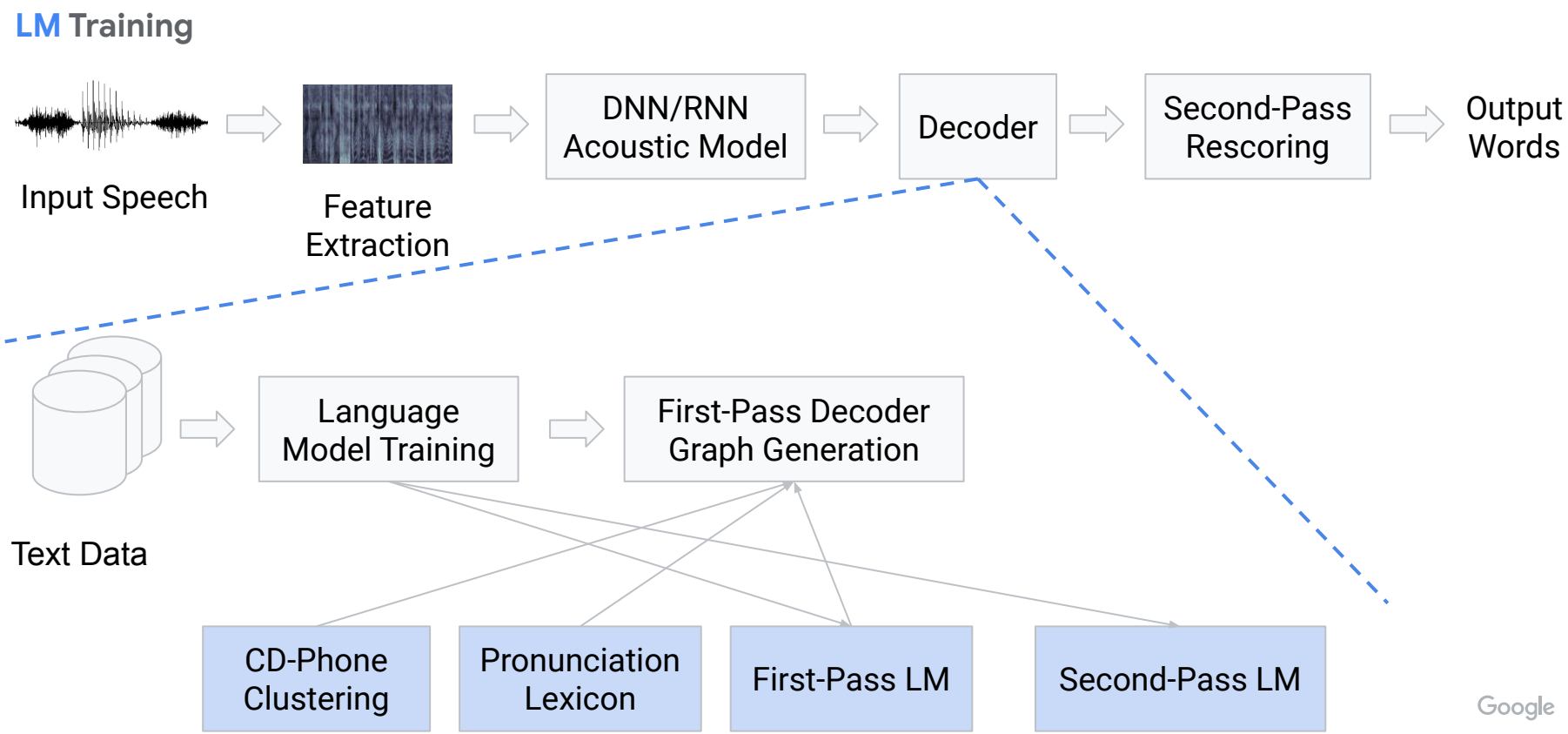
Output Words

Conventional ASR

AM Training



Conventional ASR



Conventional ASR

- Most ASR systems involve acoustic, pronunciation and language model components which are trained separately
  - Discriminative Sequence Training of AMs does couple these components
- Curating pronunciation lexicon, defining phoneme sets for the particular language requires expert knowledge, and is time-consuming

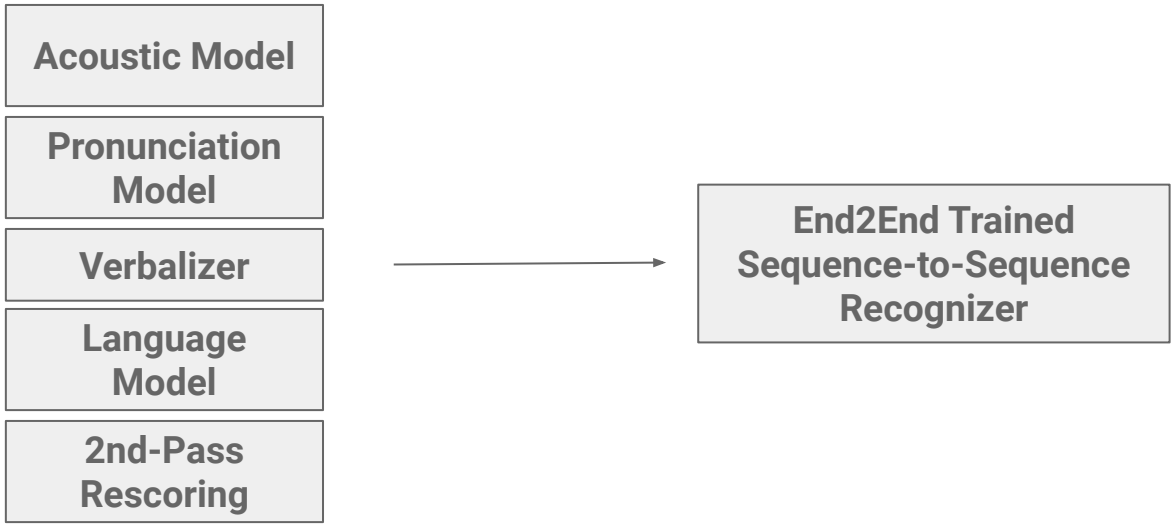
What is **End-to-End** ASR?

> A system which directly maps a sequence of input acoustic features into a sequence of graphemes or words.

> A system which is trained to optimize criteria that are related to the final evaluation metric that we are interested in (typically, word error rate).

Motivation

Typical Speech System



Key Takeaway

A single end-to-end trained sequence-to-sequence model, which directly outputs words or graphemes, could greatly simplify the speech recognition pipeline.

Agenda

Research developments on end-to-end models towards productionisation.

Attention	Pushing the limit of attention-based end-to-end models.
Online Models	Streaming models for real world applications.
Production	Interaction with other components in production.

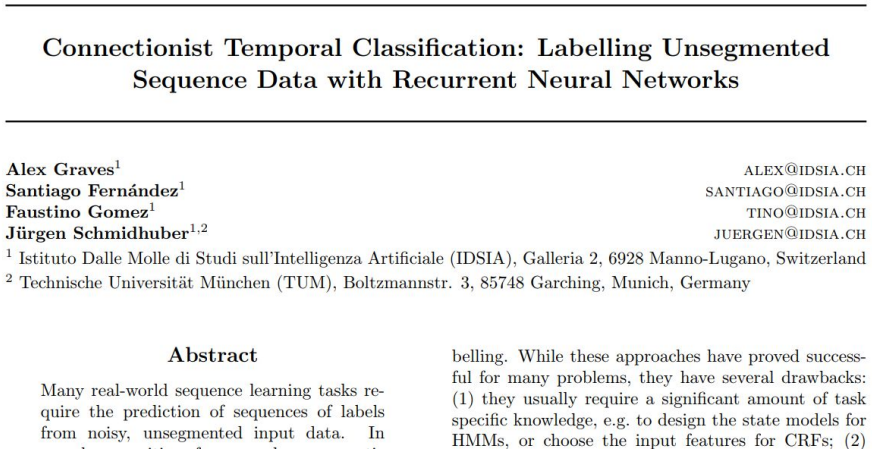
Historical Development of End-to-End ASR

CTC

Connectionist Temporal Classification

Connectionist Temporal Classification (CTC)

- CTC was proposed by [Graves et al., 2006] as a way to train an acoustic model without requiring frame-level alignments
- Early work, used CTC with phoneme output targets - not “end-to-end”
- CD-phoneme based CTC models achieve state-of-the-art performance for conventional ASR, but word-level lagged behind [Sak et al., 2015]



Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks

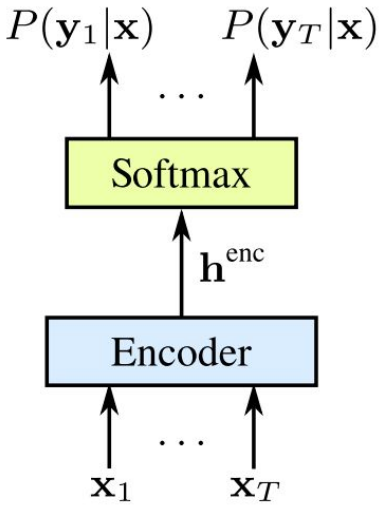
Alex Graves[1] ALEX@IDSIA.CH
Santiago Fernández[1] SANTIAGO@IDSIA.CH
Faustino Gomez[1] TINO@IDSIA.CH
Jürgen Schmidhuber[1,2] JUERGEN@IDSIA.CH
[1] Istituto Dalle Molle di Studi sull’Intelligenza Artificiale (IDSIA), Galleria 2, 6928 Manno-Lugano, Switzerland
[2] Technische Universität München (TUM), Boltzmannstr. 3, 85748 Garching, Munich, Germany

Abstract

Many real-world sequence learning tasks require the prediction of sequences of labels from noisy, unsegmented input data. In belling. While these approaches have proved successful for many problems, they have several drawbacks: (1) they usually require a significant amount of task specific knowledge, e.g. to design the state models for HMMs, or choose the input features for CRFs; (2)

[Graves et al., 2006] ICML

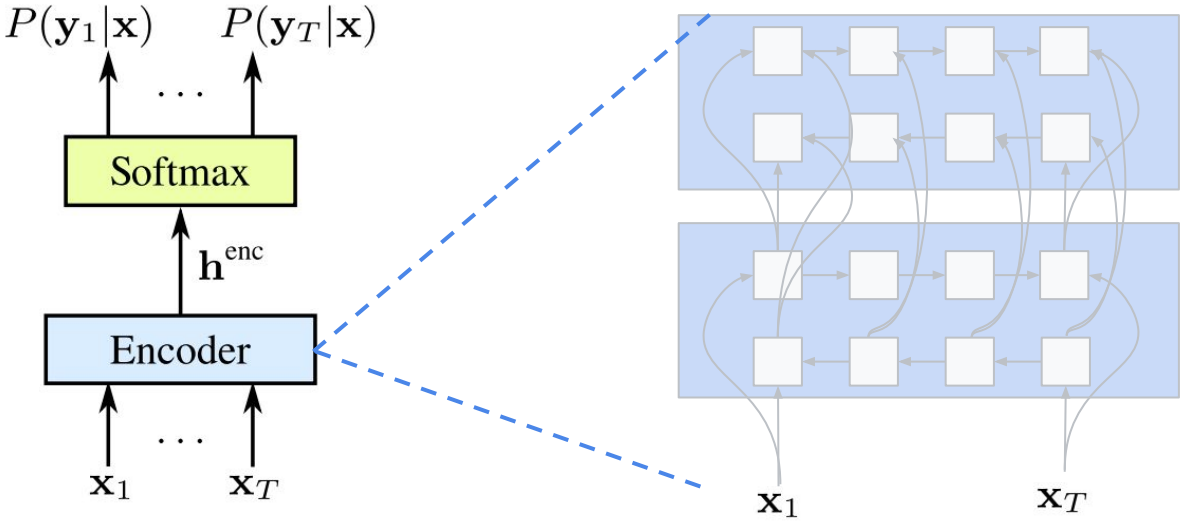
Connectionist Temporal Classification (CTC)



Key Takeaway

CTC allows for training an acoustic model without the need for frame-level alignments between the acoustics and the transcripts.

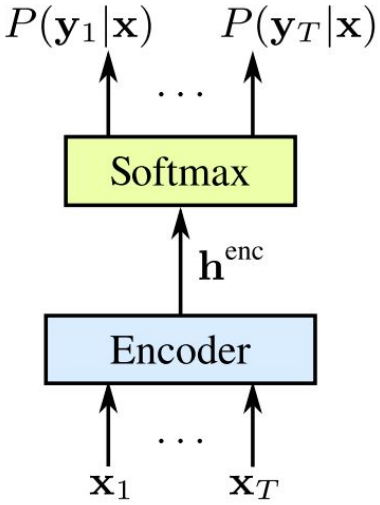
Connectionist Temporal Classification (CTC)



Key Takeaway

Encoder: Multiple layers of Uni- or Bi-directional RNNs (often LSTMs).

Connectionist Temporal Classification (CTC)



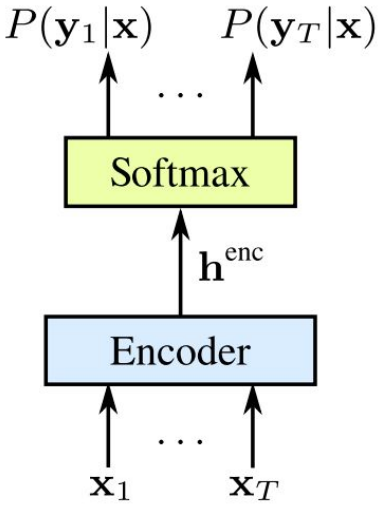
B	B	**c**	B	B	**a**	**a**	B	B	**t**
B	**c**	**c**	B	**a**	B	B	B	B	**t**
				...					
B	**c**	B	B	**a**	B	B	**t**	**t**	B

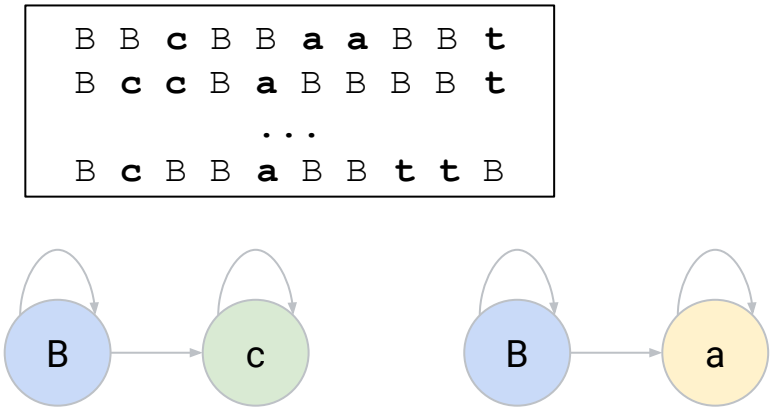
$$P(\mathbf{y}|\mathbf{x}) = \sum_{\hat{\mathbf{y}} \in \mathcal{B}(\mathbf{y},\mathbf{x})} \prod_{t=1}^{T} P(\hat{y}_t|\mathbf{x})$$

Key Takeaway

CTC introduces a special symbol - blank (denoted by B) - and maximizes the total probability of the label sequence by marginalizing over all possible alignments

Connectionist Temporal Classification (CTC)

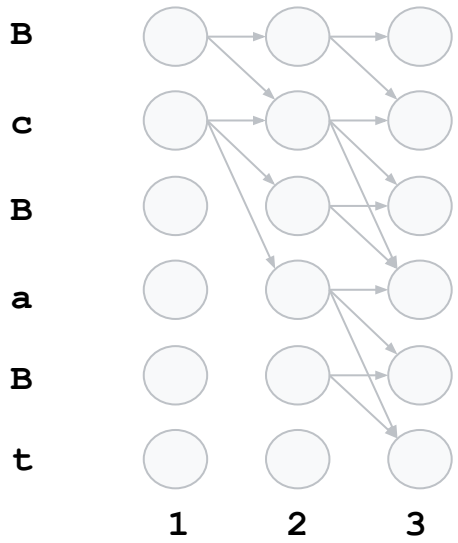




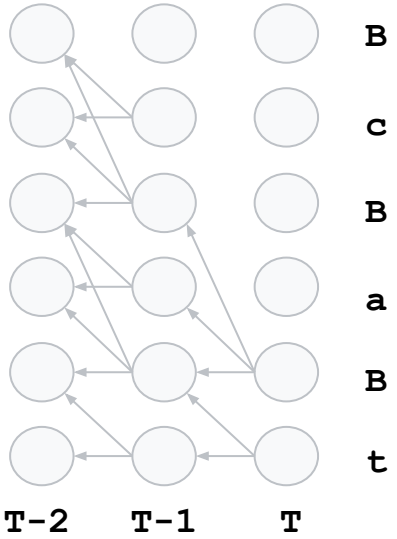
Key Takeaway

In a conventional hybrid system, this would correspond to defining the HMMs corresponding to each unit to consist of a shared initial state (blank), followed by a separate state(s) for the actual unit.

Connectionist Temporal Classification (CTC)



Forward-Backward Algorithm Computation



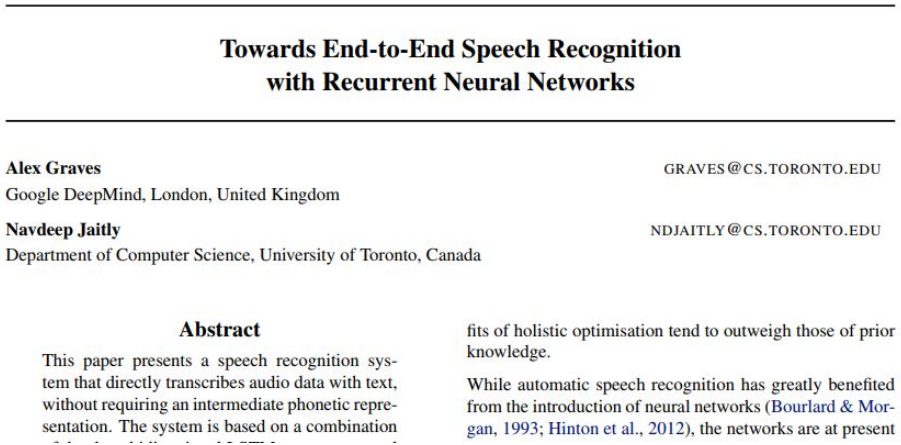
Frames, t

Key Takeaway

Computing the gradients of the loss requires the computation of the alpha-beta variables using the forward-backward algorithm [Rabiner, 1989]

CTC-Based End-to-End ASR

- Graves and Jaitly proposed a system with character-based CTC which directly output word sequences given input speech
- Using an external LM was important for getting good performance. Results reported by rescoring a baseline system.
- Also proposed minimizing expected transcription error [WSJ: 8.7% → 8.2%]



Towards End-to-End Speech Recognition with Recurrent Neural Networks

Alex Graves GRAVES@CS.TORONTO.EDU
Google DeepMind, London, United Kingdom

Navdeep Jaitly NDJAITLY@CS.TORONTO.EDU
Department of Computer Science, University of Toronto, Canada

Abstract

This paper presents a speech recognition system that directly transcribes audio data with text, without requiring an intermediate phonetic representation. The system is based on a combination

fits of holistic optimisation tend to outweigh those of prior knowledge.

While automatic speech recognition has greatly benefited from the introduction of neural networks (Bourlard & Morgan, 1993; Hinton et al., 2012), the networks are at present

[Graves and Jaitly, 2014] ICML

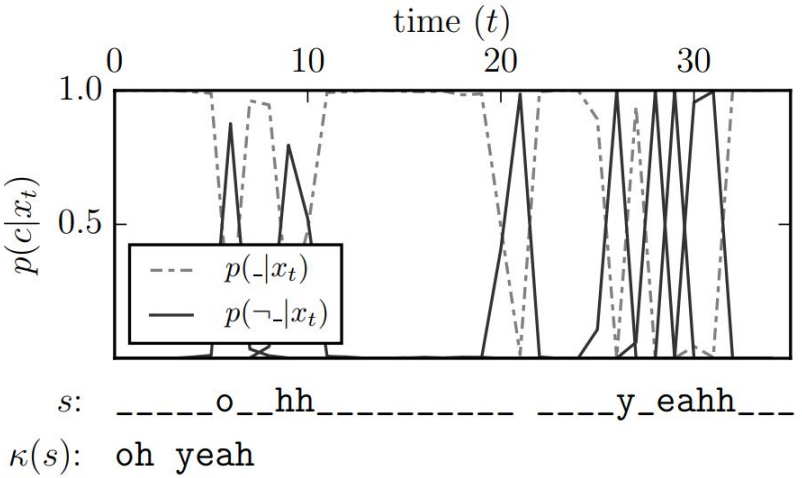
CTC-Based ASR

Refinements since [Graves & Jaitly, 2014]

- LM incorporated into first-pass decoding; easy integration with WFSTs
  - [Hannun et al., 2014] [Maas et al., 2015]: Direct first-pass decoding with an LM as opposed to rescoring as in [Graves & Jaitly, 2014]
  - [Miao et al., 2015]: EESEN framework for decoding with WFSTs, open source toolkit
- Large-scale GPU training; data augmentation; multiple languages
  - [Hannun et al., 2014; DeepSpeech] [Amodei et al., 2015; DeepSpeech2]: Large scale GPU training; Data Augmentation; Mandarin and English
- Using longer span units: words instead of characters
  - [Soltau et al., 2017]: Word-level CTC targets, trained on 125,000 hours of speech. Performance close to or better than a conventional system, even without using an LM!
  - [Audhkhasi et al., 2017]: Direct Acoustics-to-Word Models on Switchboard
- And many others ...

CTC-Based End-to-End ASR

CTC produces “spiky” and sparse activations - can sometimes directly read off the final transcription from the activations even without an LM



Reproduced from [Maas et al., 2015] NAACL

Shortcomings of CTC

- For efficiency, CTC makes an important independence assumption - network outputs at different frames are conditionally independent
- Obtaining good performance from CTC models requires the use of an external language model - direct greedy decoding does not perform very well

Attention-based Encoder-Decoder Models

Attention-based Encoder-Decoder Models

- Attention-based Encoder-Decoder Models emerged first in the context of neural machine translation.
- Were first applied to ASR by [Chan et al., 2015] [Chorowski et al., 2015]

Listen, Attend and Spell

William Chan
Carnegie Mellon University
williamchan@cmu.edu

Navdeep Jaitly, Quoc V. Le, Oriol Vinyals
Google Brain
{ndjaitly,qvl,vinyals}@google.com

[Chan et al., 2015]

Attention-Based Models for Speech Recognition

Jan Chorowski
University of Wrocław, Poland
jan.chorowski@ii.uni.wroc.pl

Dzmitry Bahdanau
Jacobs University Bremen, Germany

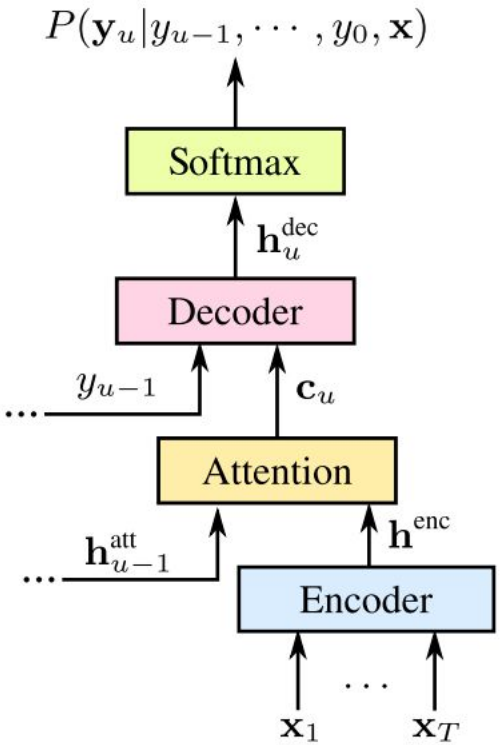
Dmitriy Serdyuk
Université de Montréal

Kyunghyun Cho
Université de Montréal

Yoshua Bengio
Université de Montréal
CIFAR Senior Fellow

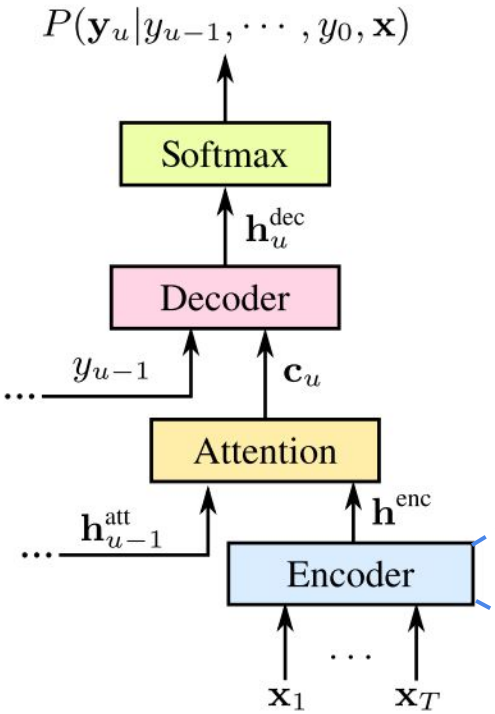
[Chorowski et al., 2015]

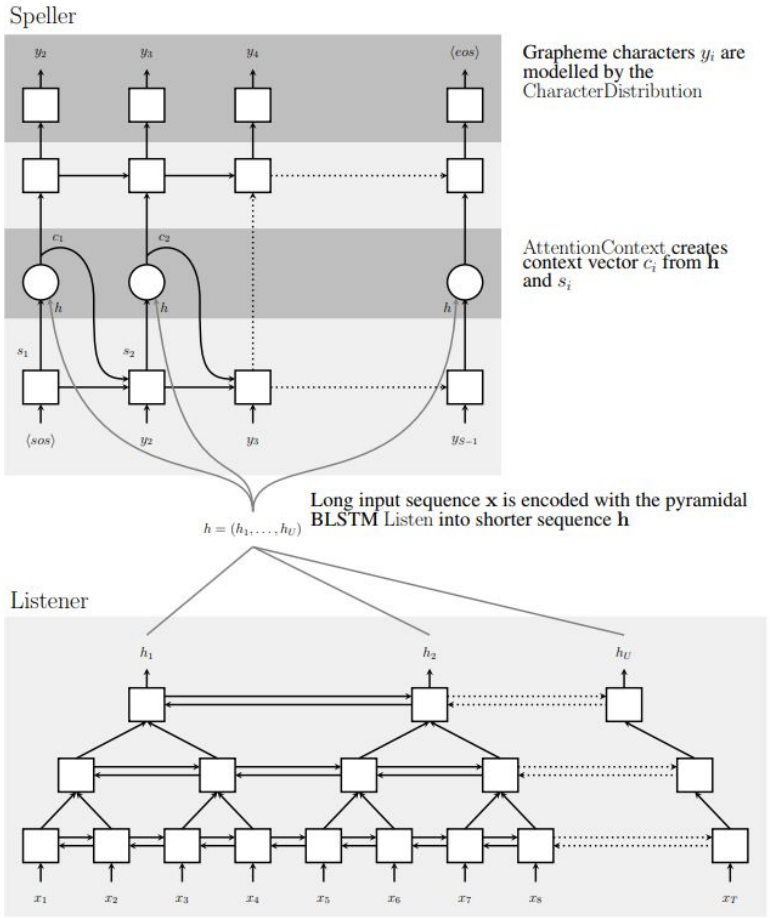
Attention-based Encoder-Decoder Models



- Encoder (analogous to AM):
  - Transforms input speech into higher-level representation
- Attention (alignment model):
  - Identifies encoded frames that are relevant to producing current output
- Decoder (analogous to PM, LM):
  - Operates autoregressively by predicting each output token as a function of the previous predictions

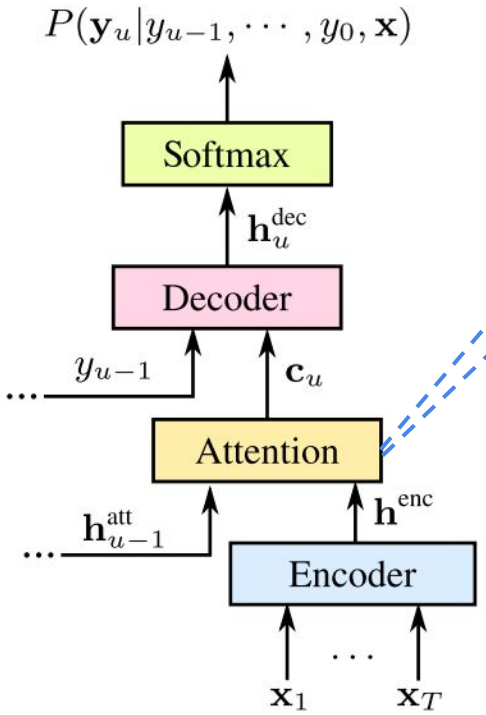
Attention-based Models

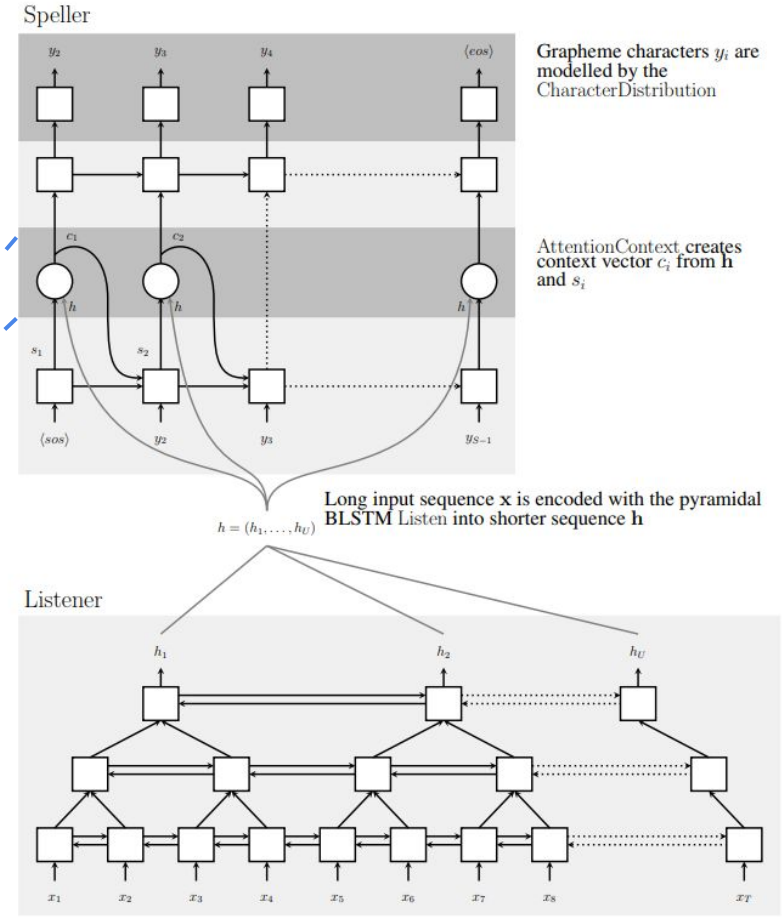




Reproduced from [Chan et al., 2015]

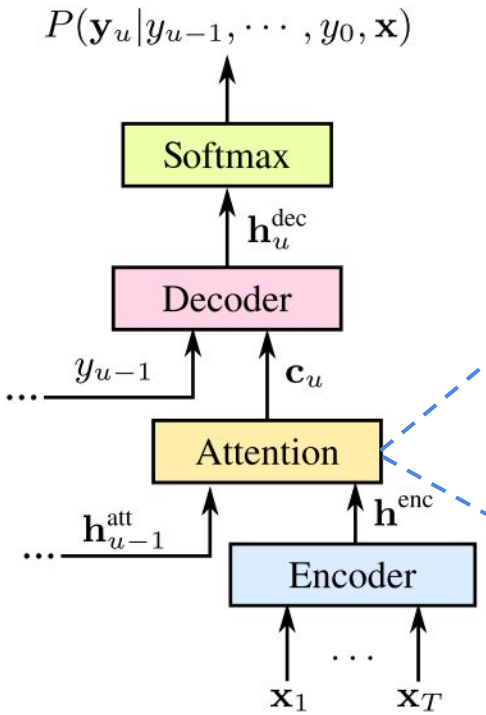
Attention-based Models





Reproduced from [Chan et al., 2015]

Attention-based Models



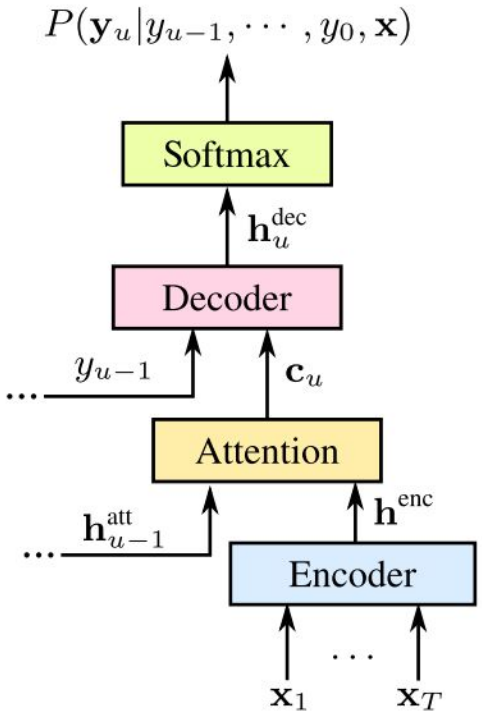
Attention module computes a similarity score between the decoder and each frame of the encoder

$$e_{u,t} = \text{score}(\mathbf{h}_{u-1}^{\text{att}}, \mathbf{h}_t^{\text{enc}})$$

$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=1}^{T} \exp(e_{u,t'})}$$

$$\mathbf{c}_u = \sum_{t=1}^{T} \alpha_{u,t} h_t^{\text{enc}}$$

Attention-based Models



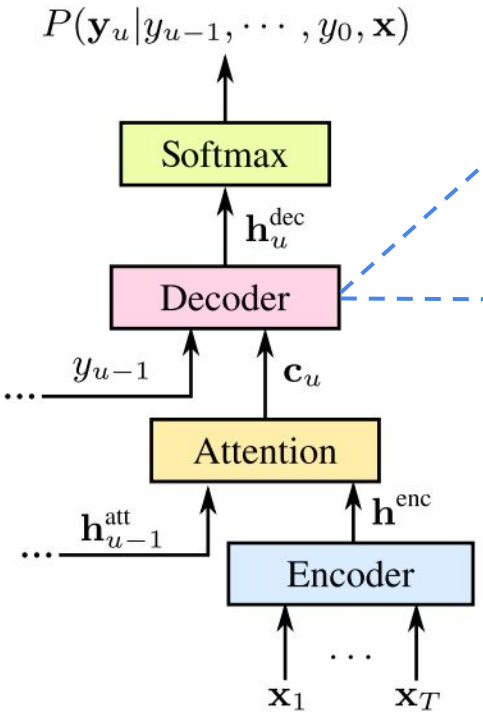
Dot-Product Attention [Chan et al., 2015]

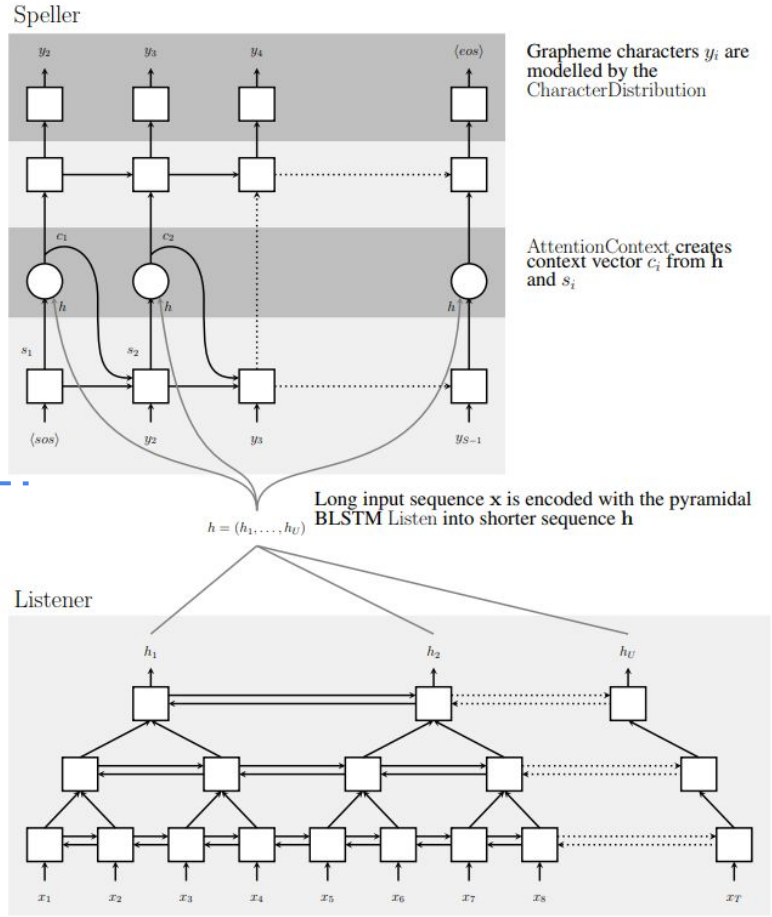
$$e_{u,t} = \left\langle \phi(W\mathbf{h}_{u-1}^{\text{att}}),\ \psi(V\mathbf{h}_t^{\text{enc}}) \right\rangle$$

Additive Attention [Chorowski et al., 2015]

$$e_{u,t} = w^T \tanh(W\mathbf{h}_{u-1}^{\text{att}} + V\mathbf{h}_t^{\text{enc}} + b)$$

Attention-based Models



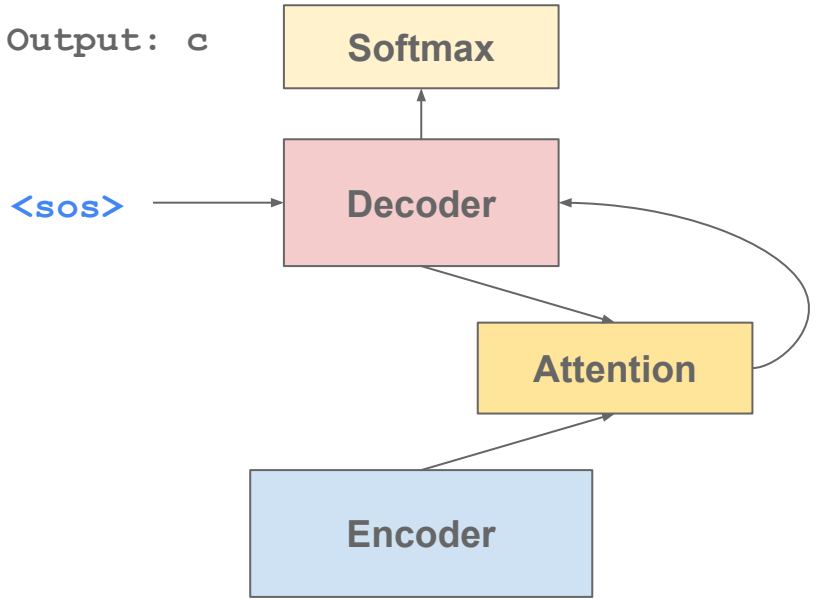


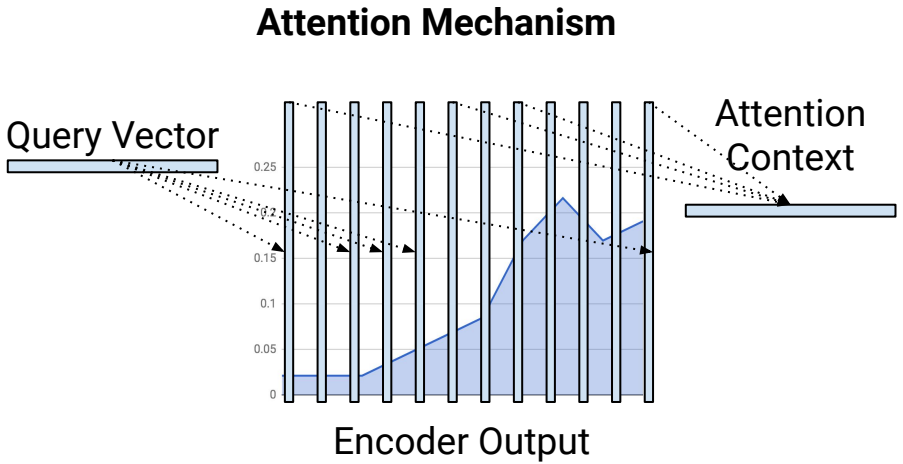
Reproduced from [Chan et al., 2015]

Attention-based Models

```
P(a|<sos>,x) = 0.01
P(b|<sos>,x) = 0.01
P(c|<sos>,x) = 0.92
       ...
```

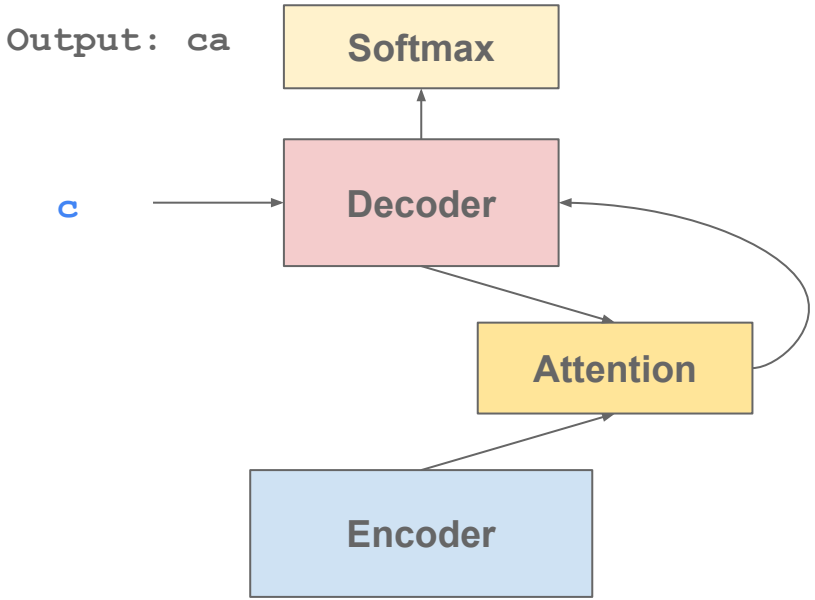
Attention mechanism summarizes encoder features relevant to predict next label





Attention-based Models

```
P(a|c,<sos>,x) = 0.95
P(b|c,<sos>,x) = 0.01
P(c|c,<sos>,x) = 0.01
          ...
```

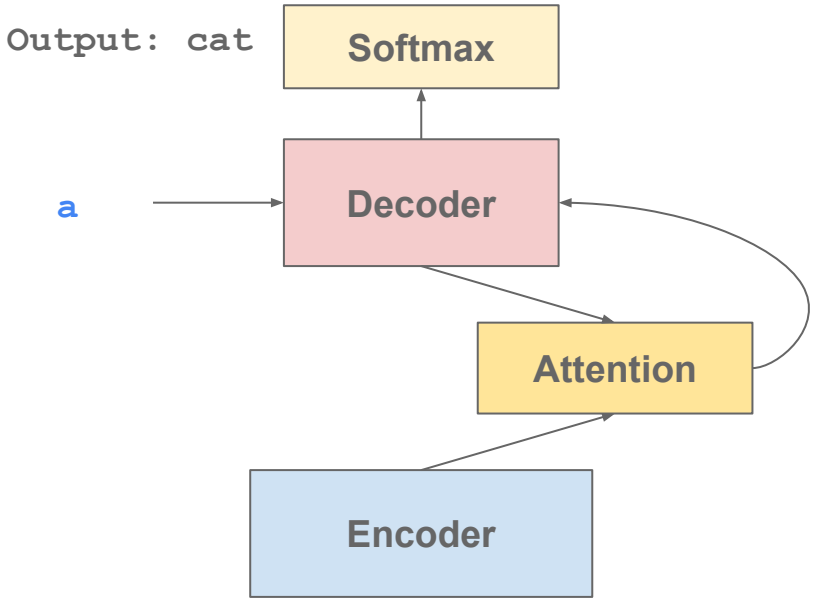


Labels from previous step are fed into decoder at the next step to predict

$$P(\mathbf{y}_u | y_{u-1}, \cdots, y_0, \mathbf{x})$$

Attention-based Models

```
P(a|a,c,<sos>,x) = 0.01
P(b|a,c,<sos>,x) = 0.08
              ...
P(t|a,c,<sos>,x) = 0.89
              ...
```

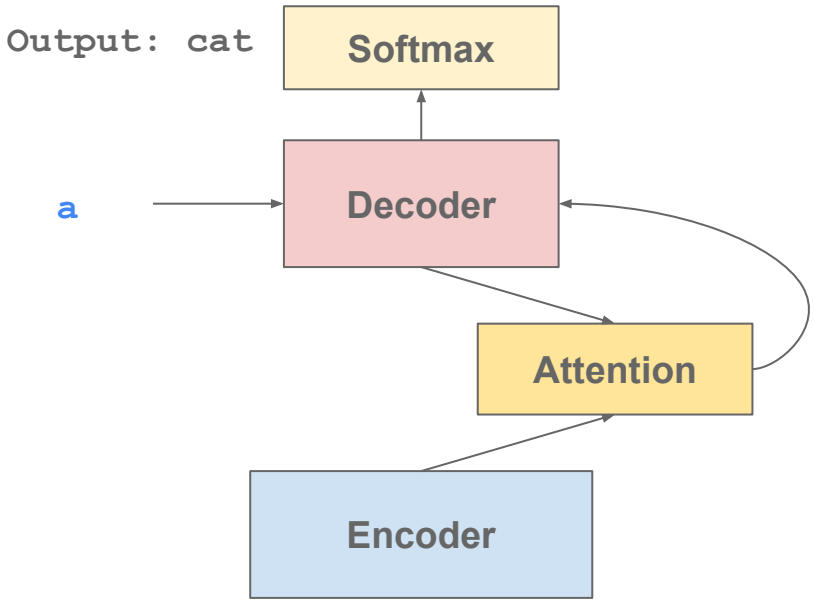


Labels from previous step are fed into decoder at the next step to predict

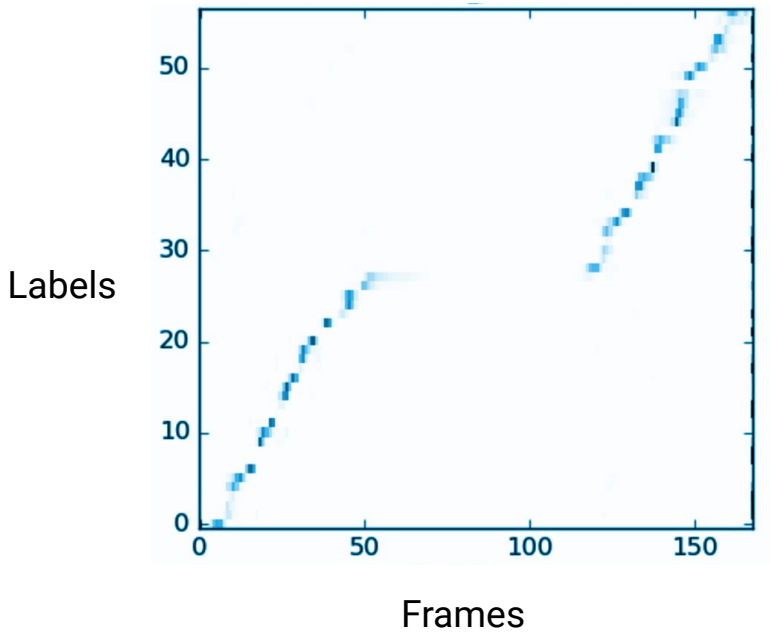
$$P(\mathbf{y}_u | y_{u-1}, \cdots, y_0, \mathbf{x})$$

Attention-based Models

```
P(a|t,a,c,<sos>,x) = 0.01
P(b|t,a,c,<sos>,x) = 0.01
                ...
P(<eos>|t,a,c,<sos>,x) = 0.96
                ...
```



Process terminates when the model predicts <eos> which denotes end of sentence.



Online Models

RNN-T, NT, MoChA

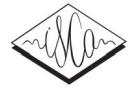
To be discussed in the 2nd part of the tutorial

Model Comparisons on a 12,500 hour Google Task

Comparing Various End-to-End Approaches

- Compare various sequence-to-sequence models head-to-head, trained on same data, to understand how these approaches compare to each other
- Evaluated on a large-scale 12,500 hour Google Voice Search Task

INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden



A Comparison of Sequence-to-Sequence Models for Speech Recognition

Rohit Prabhavalkar[1], *Kanishka Rao*[1], *Tara N. Sainath*[1], *Bo Li*[1], *Leif Johnson*[1], *Navdeep Jaitly*[2†]

[1]Google Inc., U.S.A
[2]NVIDIA, U.S.A.

{prabhavalkar,kanishkarao,tsainath,boboli,leif}@google.com, njaitly@nvidia.com

Abstract

In this work, we conduct a detailed evaluation of various all-neural, end-to-end trained, sequence-to-sequence models applied to the task of speech recognition. Notably, each of these

was shown to outperform a state-of-the-art CD-phoneme baseline on a YouTube video captioning task. The basic CTC model was extended by Graves [3] to include a separate recurrent language model component, in a model referred to as the recurrent neural network (RNN) transducer. Although this model has

[Prabhavalkar et al., 2017]

Experimental Setup

Model Configurations

- Baseline
  - State-of-the-art CD-Phoneme model: 5x700 BLSTM; ~8000 CD-Phonemes
  - CTC-training followed by sMBR discriminative sequence training
  - Decoded with large 5-gram LM in first pass
  - Second pass rescoring with much larger 5-gram LM
  - Lexicon of millions of words of expert curated pronunciations
- Sequence-to-Sequence Models
  - Trained to output graphemes: [a-z], [0-9], <space>, and punctuation
  - Models are evaluated using beam search (Keep Top 15 Hyps at Each Step)
  - ***Models are not decoded or rescored with an external language model, or a pronunciation model***

Experimental Setup

Data

- Training Set
  - ~15M Utterances (~12,500 hrs) of anonymized utterances from Google Voice Search Traffic
  - Multi-style Training: Artificially distorted using room simulator by adding noise samples extracted from YouTube videos and environmental recordings of daily events
- Evaluation Sets
  - Dictation: ~13K utterances (~124K words) open-ended dictation
  - VoiceSearch: ~12.9K utterances (~63K words) of voice-search queries

Results

Model	Clean	
	Dictation	VoiceSearch
Baseline Uni. Context Dependent Phones (CDP)	6.4	9.9
Baseline BiDi. CDP	**5.4**	**8.6**
CTC-grapheme	39.4	53.4

Key Takeaway

Decoding CTC-grapheme models without an LM performs poorly.

Results

Model	Clean	
	Dictation	VoiceSearch
Baseline Uni. CDP	6.4	9.9
Baseline BiDi. CDP	**5.4**	**8.6**
CTC-grapheme	39.4	53.4
Attention-based Model	6.6	11.7

Key Takeaway

Attention-based model performs the best, but cannot be used for streaming applicaitons.

End-to-End Comparisons [Battenberg et al., 2017]

	Architecture	SWBD WER	CH WER
Published	Iterated-CTC [29]	11.3	18.7
	BLSTM + LF MMI [21]	8.5	15.3
	LACE + LF MMI [4] [28]	8.3	14.8
	Dilated convolutions [25]	7.7	14.5
	CTC + Gram-CTC [17]	7.3	14.7
	BLSTM + Feature fusion[23]	**7.2**	**12.7**
Ours	CTC [17]	9.0	17.7
	RNN-Transducer		
	Beam Search **NO LM**	8.5	16.4
	Beam Search + LM	8.1	17.5
	Attention		
	Beam Search **NO LM**	8.6	17.8
	Beam Search + LM	8.6	17.8

Model	Dev	Test
CTC [4]		
Greedy decoding	23.03	-
Beam search + LM (beam=2000)	15.9	16.44
RNN-Transducer		
Greedy decoding	18.99	-
Beam search (beam=32)	17.41	-
+ LM rescoring	15.6	16.50
Attention		
Greedy decoding	22.67	-
Beam search (beam=256)	18.71	-
+ Length-norm weight	19.5	-
+ Coverage cost	18.9	-
+ LM rescoring	16.0	16.48

Key Takeaway

Similar conclusions were reported by [Battenberg et al., 2017] on Switchboard. RNN-T without an LM is consistently better than CTC with an LM.

Combining Approaches

- Various end-to-end approaches can be successfully combined to improve the overall system
- CTC and Attention-based models can be combined in a multi-task learning framework [Kim et al., 2017]
- RNN-T can be augmented with an attention module which can
  - condition the language model component on the acoustics [Prabhavalkar et al., 2017] or,
  - be used to bias the decoder towards particular items of interest [He et al., 2017]
- An attention model can be augmented with a secondary attention module which can bias towards an arbitrary number of phrases of interest [Pundak et al., 2018] (will be discussed in more detail in a few slides)

Further Improvements

Improvements for productions

To match an end-to-end model to a strong conventional system:

- Structural improvements
  - Wordpiece models
  - Multi-headed attention
- Optimization improvements
  - Minimum word error rate (MWER) training
  - Scheduled sampling
  - Asynchronous and synchronous training
  - Label smoothing
- External language model integration

Structure improvements

Wordpiece Model

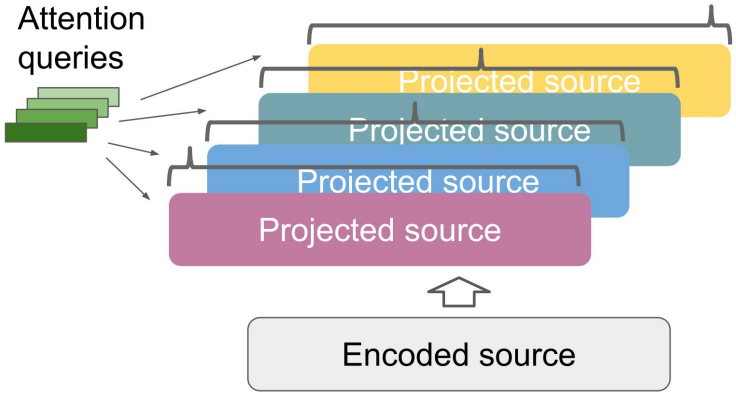
- Instead of the commonly used grapheme, we can use longer units such as wordpieces
- Motivations:
  - Typically, word-level LMs have a much lower perplexity compared to grapheme-level LMs [Kannan et al., 2018]
  - Modeling wordpiece allows for a much stronger decoder LM
  - Modeling longer units improves the effective memory of the decoder LSTMs
  - Allows the model to potentially memorize pronunciations for frequently occurring words
  - longer units require fewer decoding steps; this speeds up inference in these models significantly
- good performance for LAS and RNN-T [Rao et al., 2017].

Wordpiece Model

- sub-word units, ranging from graphemes all the way up to entire words.
- there are no out-of-vocabulary words with word piece models
- The word piece models are trained to maximize the language model likelihood over the training set
- the word pieces are “position-dependent”, in that a special word separator marker is used to denote word boundaries.
- Words are segmented deterministically and independent of context, using a greedy algorithm.

Multi-headed Attention

- Multi-head attention (MHA) was first explored in [Vaswani et al., 2017] for machine translation
- MHA extends the conventional attention mechanism to have multiple heads, where each head can generate a different attention distribution.



Multi-headed Attention

- Results

Exp-ID	Model	WER	WERR
E1	Grapheme	9.2	-
E2	WPM	9.0	2.2%
E3	+ MHA	8.0	11.1%

Optimization improvements

Minimum Word Error Rate (MWER)

- Attention-based Sequence-to-Sequence models are typically trained by optimizing cross entropy loss (i.e., maximizing log-likelihood of the training data)

$$\mathcal{L}_{\text{CE}} = \sum_{(\mathbf{x},\mathbf{y}^*)} \sum_{u=1}^{L+1} -\log P(y_u^* | y_{u-1}^*, \cdots, y_0^* = \langle \text{sos} \rangle, \mathbf{x})$$

- Training criterion does not match metric of interest: Word Error Rate
- Goal: Optimize a loss that minimizes or is correlated with minimizing word error rate

Minimum Word Error Rate (MWER)

- In the context of conventional ASR system, for Neural Network Acoustic Models
  - State-level Minimum Bayes Risk (sMBR) [Kingsbury, 2009]
  - Word-level edit-based Minimum Bayes Risk (EMBR) [Shannon, 2017]
- In the context of end-to-end models
  - Connectionist Temporal Classification (CTC) [Graves and Jaitly, 2014]
  - Recurrent Neural Aligner (RNA) [Sak et al., 2017]: Applies word-level EMBR to RNA
  - Machine Translation:
    - REINFORCE [Ranzato et al., 2016]
    - Beam Search Optimization [Wiseman and Rush, 2016]
    - Actor-Critic [Bahdanau et al., 2017]

Minimum Word Error Rate (MWER)

$$\mathcal{L}_{\text{werr}}(\mathbf{x}, \mathbf{y}^*) = \mathbb{E}[\mathcal{W}(\mathbf{y}, \mathbf{y}^*)] = \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})\mathcal{W}(\mathbf{y}, \mathbf{y}^*)$$

Number of Word Errors

Key Takeaway

Minimizing expected WER directly is intractable since it involves a summation over all possible label sequences. Approximate expectation using samples.

Minimum Word Error Rate (MWER)

- Approximate expectation using samples [Shannon, 2017].

$$\mathcal{L}_{\text{werr}}(\mathbf{x}, \mathbf{y}^*) \approx \mathcal{L}_{\text{werr}}^{\text{Sample}}(\mathbf{x}, \mathbf{y}^*) = \frac{1}{N} \sum_{\mathbf{y}_i \sim P(\mathbf{y}|\mathbf{x})} \mathcal{W}(\mathbf{y}_i, \mathbf{y}^*)$$

- Approximation using N-Best List [Stolcke et al., 1997][Povey, 2003]

$$\mathcal{L}_{\text{werr}}^{\text{N-best}}(\mathbf{x}, \mathbf{y}^*) = \sum_{\mathbf{y}_i \in \text{Beam}(\mathbf{x}, N)} \widehat{P}(\mathbf{y}_i|\mathbf{x}) \left[\mathcal{W}(\mathbf{y}_i, \mathbf{y}^*) - \widehat{W} \right]$$

Minimum Word Error Rate (MWER)

- Results on Google's Voice Search

Model	Uni-Directional Encoder	Bi-Directional Encoder
Baseline	8.1	7.2
+MWER Training	**7.5 (7.4%)**	**6.9 (4.2%)**

Minimum Word Error Rate (MWER)

- Since [Prabhavalkar et al., 2018] we have repeated the experiments with MWER training on a number of models including RNN-T [Graves et al., 2013] and other streaming attention-based models such as MoChA [Chiu and Raffel, 2017] and the Neural Transducer [Jaitly et al., 2016]
- In all cases we have observed between 8% to 20% relative WER reduction
- Implementing MWER requires the ability to decode N-best hypotheses from the model which can be somewhat computationally expensive

Scheduled Sampling

- Feeding the ground-truth label as the previous prediction (so-called teacher forcing) helps the decoder to learn quickly at the beginning, but introduces a mismatch between training and inference.
- The scheduled sampling process, on the other hand, samples from the probability distribution of the previous prediction (i.e., from the softmax output) and then uses the resulting token to feed as the previous token when predicting the next label
- This process helps reduce the gap between training and inference behavior. Our training process uses teacher forcing at the beginning of training steps, and as training proceeds, we linearly ramp up the probability of sampling from the model's prediction to 0.4 at the specified step, which we then keep constant until the end of training

Asynchronous and Synchronous Training

- synchronous training can potentially provide faster convergence rates and better model quality, but also requires more effort in order to stabilize network training.
- Both approaches have a high gradient variance at the beginning of the training when using multiple replicas
    - In asynchronous training we use replica ramp up: that is, the system will not start all training replicas at once, but instead start them gradually
    - In synchronous training we use two techniques: learning rate ramp up and a gradient norm tracker

Label Smoothing

- a regularization mechanism to prevent the model from making over-confident predictions.
- It encourages the model to have higher entropy at its prediction, and therefore makes the model more adaptable
- We followed the same design as [Szegedy et al., 2016] by smoothing the ground-truth label distribution with a uniform distribution over all labels.

Optimization improvements

- Results

Exp-ID	Model	WER	WERR
E2	WPM	9.0	-
E3	+ MHA	8.0	11.1%
E4	+ Sync	7.7	3.8%
E5	+ SS	7.1	7.8%
E6	+ LS	6.7	5.6%
E7	+ MWER	5.8	13.4%

External Language Model

Motivation

Reference	LAS model output
What language is built into electrical circuitry of a computer?	what language is built into electrical circuit tree of a computer
Leona Lewis believe	vienna lewis believe
Suns-Timberwolves score	sun's timberwolves score

Key Takeaway

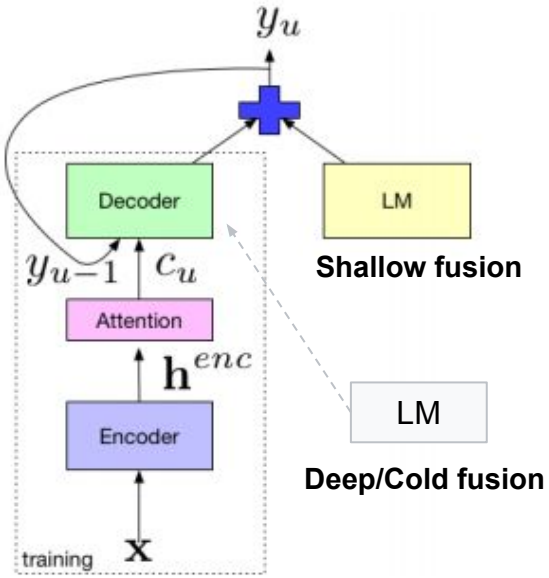
Some Voice Search errors appear to be fixable with a good language model trained on more text-only data.

Motivation

- The LAS model requires audio-text pairs: we have only 15M of these
- Our production LM is trained on billions of words of text-only data
- How can we look at incorporating a larger LM into our LAS model?
- More details can be found in [Kannan et al., 2018]

Extending LAS with an LM

- Listen, Attend and Spell [Chan et al., 2015]
- How to incorporate an LM?
  - **Shallow fusion** [Kannan et al., 2018]
    - LM is applied on output
  - **Deep fusion** [Gulcehre et al., 2015]
    - Assumes LM is fixed
  - **Cold fusion** [Sriram et al., 2018]
    - Simple interface between a deep lm and the encoder
    - Allows to swap in task-specific LMs
- In these experiments, fusion is used during the beam search rather than n-best rescoring.



Shallow Fusion

- Log-linear interpolation between language model and seq2seq model:

$$\mathbf{y}^* = \arg\max_y \log p(y|x) + \lambda \log p_{LM}(y)$$

- Typically only performed at inference time
- Language model is trained ahead of time and fixed
- LM can be either n-gram (FST) or RNN.
- Analogous to 1st pass rescoring.
- [Chorowski and Jaitly, 2017]. [Kannan et al., 2018].

Comparison of Fusion Results

- Shallow Fusion still seems to perform the best
- Full comparison in [Toshniwal, 2018]

System	Voice Search	Dictation
Baseline LAS	5.6	4.0
Shallow Fusion	**5.3**	**3.7**
Deep Fusion	5.5	4.1
Cold Fusion	**5.3**	3.9

State-of-the-art **Performance**

Google's Voice Search Task

Exp-ID	Model	VS/D	1st pass Model Size
E8	Proposed	**5.6/4.1**	**0.4 GB**
E9	Conventional LFR system	6.7/5.0	0.1 GB (AM) + 2.2 GB (PM) + 4.9 GB (LM) = 7.2GB

Key Takeaway

LAS achieves better performance than convention models.

Librispeech

	Dev	Test	WERR
Char baseline	6.57	6.61	-
+ ST	5.72	5.75	13.0%
WP + ST	5.29	5.32	7.5%
+ LA-attention	4.70	4.83	9.2%
+ LS	4.37	4.62	4.3%
+ SS	4.31	4.51	2.2%
+ DT	4.29	4.40	2.4%
+ DA	4.21	4.34	1.4%

Table 3: *WERs (%) for the **960hr** dataset.*

Unit	LM	dev		test	
		clean	other	clean	other
Phoneme	3-gram	5.6	15.8	6.2	15.8
Grapheme	None	5.3	15.2	5.5	15.3
Word-Piece 16K	None	4.4	13.2	4.7	13.4
Word-Piece 16K	LSTM	3.4	10.3	3.7	10.6
BPE 10K (Zeyer et al. [22])	None	4.9	14.4	4.9	15.4
	LSTM	3.5	11.5	3.8	12.8
Hybrid system (Han et al. [23])	N-gram	3.4	8.8	3.6	8.9
	LSTM	3.1	8.3	3.5	8.6

Key Takeaway

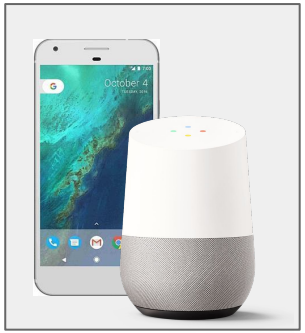
Similar observations have been reported by other groups.

Yan et.al; Attention-based sequence-to-sequence model for speech recognition: development of state-of-the-art system on LibriSpeech and its application to non-native English
Kazuki et.al; MODEL UNIT EXPLORATION FOR SEQUENCE-TO-SEQUENCE SPEECH RECOGNITION

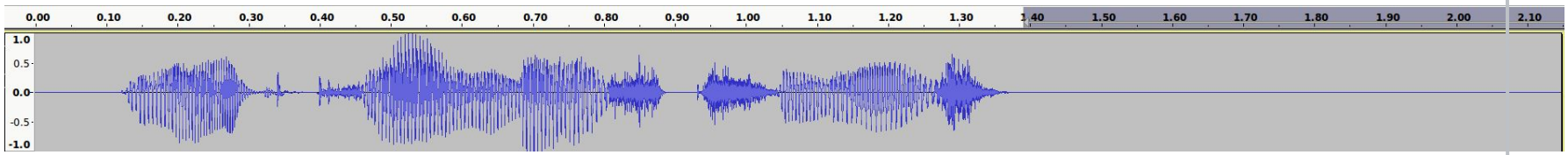
Online Models

RNN-T, NT, MoChA

Streaming Speech Recognition

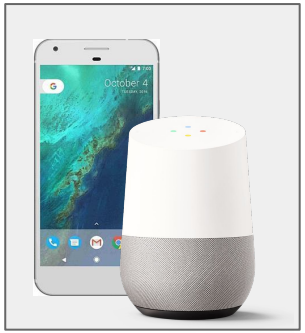


Finalize recognition &
Taking action / fetching the search results



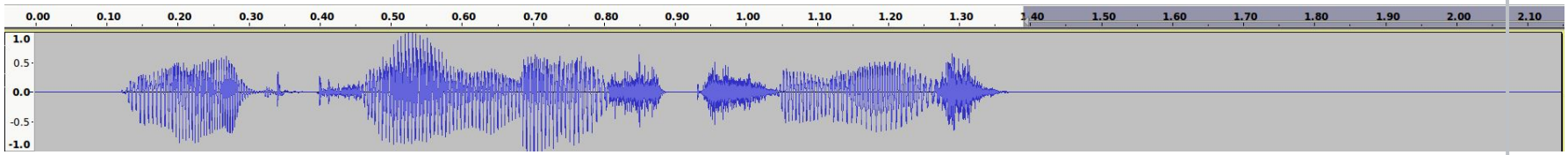
Recognize the audio

Streaming Speech Recognition



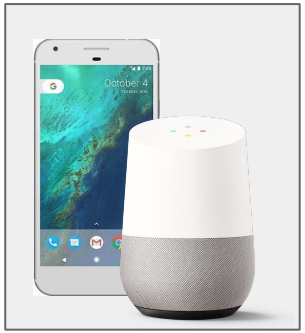
Finalize recognition &
Taking action / fetching the search results

Endpoint quickly



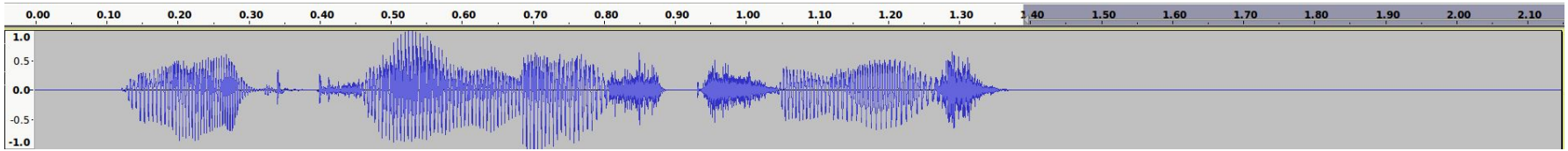
Recognize the audio

Streaming Speech Recognition



Important implications for ***embedded*** applications:

- Reliability
- Latency
- Privacy



Recognize the audio

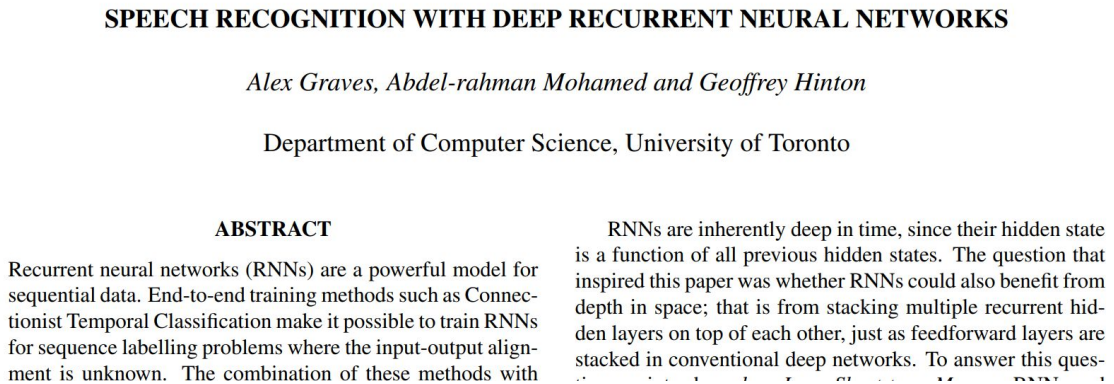
Online Models

- LAS is not streaming
- We will show a thorough comparison of different online models
  - RNN-T [Graves, 2012], [Rao et al., 2017], [He et al., 2018]
  - Neural Transducer [Jaitly et al., 2015], [Sainath et al., 2018]
  - MoChA [Chiu and Raffel, 2018]

Recurrent Neural Network Transducer

Recurrent Neural Network Transducer (RNN-T)

- Proposed by Graves et al., RNN-T augments a CTC-based model with a recurrent LM component
- Both components are trained jointly on the available acoustic data
- As with CTC, the method does not require aligned training data.



SPEECH RECOGNITION WITH DEEP RECURRENT NEURAL NETWORKS

Alex Graves, Abdel-rahman Mohamed and Geoffrey Hinton

Department of Computer Science, University of Toronto

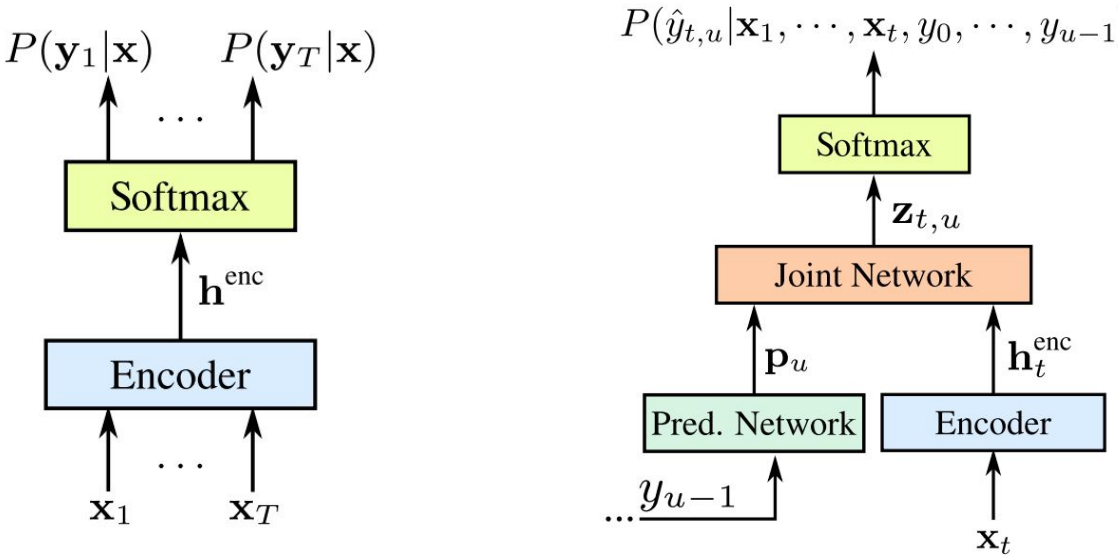
ABSTRACT

Recurrent neural networks (RNNs) are a powerful model for sequential data. End-to-end training methods such as Connectionist Temporal Classification make it possible to train RNNs for sequence labelling problems where the input-output alignment is unknown. The combination of these methods with

RNNs are inherently deep in time, since their hidden state is a function of all previous hidden states. The question that inspired this paper was whether RNNs could also benefit from depth in space; that is from stacking multiple recurrent hidden layers on top of each other, just as feedforward layers are stacked in conventional deep networks. To answer this ques-

[Graves et al., 2013] ICASSP;
[Graves, 2012] ICML Representation Learning Workshop

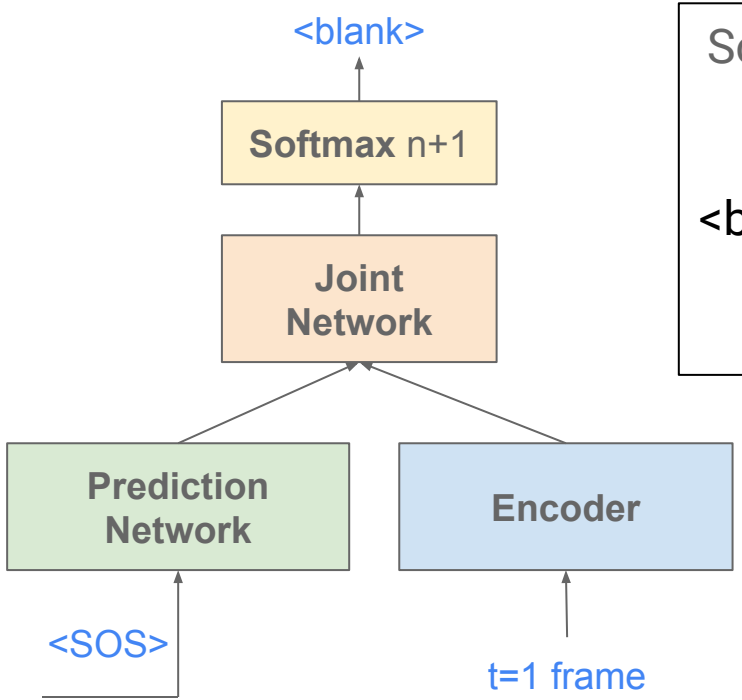
Recurrent Neural Network Transducer (RNN-T)



RNN-T [Graves, 2012] augments CTC encoder with a recurrent neural network LM

Recurrent Neural Network Transducer (RNN-T)

Output:

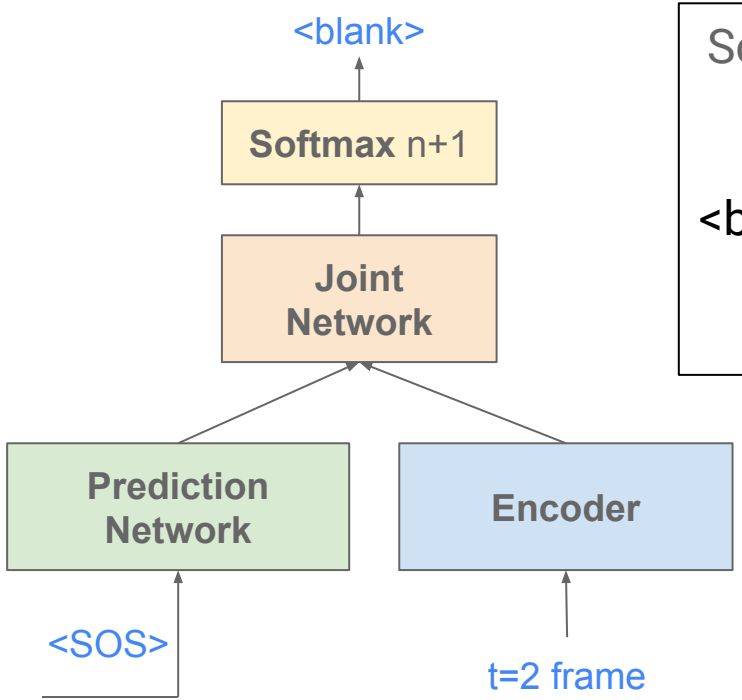


Softmax over n + 1 labels, includes a blank like CTC.

<blank> → advance in Encoder, retain prediction network state

Recurrent Neural Network Transducer (RNN-T)

Output:

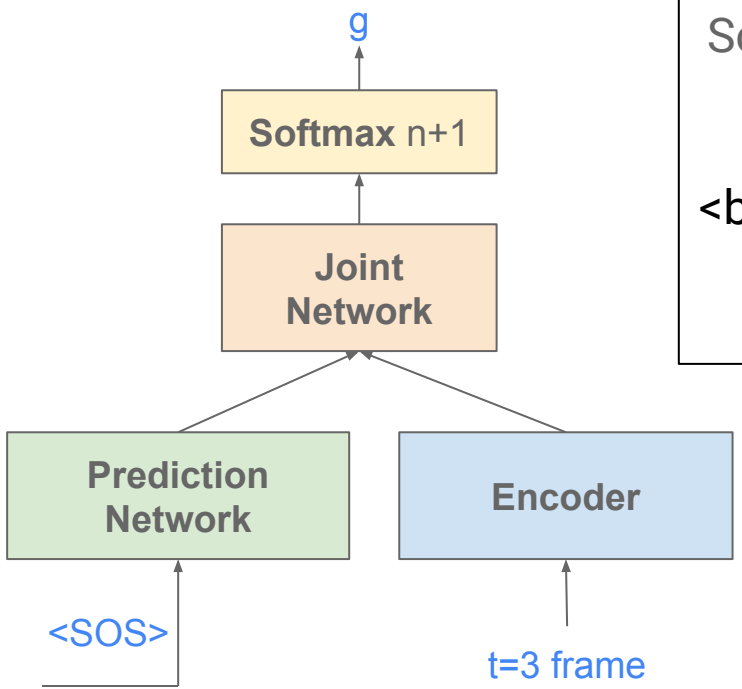


Softmax over n + 1 labels, includes a blank like CTC.

<blank> → advance in Encoder, retain prediction network state

Recurrent Neural Network Transducer (RNN-T)

Output: g

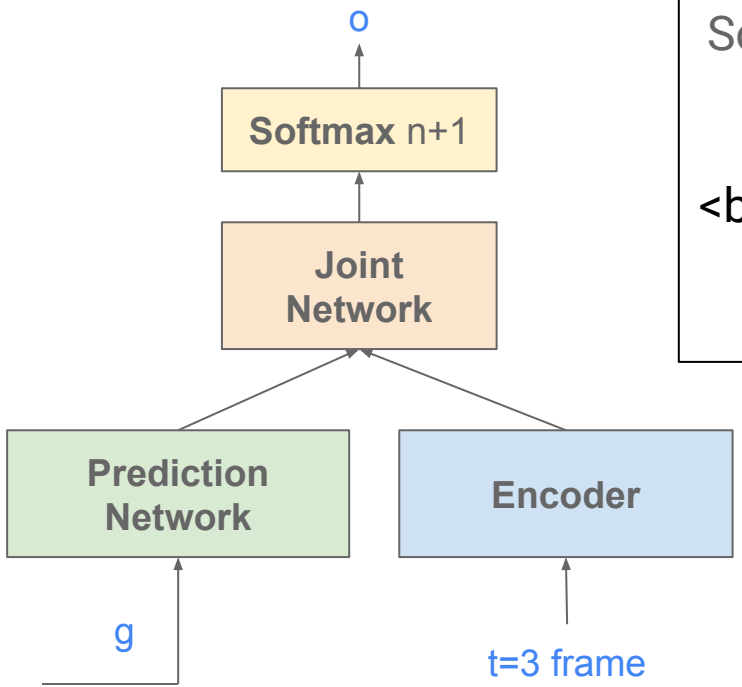


Softmax over n + 1 labels, includes a blank like CTC.

<blank> → advance in Encoder, retain prediction network state

Recurrent Neural Network Transducer (RNN-T)

Output: go

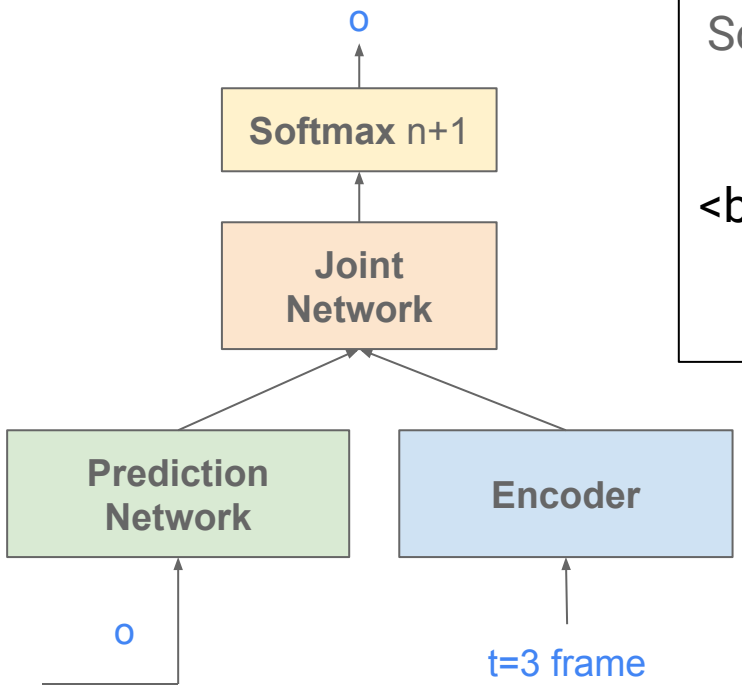


Softmax over n + 1 labels, includes a blank like CTC.

<blank> → advance in Encoder, retain prediction network state

Recurrent Neural Network Transducer (RNN-T)

Output: goo

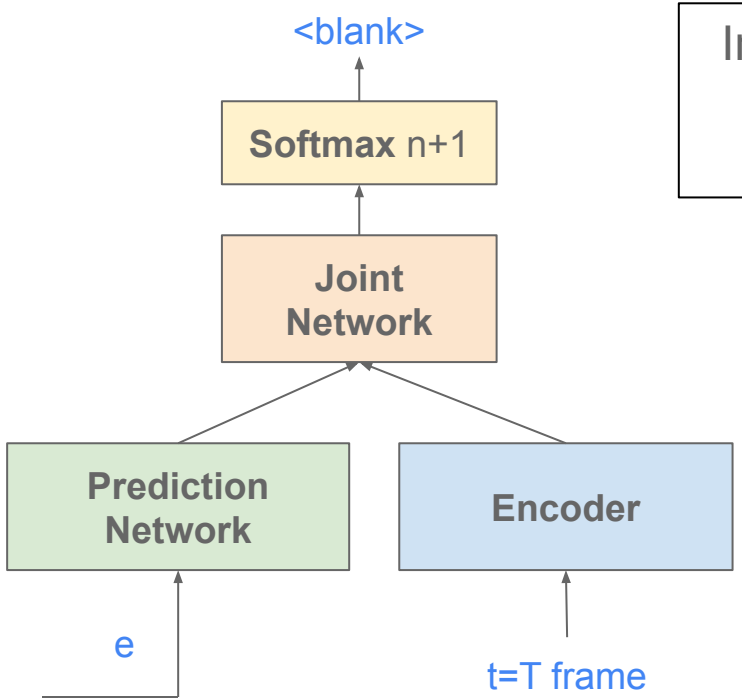


Softmax over n + 1 labels, includes a blank like CTC.

<blank> → advance in Encoder, retain prediction network state

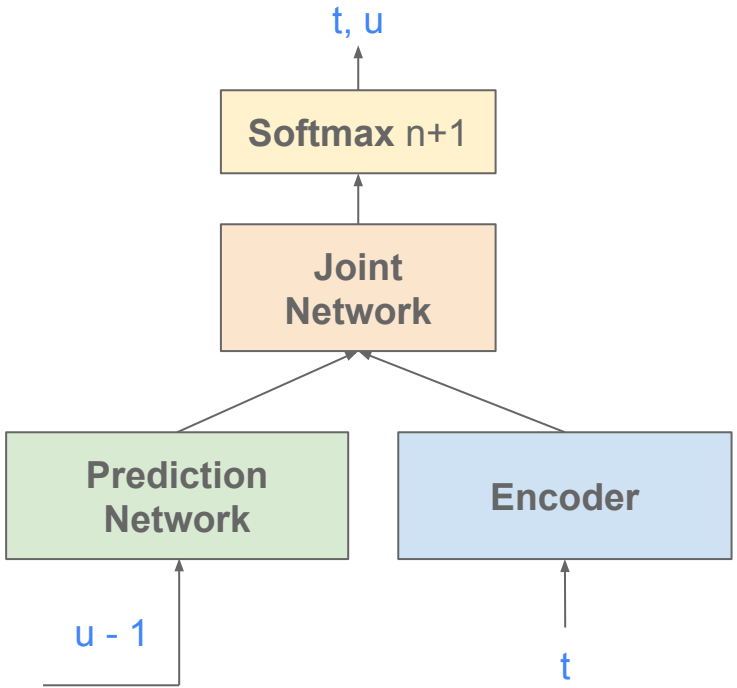
Recurrent Neural Network Transducer (RNN-T)

Output: google

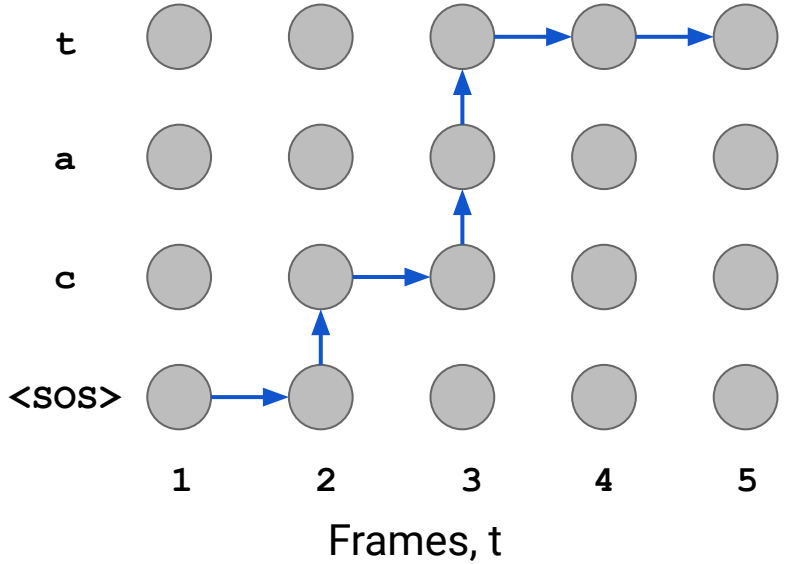


Inference terminates when all input frames have been consumed

Recurrent Neural Network Transducer (RNN-T)



During training feed the true label sequence to the LM.
Given a target sequence of length U and T acoustic frames we generate UxT softmax



Recurrent Neural Network Transducer (RNN-T)

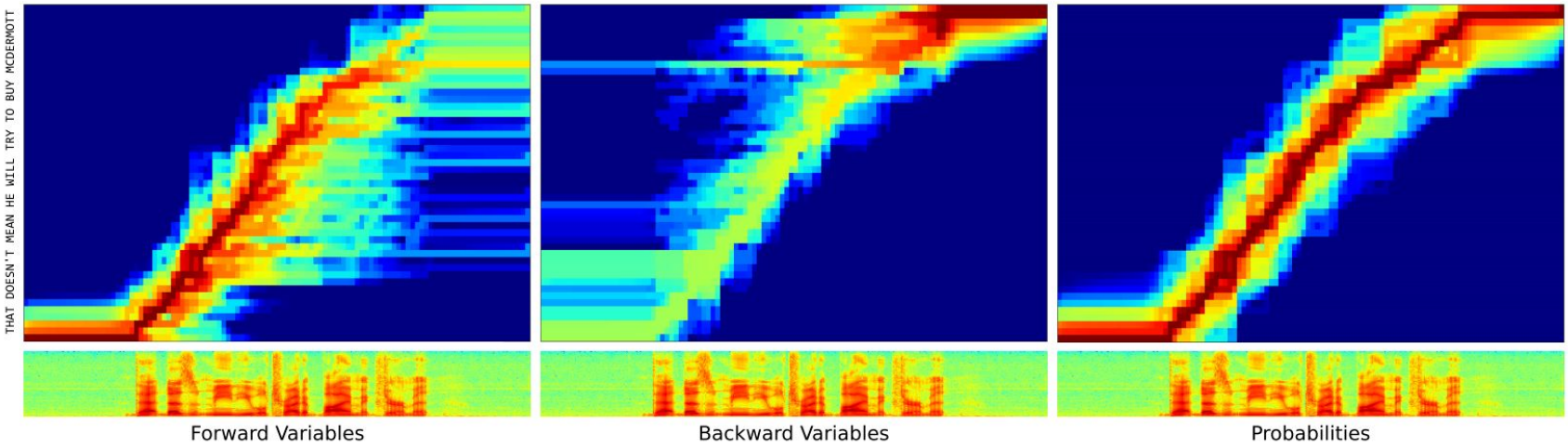


Figure 2. **Forward-backward variables during a speech recognition task.** The image at the bottom is the input sequence: a spectrogram of an utterance. The three heat maps above that show the logarithms of the forward variables (top) backward variables (middle) and their product (bottom) across the output lattice. The text to the left is the target sequence.

Reproduced from [Graves, 2012] ICML Representation Learning Workshop

Recurrent Neural Network Transducer (RNN-T)

Table 1. TIMIT Phoneme Recognition Results. 'Epochs' is the number of passes through the training set before convergence. 'PER' is the phoneme error rate on the core test set.

NETWORK	WEIGHTS	EPOCHS	PER
CTC-3L-500H-TANH	3.7M	107	37.6%
CTC-1L-250H	0.8M	82	23.9%
CTC-1L-622H	3.8M	87	23.0%
CTC-2L-250H	2.3M	55	21.0%
CTC-3L-421H-UNI	3.8M	115	19.6%
CTC-3L-250H	3.8M	124	18.6%
CTC-5L-250H	6.8M	150	18.4%
TRANS-3L-250H	4.3M	112	18.3%
PRETRANS-3L-250H	**4.3M**	**144**	**17.7%**

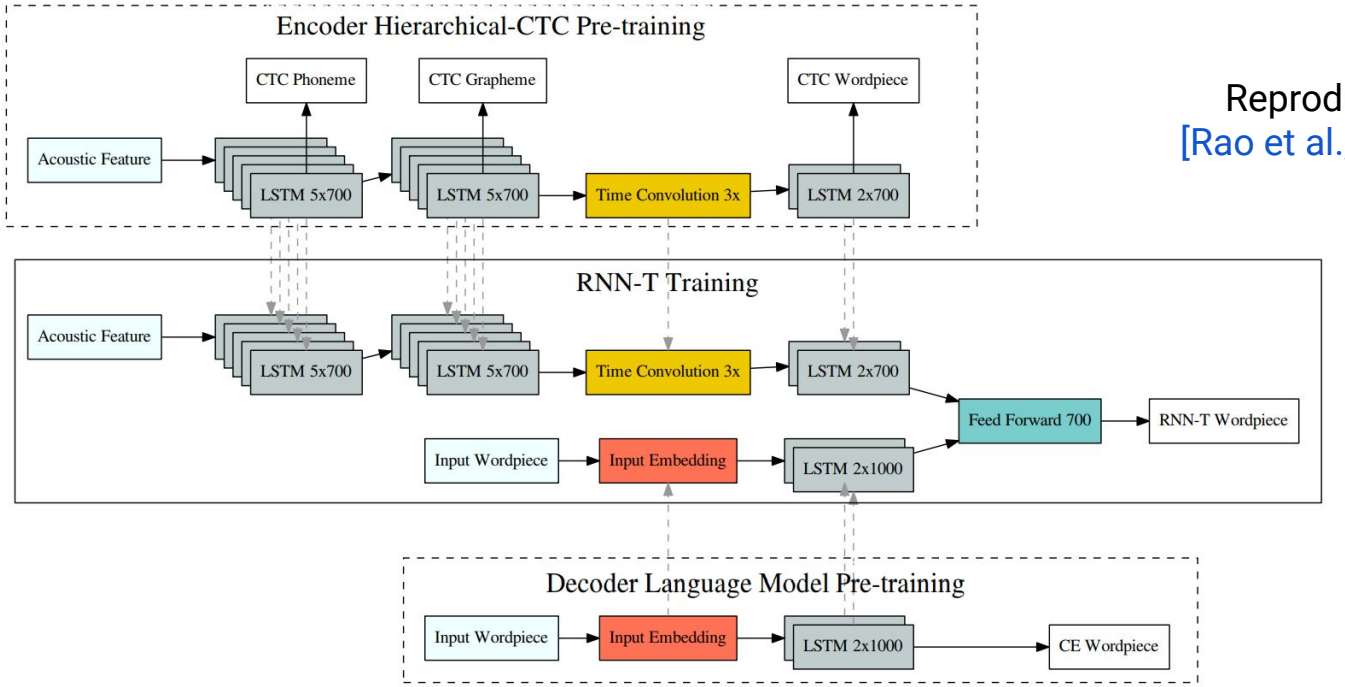
[Graves et al., 2013] showed promising results on TIMIT phoneme recognition, but the work did not seem to get as much traction in the field as CTC.

Reproduced from [Graves, 2013] ICASSP

Recurrent Neural Network Transducer (RNN-T)

- Intuitively, the prediction network corresponds to the “language model” component and the encoder corresponds to the “acoustic model” component
  - Both components can be initialized from a separately trained CTC-AM and a RNN-LM (which can be trained on text only data)
  - Initialization provides some gains [Rao et al., 2017] but is not critical to get good performance
- Generally speaking, RNN-T always seems to perform better than CTC alone in our experiments (even when decoded with a separate LM)
  - More on this in a bit when we compare various approaches on a voice search task.

RNN-T: Case Study on ~18,000 hour Google Data



Reproduced from
[Rao et al., 2017] ASRU

RNN-T components can be initialized separately from (hierarchical) CTC-trained AM, and recurrent LM. Initialization generally improves performance.

RNN-T: Case Study on ~18,000 hour Google Data

- If graphemes are used as output units, then the model has limited language modeling context: e.g. errors: “the tortoise and the hair”
- Using words as output targets would allow modeling additional context, but would introduce OOVs
- Intermediate: Use “word pieces” [Schuster & Nakajima, 2012]
  - Iteratively learn a vocabulary of units from text data.
  - Start with single graphemes, and train an LM from the data.
  - Iteratively combine units in a greedy manner which improve training perplexity
  - Continue to combine units until reaching a predefined number of units or perplexity improvements are below a threshold
  - E.g., “tortoise and the hare” → _tor to ise _and _the _hare

RNN-T: Case Study on ~18,000 hour Google Data

Units	Layers Encoder	Layers Decoder	Pre-trained Encoder	Pre-trained Decoder	Training Data Used Acoustic	Training Data Used Pronunciation	Training Data Used Text	Params	WER(%) VS	WER(%) IME
RNN-T										
Graphemes	5x700	2x700	no	no	yes	no	no	21M	13.9	8.4
Graphemes	5x700	2x700	yes	no	yes	no	no	21M	13.2	8.0
Graphemes	8x700	2x700	yes	no	yes	no	no	33M	12.0	6.9
Graphemes	8x700	2x700	yes	no	yes	yes	no	33M	11.4	6.8
Graphemes	8x700	2x700	yes	yes	yes	yes	yes	33M	10.8	6.4
Wordpieces-1k	12x700	2x700	yes	yes	yes	yes	yes	55M	9.9	6.0
Wordpieces-10k	12x700	2x700	yes	yes	yes	yes	yes	66M	9.1	5.3
Wordpieces-30k	12x700	2x1000	yes	yes	yes	yes	yes	96M	8.5	5.2
Baseline										
-	-	-	-	-	yes	yes	yes	120.2M	8.3	5.4

Initializing the “encoder” (i.e., acoustic model) helps improve performance by ~5%.

RNN-T: Case Study on ~18,000 hour Google Data

Units	Layers Encoder	Layers Decoder	Pre-trained Encoder	Pre-trained Decoder	Training Data Used Acoustic	Training Data Used Pronunciation	Training Data Used Text	Params	WER(%) VS	WER(%) IME
RNN-T										
Graphemes	5x700	2x700	no	no	yes	no	no	21M	13.9	8.4
Graphemes	5x700	2x700	yes	no	yes	no	no	21M	13.2	8.0
Graphemes	8x700	2x700	yes	no	yes	no	no	33M	12.0	6.9
Graphemes	8x700	2x700	yes	no	yes	yes	no	33M	11.4	6.8
Graphemes	8x700	2x700	yes	yes	yes	yes	yes	33M	10.8	6.4
Wordpieces-1k	12x700	2x700	yes	yes	yes	yes	yes	55M	9.9	6.0
Wordpieces-10k	12x700	2x700	yes	yes	yes	yes	yes	66M	9.1	5.3
Wordpieces-30k	12x700	2x1000	yes	yes	yes	yes	yes	96M	8.5	5.2
Baseline										
-	-	-	-	-	yes	yes	yes	120.2M	8.3	5.4

Initializing the “decoder” (i.e., prediction network, language model) helps improve performance by ~5%.

RNN-T: Case Study on ~18,000 hour Google Data

Units	Layers Encoder	Layers Decoder	Pre-trained Encoder	Pre-trained Decoder	Training Data Used Acoustic	Training Data Used Pronunciation	Training Data Used Text	Params	WER(%) VS	WER(%) IME
RNN-T										
Graphemes	5x700	2x700	no	no	yes	no	no	21M	13.9	8.4
Graphemes	5x700	2x700	yes	no	yes	no	no	21M	13.2	8.0
Graphemes	8x700	2x700	yes	no	yes	no	no	33M	12.0	6.9
Graphemes	8x700	2x700	yes	no	yes	yes	no	33M	11.4	6.8
Graphemes	8x700	2x700	yes	yes	yes	yes	yes	33M	10.8	6.4
Wordpieces-1k	12x700	2x700	yes	yes	yes	yes	yes	55M	9.9	6.0
Wordpieces-10k	12x700	2x700	yes	yes	yes	yes	yes	66M	9.1	5.3
Wordpieces-30k	12x700	2x1000	yes	yes	yes	yes	yes	96M	8.5	5.2
Baseline										
-	-	-	-	-	yes	yes	yes	120.2M	8.3	5.4

The RNN-T model with ~96M parameters can match the performance of a conventional sequence-trained CD-phone based CTC model with a large first pass LM

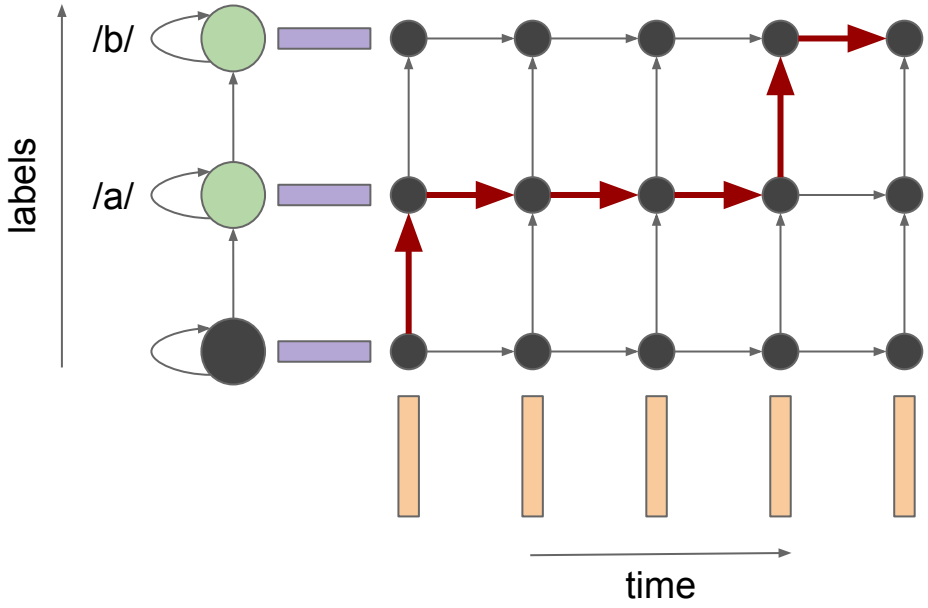
Further Improvements

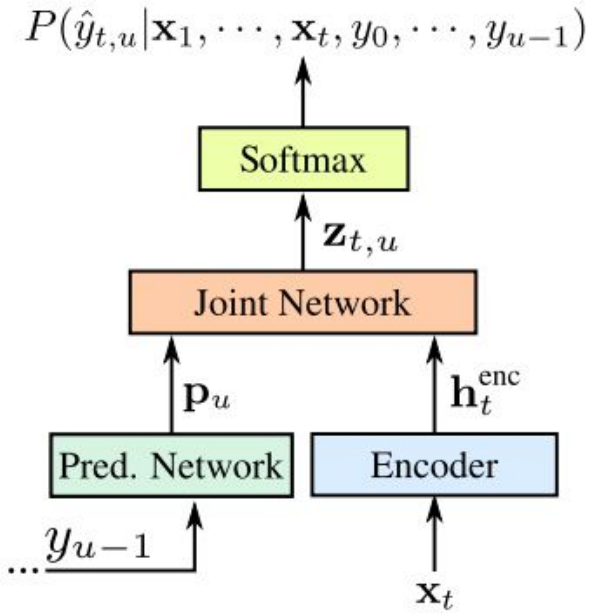
Improved Architecture

- Recurrent projection layer [Sak et al., 2014]
  - Introduced after each LSTM layer in both the encoder and decoder.
  - Improves accuracy with more compact representation.

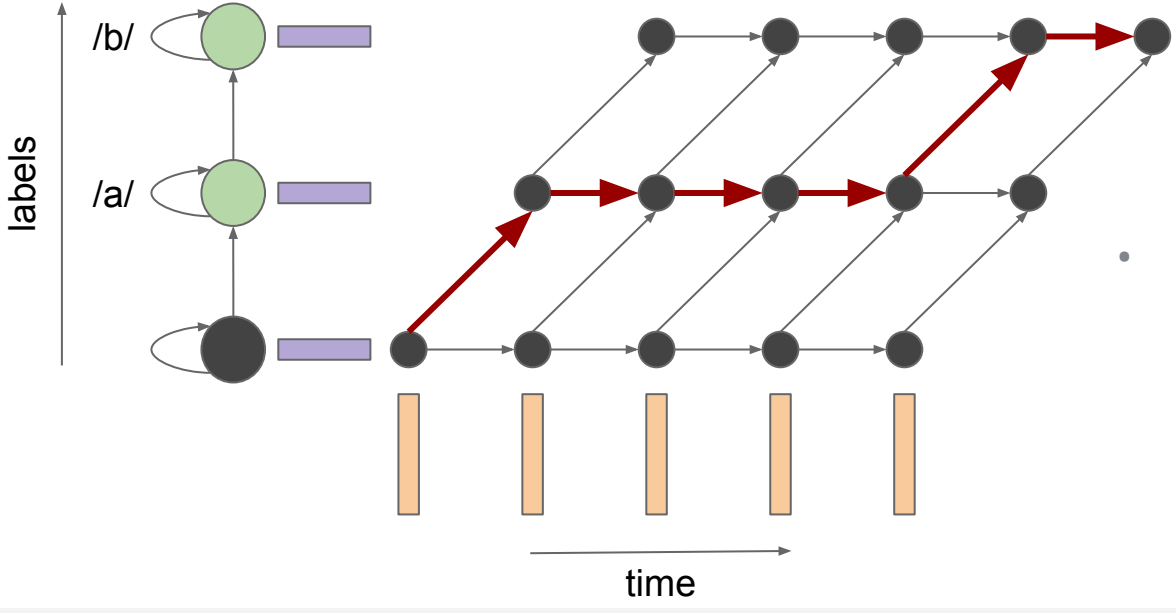
- Layer normalization [Ba et al., 2016]
  - Applied to all layers.
  - Stabilizes hidden state dynamics of the recurrent layers.
  - Improves accuracy.

Forward-Backward Algorithm for Training





Efficient Forward-Backward in TPU



Native TF support for efficient batched forward-backward computation.
[Sim et al., 2017]

- Main idea: shift the matrix to do pipelining (so that each column only depends on the previous column but not itself).

This allows us to train faster in TPUs with much larger batch sizes than would be possible using GPUs, which improves accuracy.

Comparisons of Streaming Models for Mobile Devices

More details: [He et al., 2018]

- Inputs:
  - Each 10ms frame feature: 80-dimensional log-Mel.
  - Every 3 frames are stacked as the input to the networks, so the effective frame rate is 30ms.

- RNN-T:
  - 8-layer encoder, 2048 unidirectional LSTM cells + 640 projection units.
  - 2-layer prediction network, 2048 unidirectional LSTM cells + 640 projection units.
  - Model output units: grapheme or word-piece.
  - Total system size: ~120MB after quantization (more on this in a later slide).

- A competitive CTC baseline model for embedded speech recognition:
  - Similar to [McGraw et al., 2016], but with bigger models.
  - 6 layers, 1200 unidirectional LSTM cells + 400 projection units. sMBR sequence-trained.
  - Model output units: CI phones.
  - With a 5-gram first-pass LM, and a second-pass rescoring LSTM LM.
  - Total system size: ~130MB after quantization.

Quality Improvements

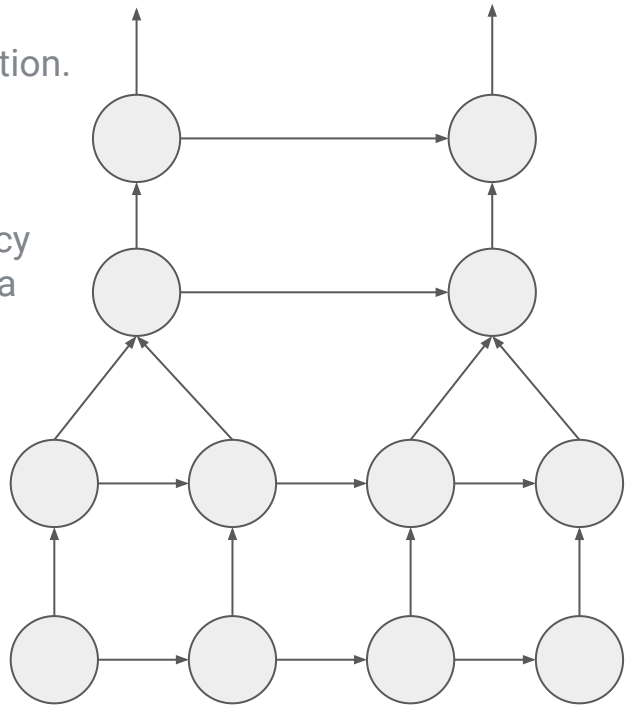
Model	VS WER	IME WER
RNN-T Grapheme	8.1%	4.9%
+ Layer Norm	7.6%	4.6%
+ Large Batch	7.5%	4.4%
+ Word-piece	**6.8%**	**4.0%**
CTC	9.3%	5.3%

With all the optimizations, a streaming RNN-T model improves WER by more than 20% over a conventional CTC embedded model + word LMs.

Real-Time Recognition

Time Reduction Layer

- **Time reduction layer** [Chan et al., 2015, Soltau et al., 2017]
  - Reduces the sequence length in the encoder by concatenation.
  - Speeds up training and inference.
  - Inserted after two encoder LSTM layers to maximize computation saving without hurting accuracy.
  - Input frame rate 30ms, output frame rate 60ms: no accuracy loss in RNN-T grapheme and wordpiece models, but hurts a CTC phoneme model.



More Inference Optimization

- **Prediction network state caching**
  - The prediction network computation is independent of the acoustics.
  - Analogous to an RNN language model.
  - Apply the same state caching technique used in RNN LM in order to avoid redundant computation for identical prediction histories.
  - In practice, 50–60% of the prediction network computations are cached during beam search, with different beam sizes.

- **Encoder / prediction network multithreading**
  - Encoder splits over two threads: before and after time-reduction layer.
  - Prediction network runs in a separate thread.
  - Enables pipelining among different threads through asynchrony.
  - Results in a speed-up of 28% with respect to a single-threaded execution.

Parameter Quantization

- Quantizes parameters from 32-bit floating-point precision into 8-bit fixed-point.
  - Reduces model size by 4x.
- Symmetric quantization
  - Assumes that parameter values are distributed around floating point zero for quantization.
  - Directly uses the quantized vector/matrix for multiplication without offset, which is more efficient compared to asymmetric quantization with offset.
  - 3x speedup compared to floating point execution.

Comparisons of Streaming Models for Mobile Devices (cont'd)

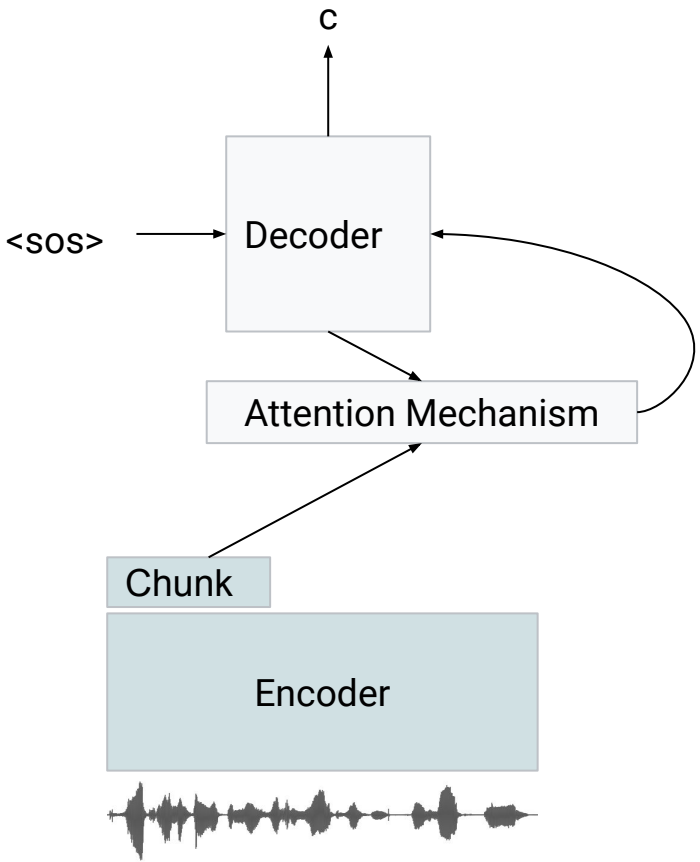
Real Time (RT) Factor (processing time divided by audio duration) is measured on a Google Pixel Phone.
RT90: Real time factor at 90 percentile. Lower is better.

Model	Weight Storage Type	Size	Output Frame Rate	VS WER	IME WER	RT90
RNN-T Grapheme	Float	468MB	30ms	7.4%	4.5%	2.71
+ Time Reduction Layer	Float	468MB	60ms	7.5%	4.4%	1.58
+ Word-piece	Float	480MB	60ms	7.0%	4.1%	1.43
+ Asymmetric Quantization	Int	120MB	60ms	7.1%	4.2%	1.03
+ Symmetric Quantization	**Int**	**120MB**	**60ms**	**7.3%**	**4.2%**	**0.51**
CTC + Symmetric Quantization	Int	130MB	30ms	9.2%	5.4%	0.86

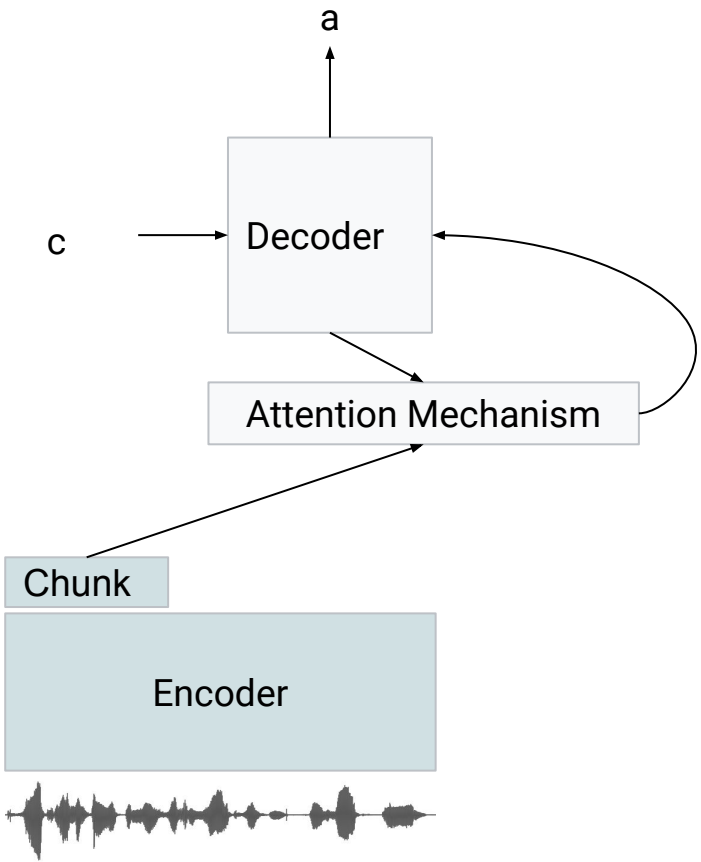
RNN-T decodes speech twice as fast as real time on a Google Pixel phone, which improves WER by more than 20% relative to a conventional CTC embedded model.

Neural Transducer

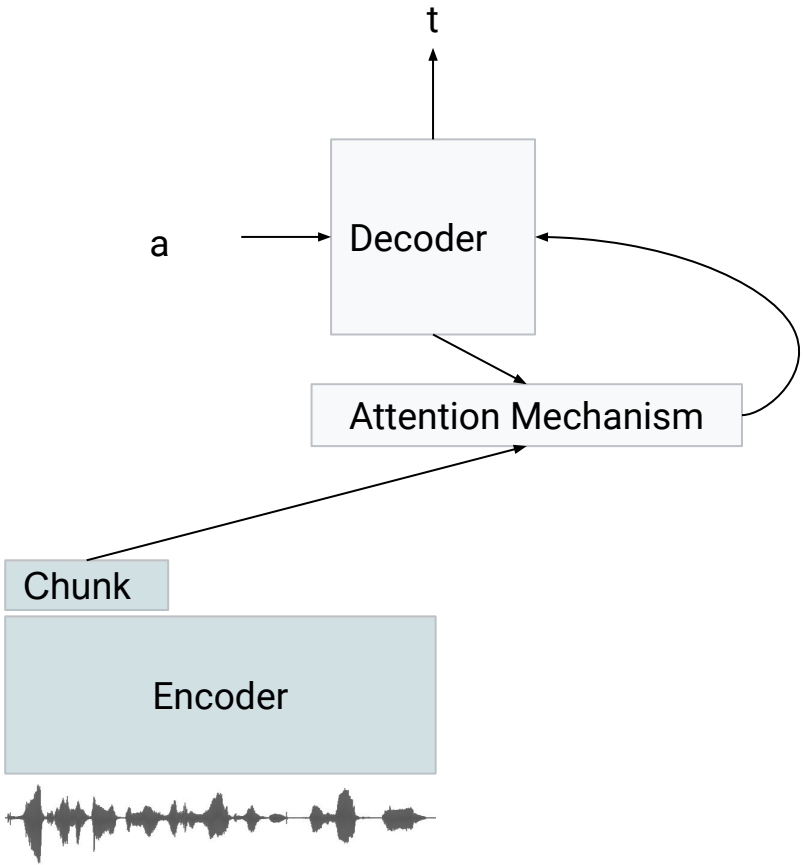
Neural Transducer: “Online” Attention Models



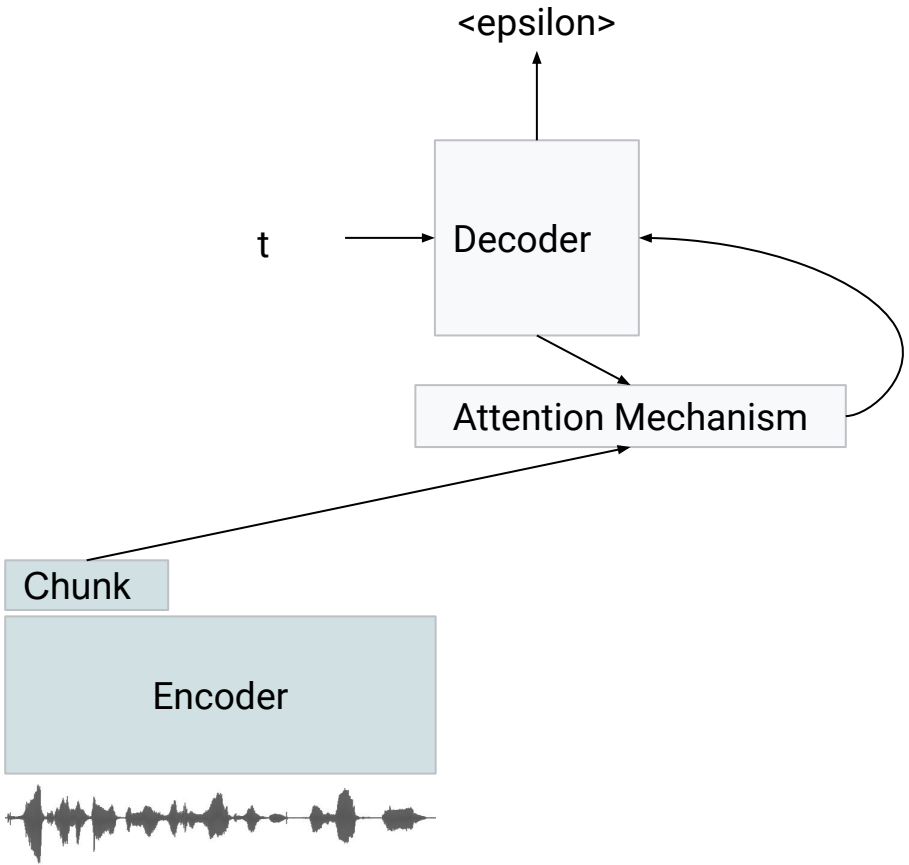
Neural Transducer: “Online” Attention Models



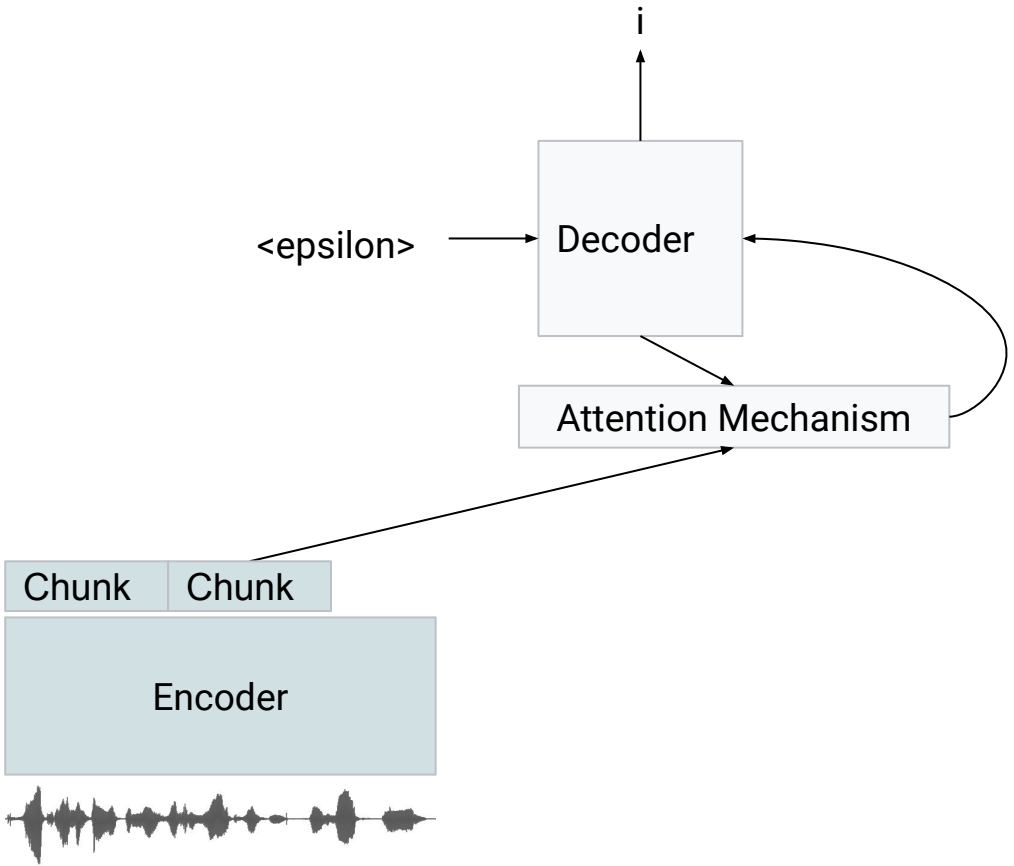
Neural Transducer: “Online” Attention Models



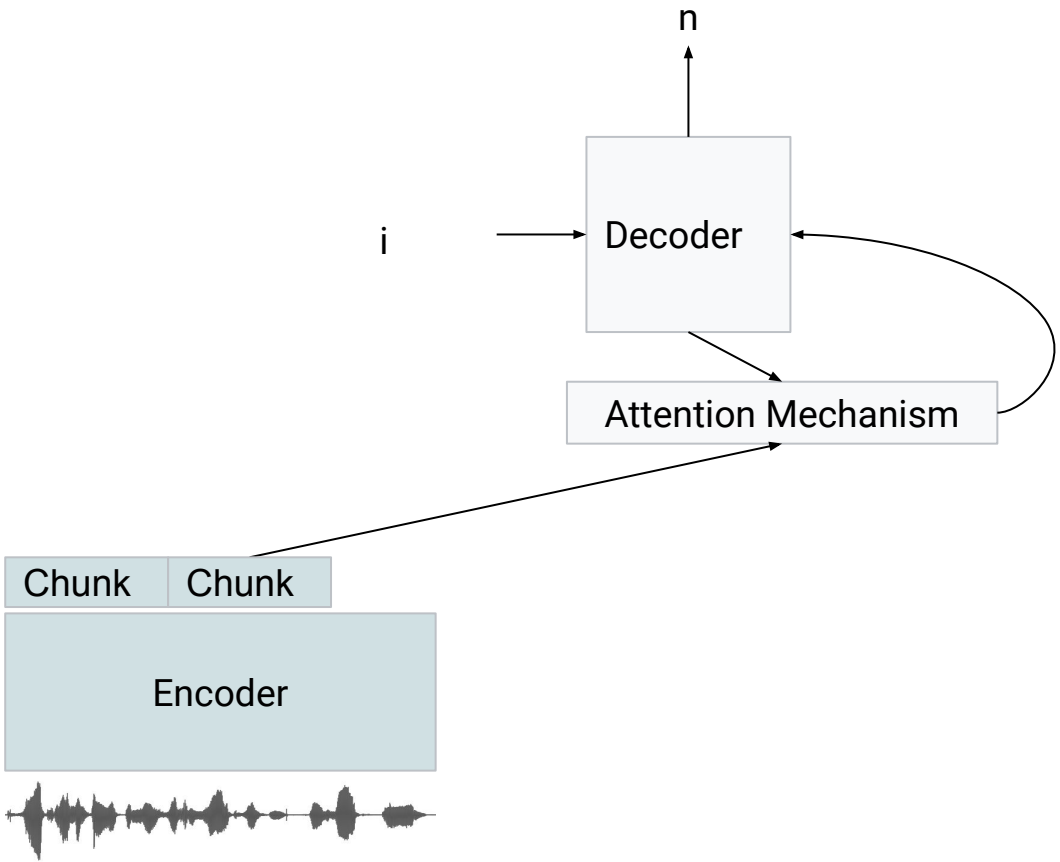
Neural Transducer: “Online” Attention Models



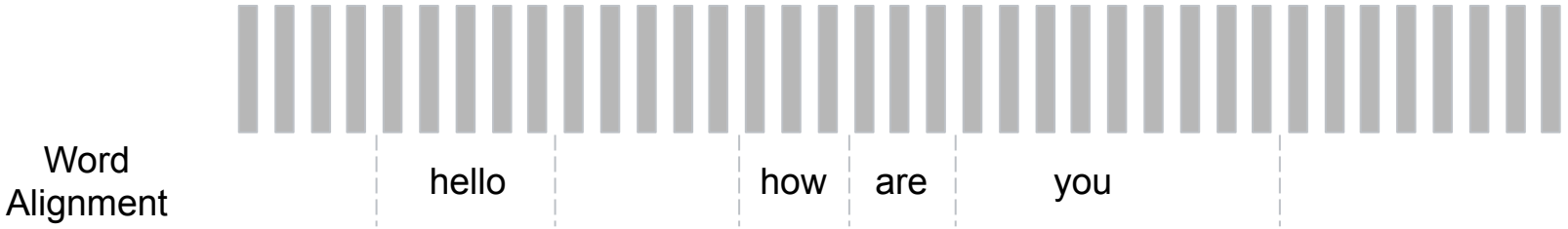
Neural Transducer: “Online” Attention Models



Neural Transducer: “Online” Attention Models

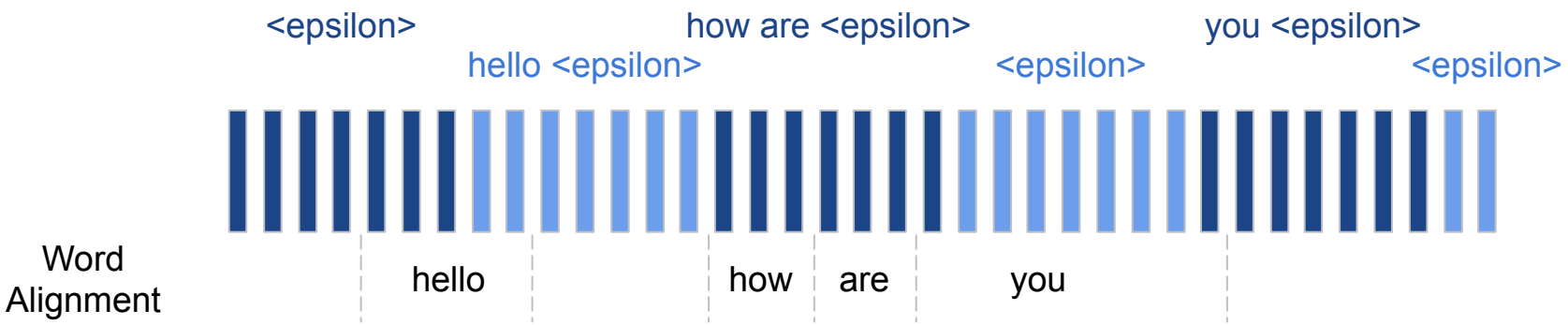


Training Data for Neural Transducer



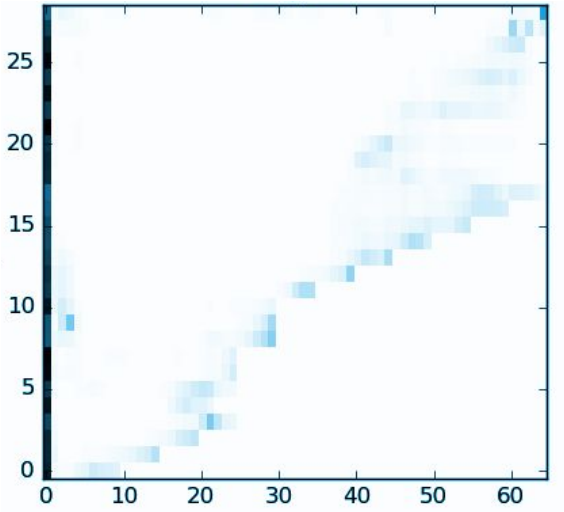
- Online methods like RNN-T, Policy Gradient learn alignment jointly with model
- We train neural transducer with a pre-specified alignment, so don't need to re-compute alignments (e.g., forward-backward) during training, which slows things down on GPU/TPU.

Training Data for Neural Transducer

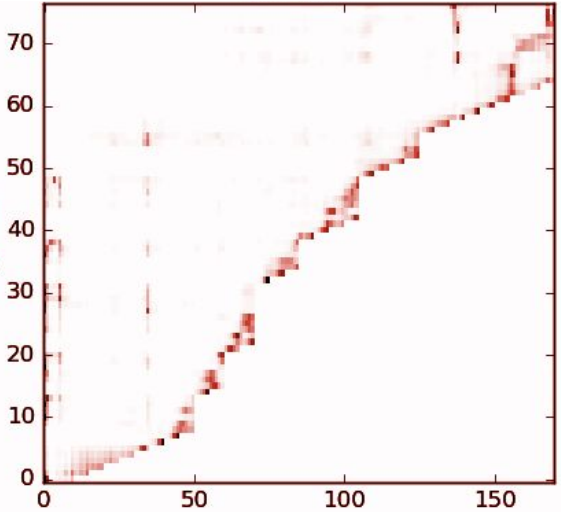


- <epsilon> signals end-of-chunk
- Since we don't have grapheme-level alignments, we wait till the end of the word to emit the entire word's graphemes

Neural Transducer Attention Plot



Unidirectional LAS with Multi-Headed Attention

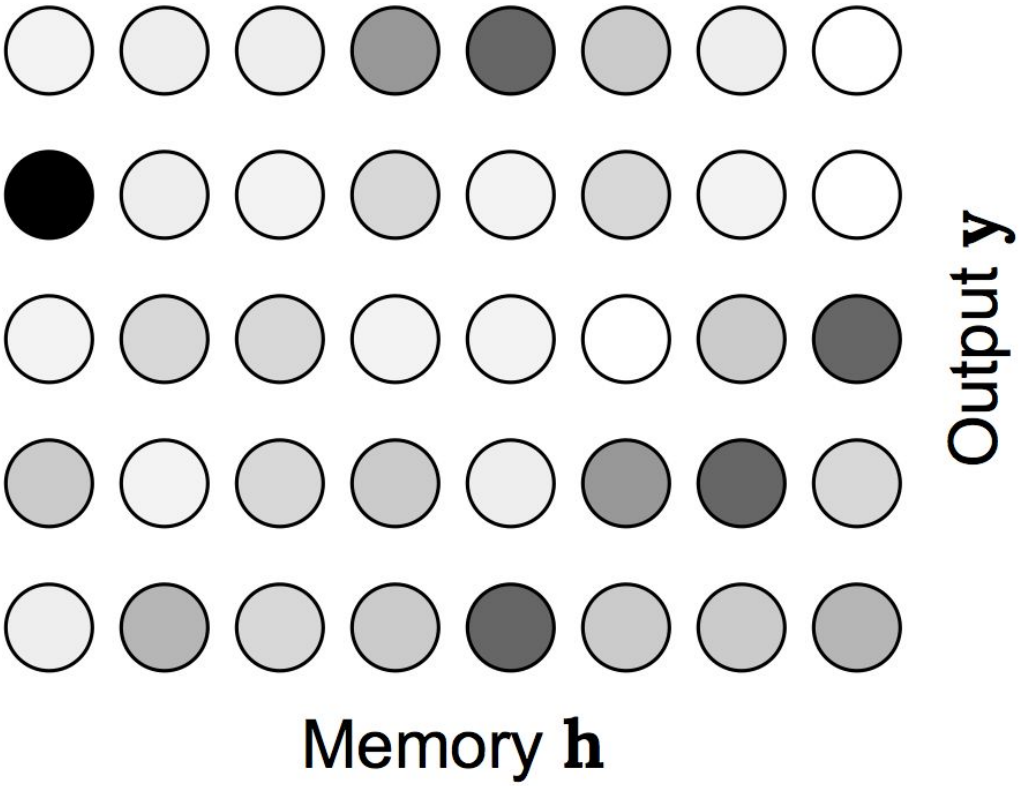


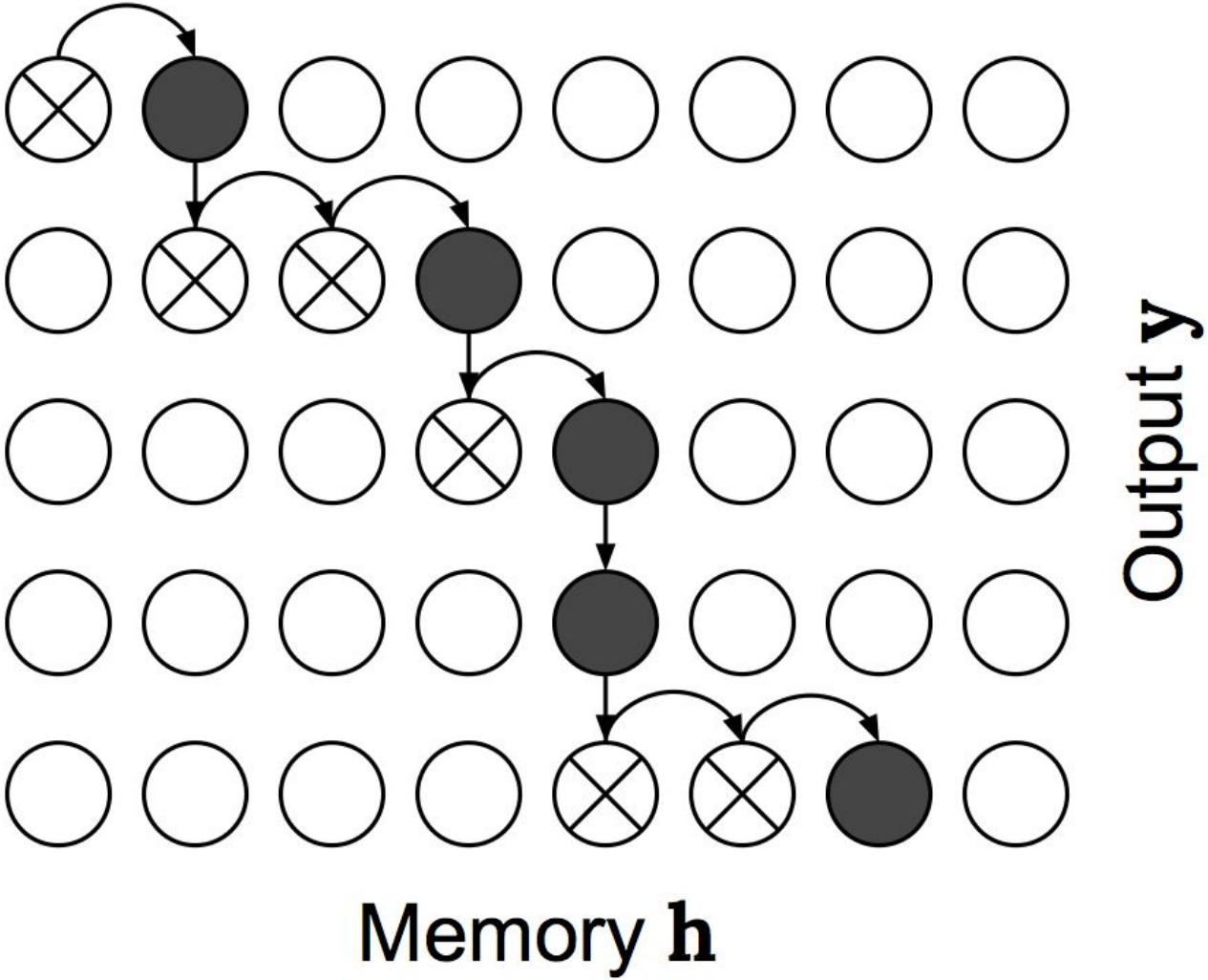
Neural Transducer Attention

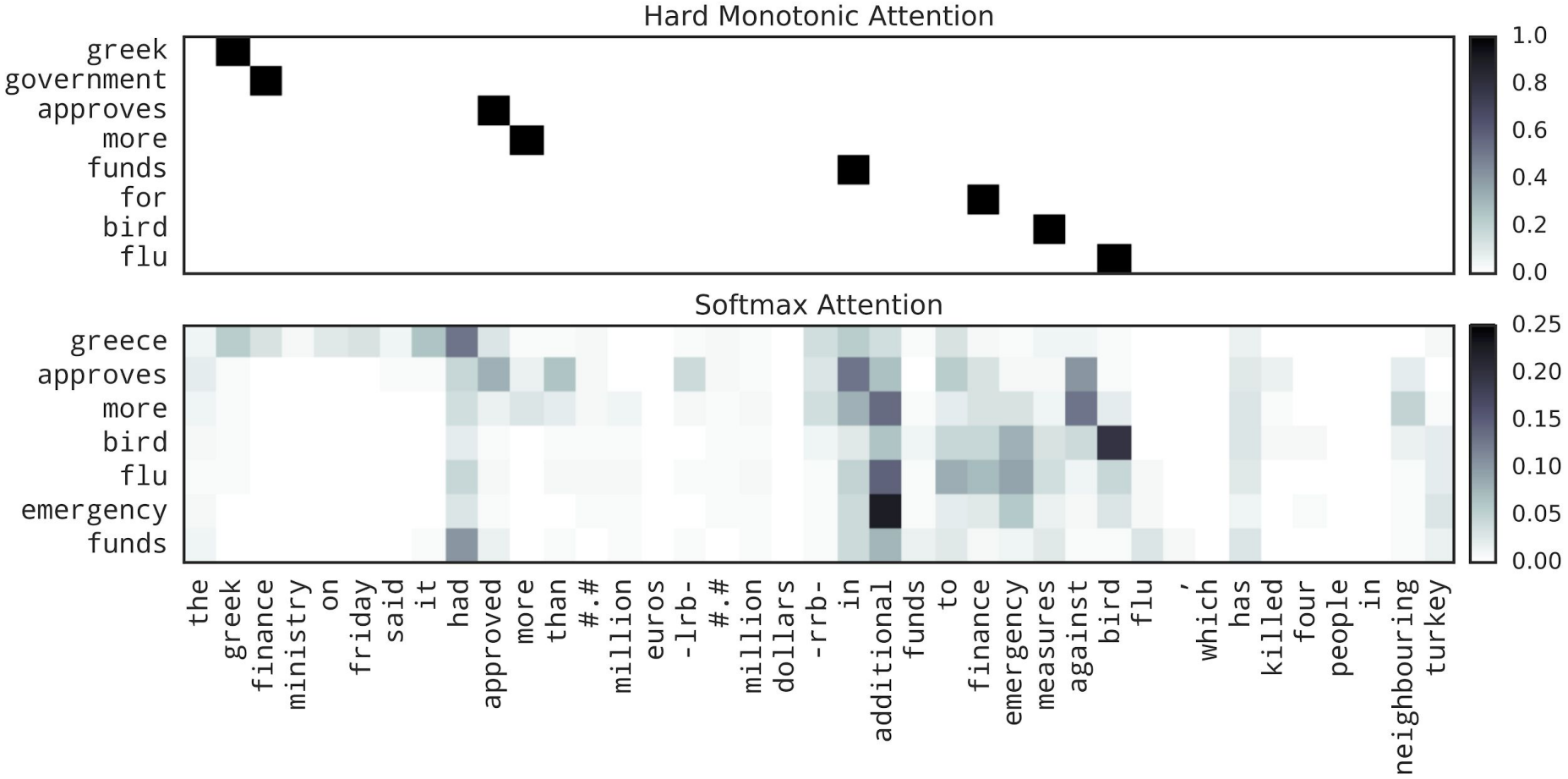
NT model examines previous frames without looking beyond the current chunk

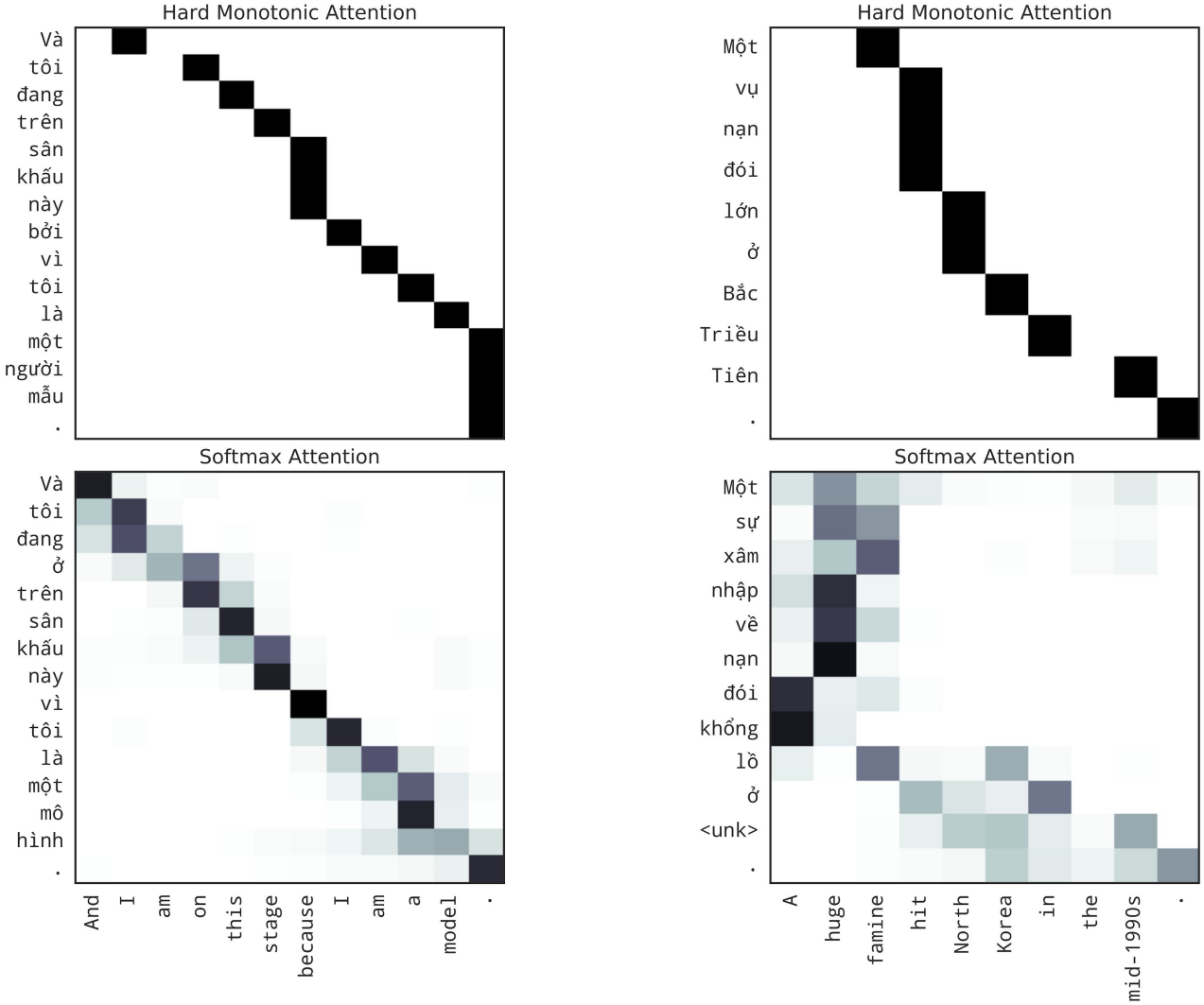
Monotonic Chunkwise Attention

Monotonic Attention

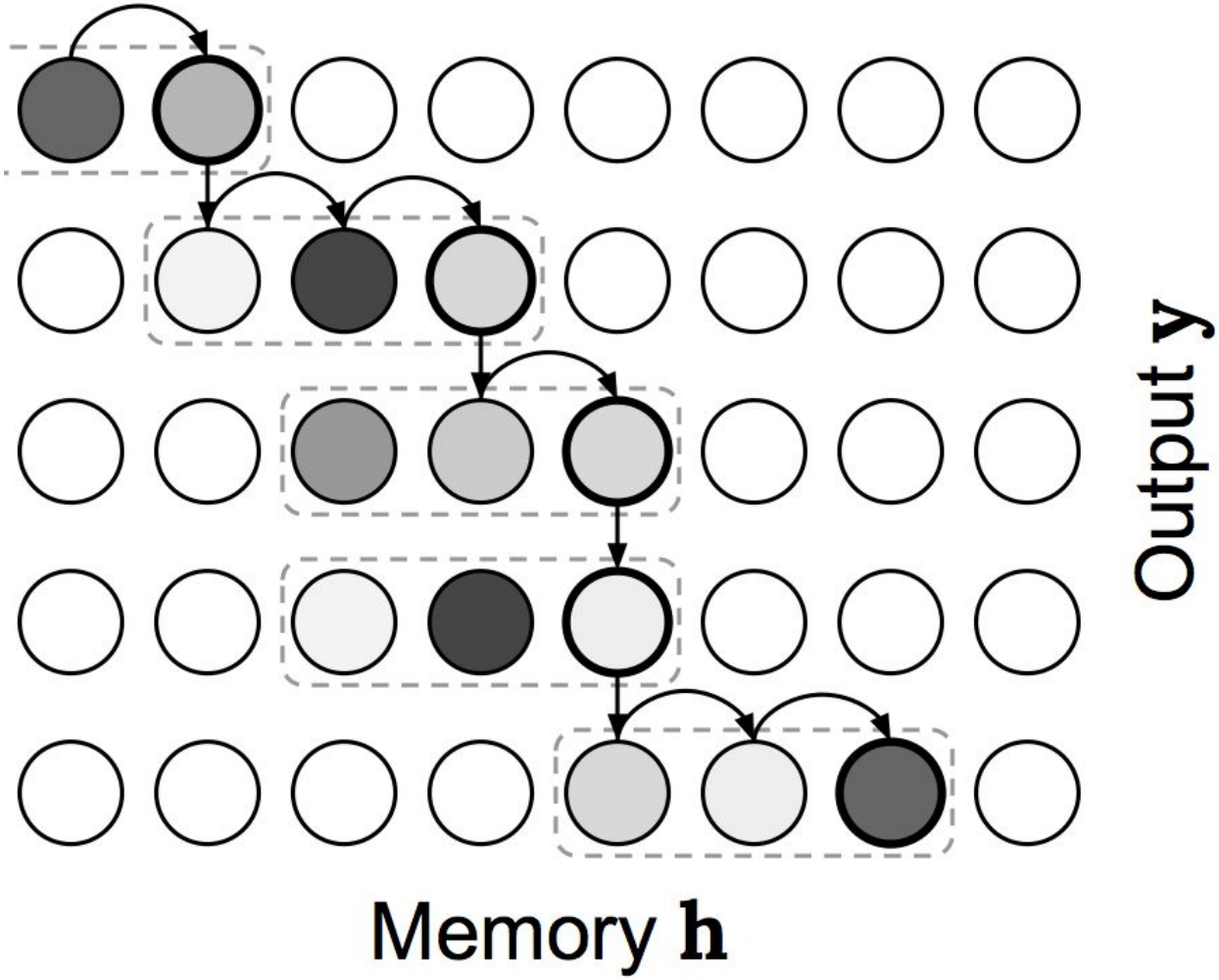


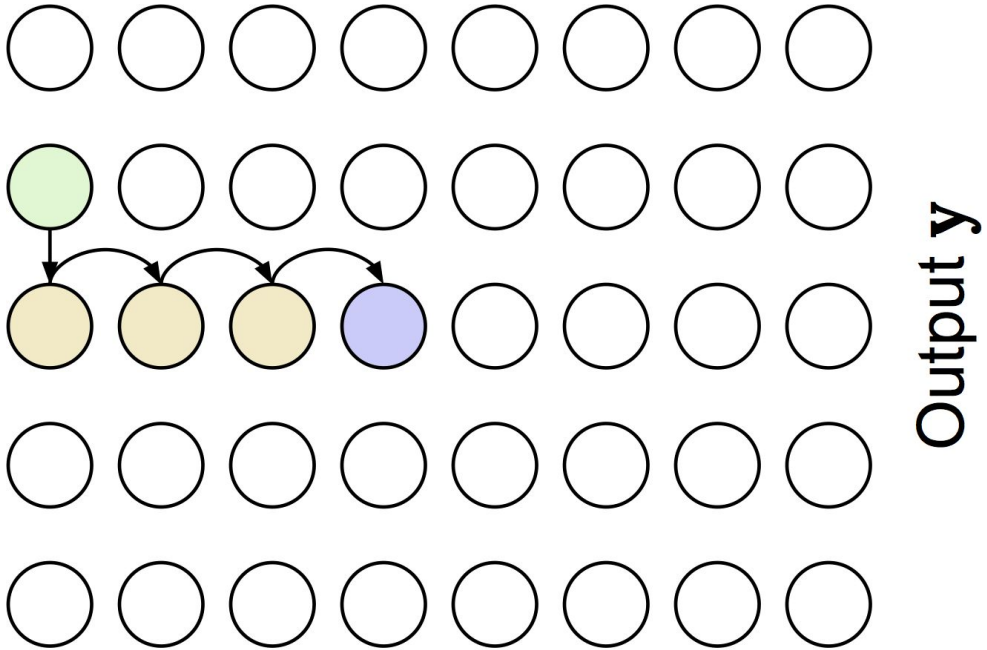

Output y
Memory h


Hard Monotonic Attention
greek
government
approves
more
funds
for
bird
flu
1.0
0.8
0.6
0.4
0.2
0.0
Softmax Attention
greece
approves
more
bird
flu
emergency
funds
0.25
0.20
0.15
0.10
0.05
0.00
the
greek
finance
ministry
on
friday
said
it
had
approved
more
than
#.#
million
euros
-lrb-
#.#
million
dollars
-rrb-
in
additional
funds
to
finance
emergency
measures
against
bird
flu
,
which
has
killed
four
people
in
neighbouring
turkey


Hard Monotonic Attention
Và
tôi
đang
trên
sân
khấu
này
bởi
vì
tôi
là
một
người
mẫu
.
Softmax Attention
Và
tôi
đang
ở
trên
sân
khấu
này
vì
tôi
là
một
mô
hình
.
And
I
am
on
this
stage
because
I
am
a
model
.
Hard Monotonic Attention
Một
vụ
nạn
đói
lớn
ở
Bắc
Triều
Tiên
.
Softmax Attention
Một
sự
xâm
nhập
về
nạn
đói
khổng
lồ
ở
<unk>
.
A
huge
famine
hit
North
Korea
in
the
mid-1990s
.

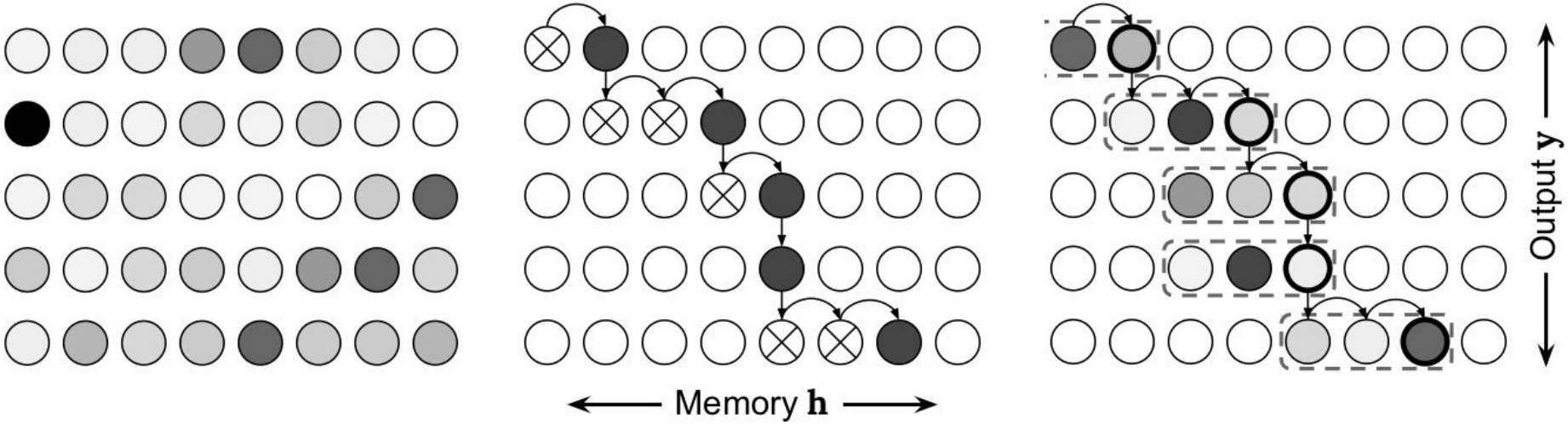
MoChA





$$\text{attention}_{i,j} = \text{select}_{i,j} \sum_{k=1}^{j} \left(\text{attention}_{i-1,k} \prod_{l=k}^{j-1} (1 - \text{select}_{i,l}) \right)$$

Monotonic Chunkwise Attention (MoChA)



Soft attention

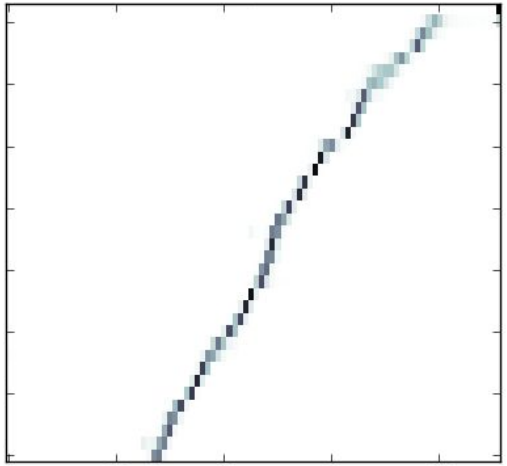
Hard monotonic attention

Monotonic chunkwise attention

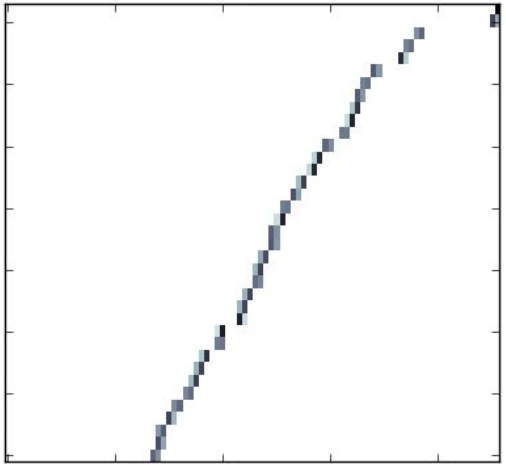
[Chiu and Raffel, 2018]

Training Monotonic Chunkwise Attention

- Compute expected probability of hard attention
- The expected probability distribution provides a soft attention
- Same training procedure as LAS



Train



Inference

Online Model Comparison

Model	Clean	
	Voice Search	Dictation
LAS	5.7	4.1
RNN-T	6.8	4.0
MoChA	5.8	4.2
NT	8.7	7.8

Summary

- Explored multiple alternatives of online end-to-end models.

  Achieving streaming recognition at a small cost of accuracy over LAS.

- Investigated RNN-T in depth as one promising choice.

  Substantial accuracy and efficiency improvement over a conventional CTC embedded model with similar size.

Toward product excellence

Personalization

Fast endpointing

Multi-dialect ASR

Personalization

“Bias” the priors to the speech models based on personal information

Example

- Contacts - “call Joe Doe, send a message to Jason Dean”
- Songs - “play Lady Gaga, play songs from Jason Mraz”
- Dialog- “yes, no, stop, cancel”

Why Important

- Biasing can improve WER in domains by more than 10% relative

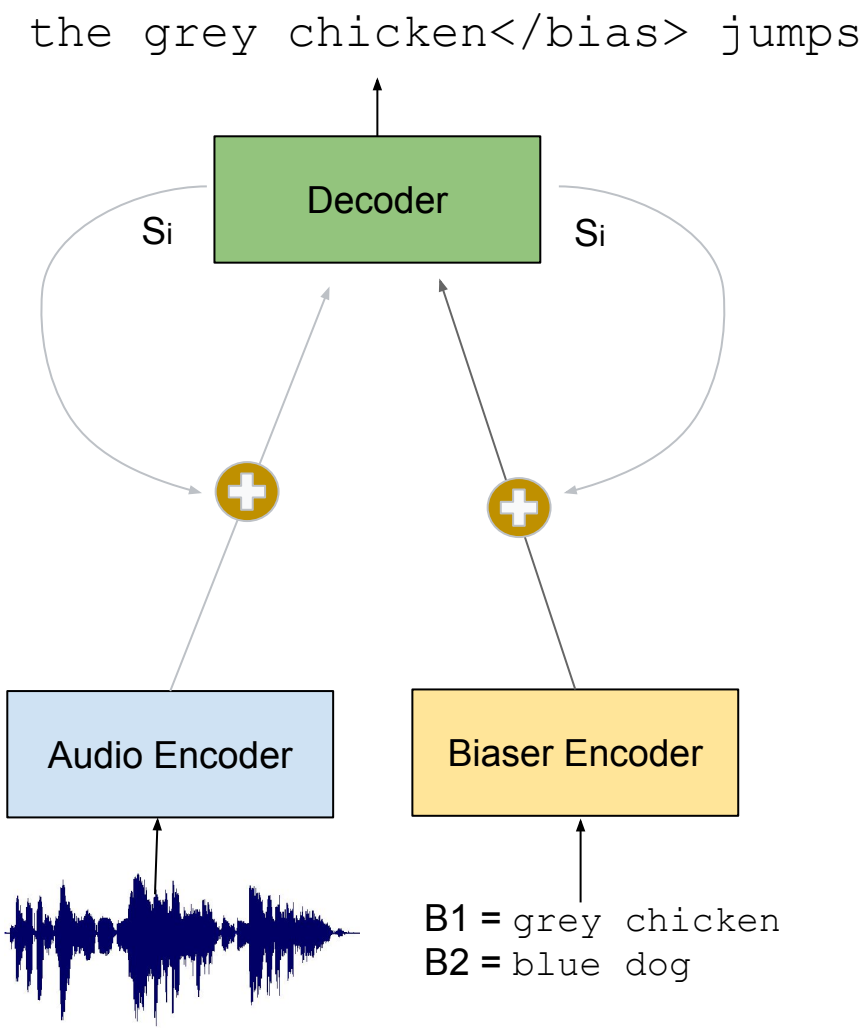
Test Set	WER, No Biasing	WER, Biasing
Contacts	15.0	2.8

How To Bias E2E Models

- Paper reference [Pundak et al., 2018]
- Test sets
  - Songs - “play Lady Gaga, play songs from Jason Mraz”

Contextual LAS Model (CLAS)

- Directly model $P(y|x, z)$
  - z: bias-phase e.g. contact name/song list
- Goal: bias the model towards outputting particular phrases
- Predict </bias>: the model has to attend to the correct bias phrase

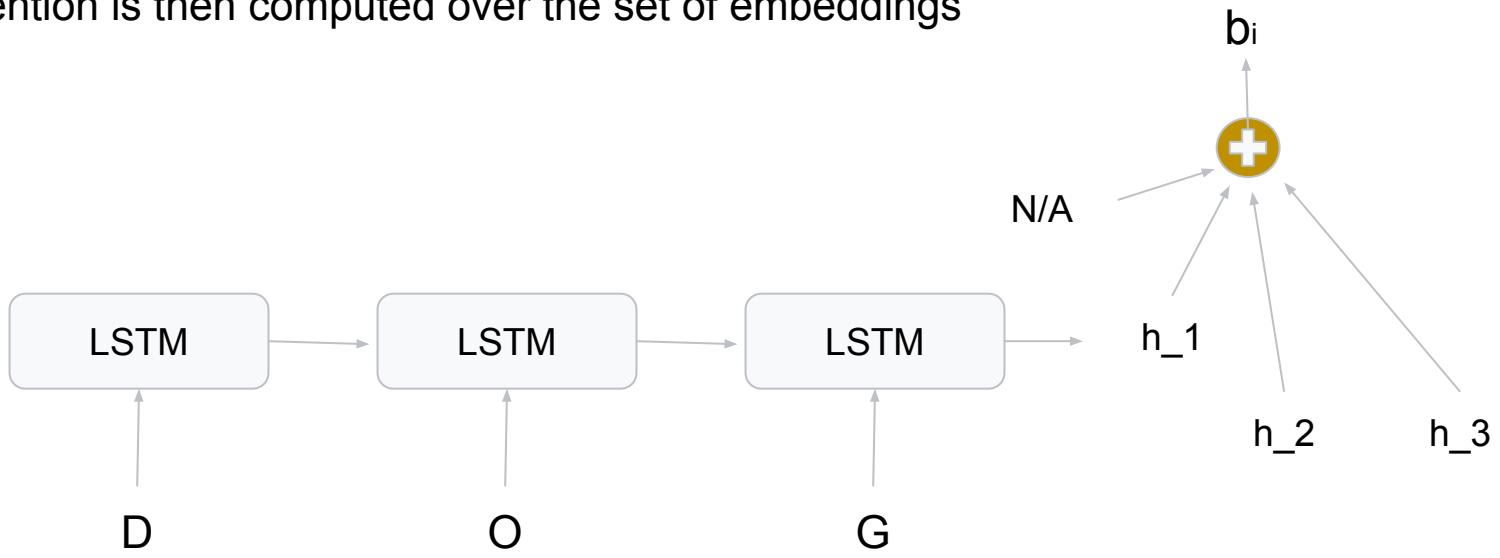


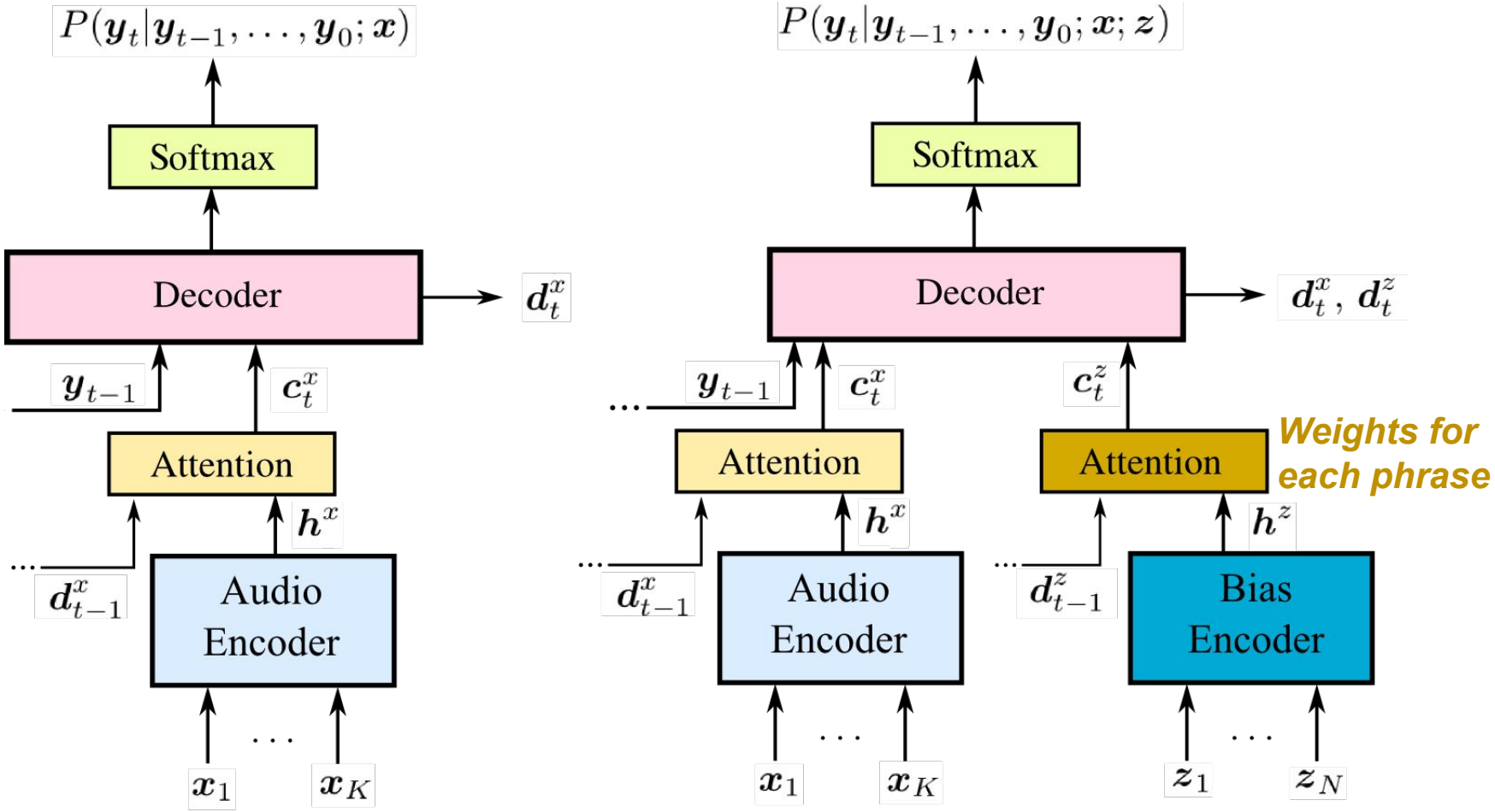
CLAS training

- Example ref: `The grey chicken jumps over the lazy dog`
- Sample uniformly a bias phrase ***b***, e.g. `grey chicken`
- With **drop-probability** p (e.g. 0.5) drop the selected phrase
- Augment with additional N-1 more bias phrases from other references in the batch
  - `quick turtle`
  - `grey chicken`
  - `brave monkey`
- If ***b*** was not dropped, insert a `</bias>` token to reference:
  - `The grey chicken</bias> jumps over the lazy dog`

The Biaser

- The Biaser embeds each phrase into a fixed length vector
  - → Last state of an LSTM
- Embedding happens once per bias phrase (possibly offline)
  - Cheap computation
- Attention is then computed over the set of embeddings

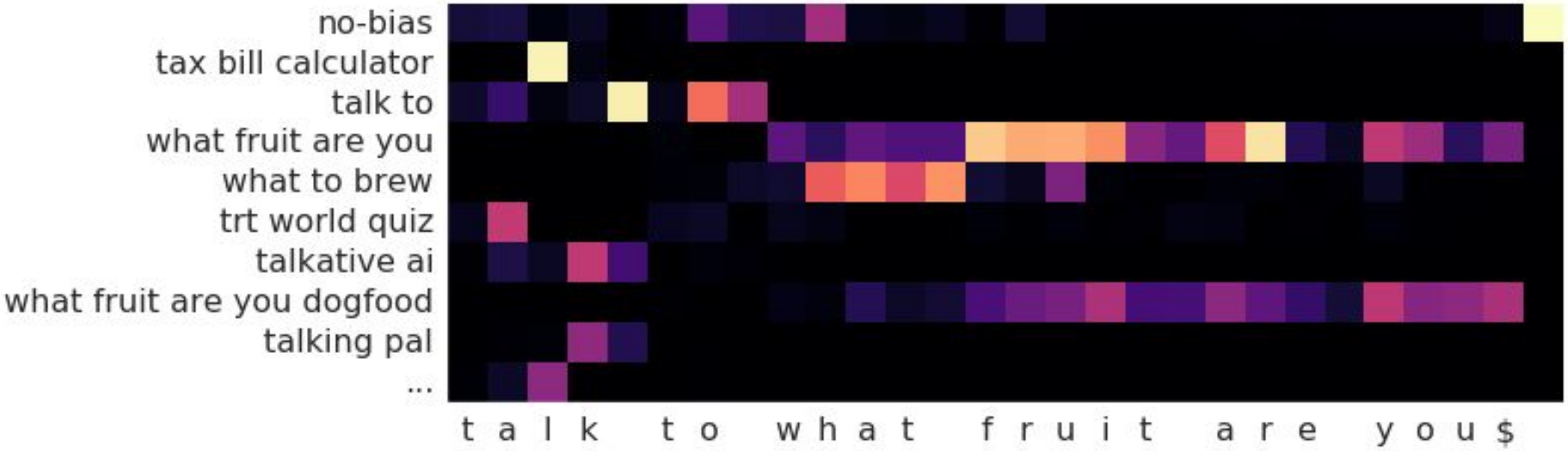




(a.) LAS Model

(b.) CLAS Model

Example


no-bias
tax bill calculator
talk to
what fruit are you
what to brew
trt world quiz
talkative ai
what fruit are you dogfood
talking pal
...
t a l k t o w h a t f r u i t a r e y o u $

Summary

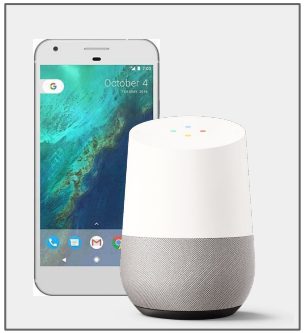
- CLAS is better than biasing with extern LM
- CLAS model + external LM works best

Method	Songs
LAS No Biasing	20.9
LAS + external LM	13.4
CLAS	6.9
CLAS + external LM	5.7

Endpointer

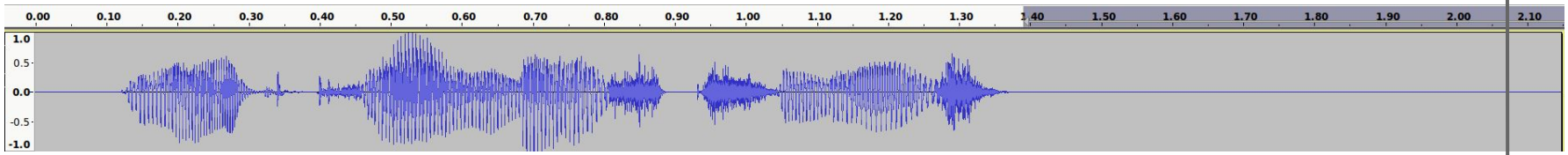
“Determine when the user has finished speaking: fast and accurate ASR”

Streaming speech recognition



Finalize recognition
Fetching the search results

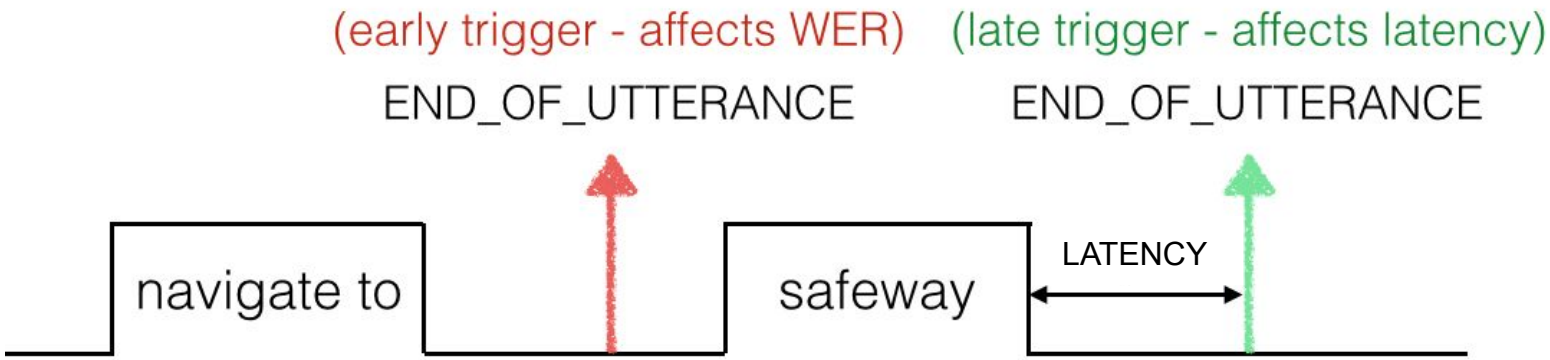
Endpoint quickly



Recognize the audio

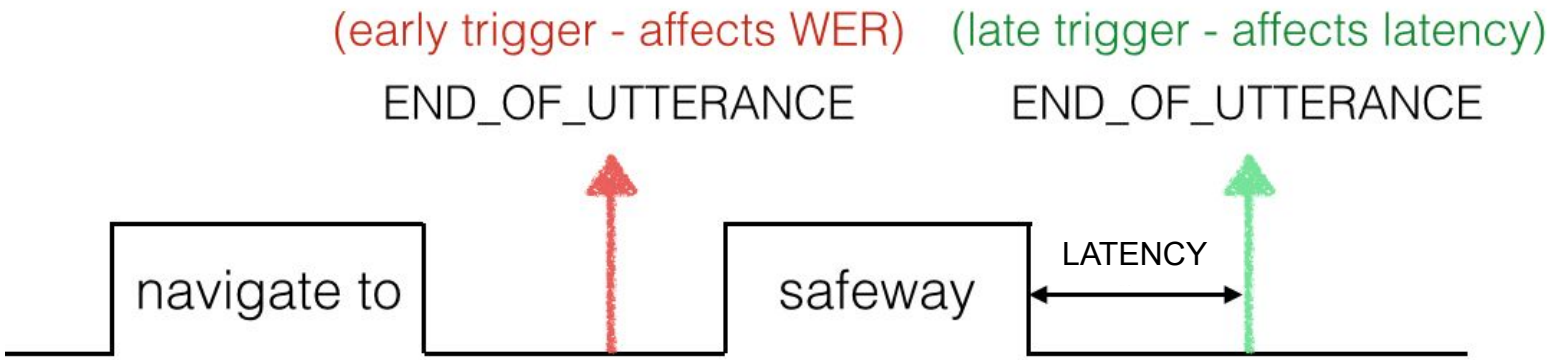
Why is it hard?

1. Latency vs WER tradeoff

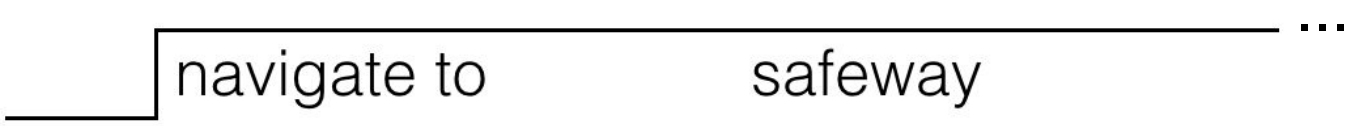


Why is it hard?

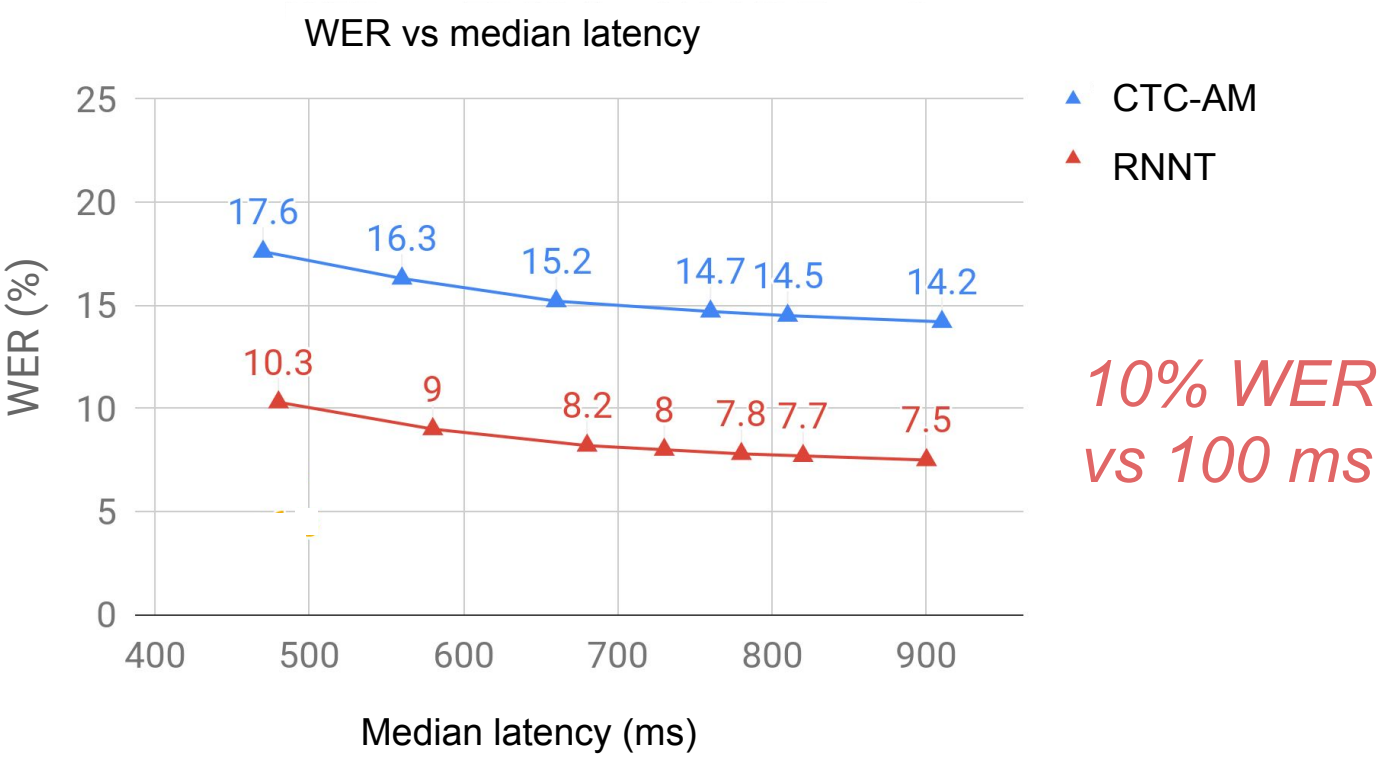
1. Latency vs WER tradeoff



2. Noisy conditions



Endpointer: Latency and WER Impact

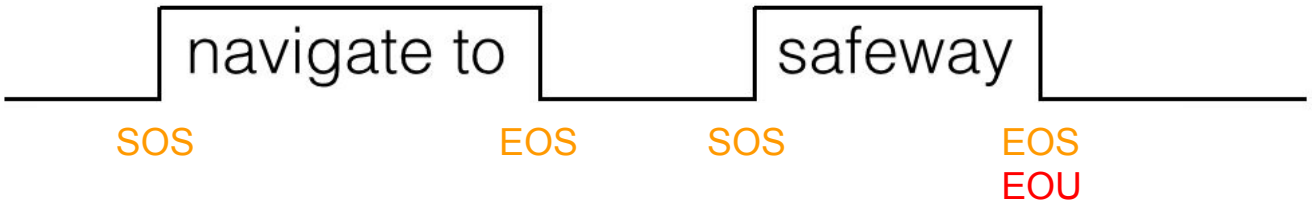


10% WER vs 100 ms

VAD vs end-of-query detection

VAD: detect speech vs non-speech

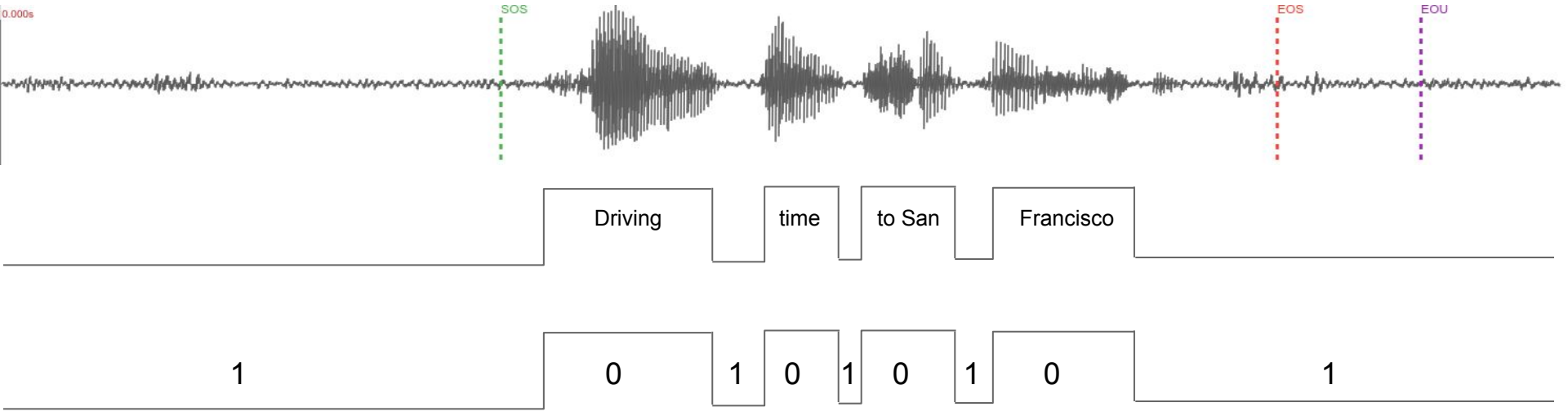
VAD and end-of-query detector (user finished speaking) are different



SOS = start of speech
EOS = end of speech
EOU = end of utterance

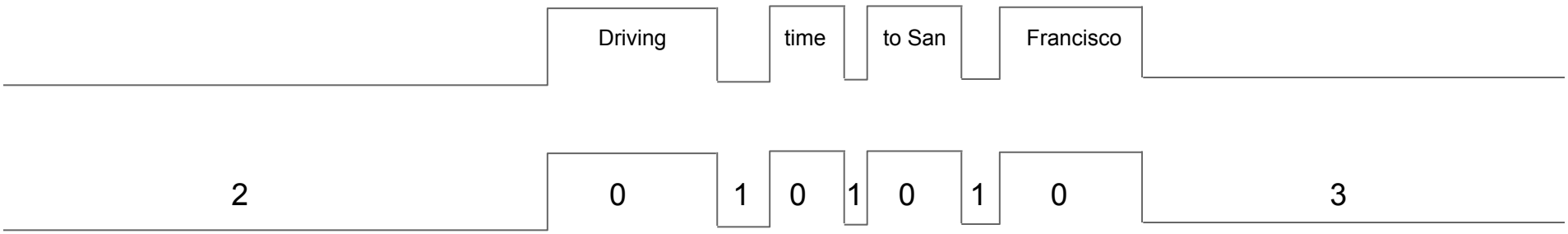
Training a Voice Activity Detector

- Take an utterance with ground-truth transcription
- Use forced alignment to find the timing of the utterance
- Based on the timing mark each frame as SPEECH (0) or NON-SPEECH (1)



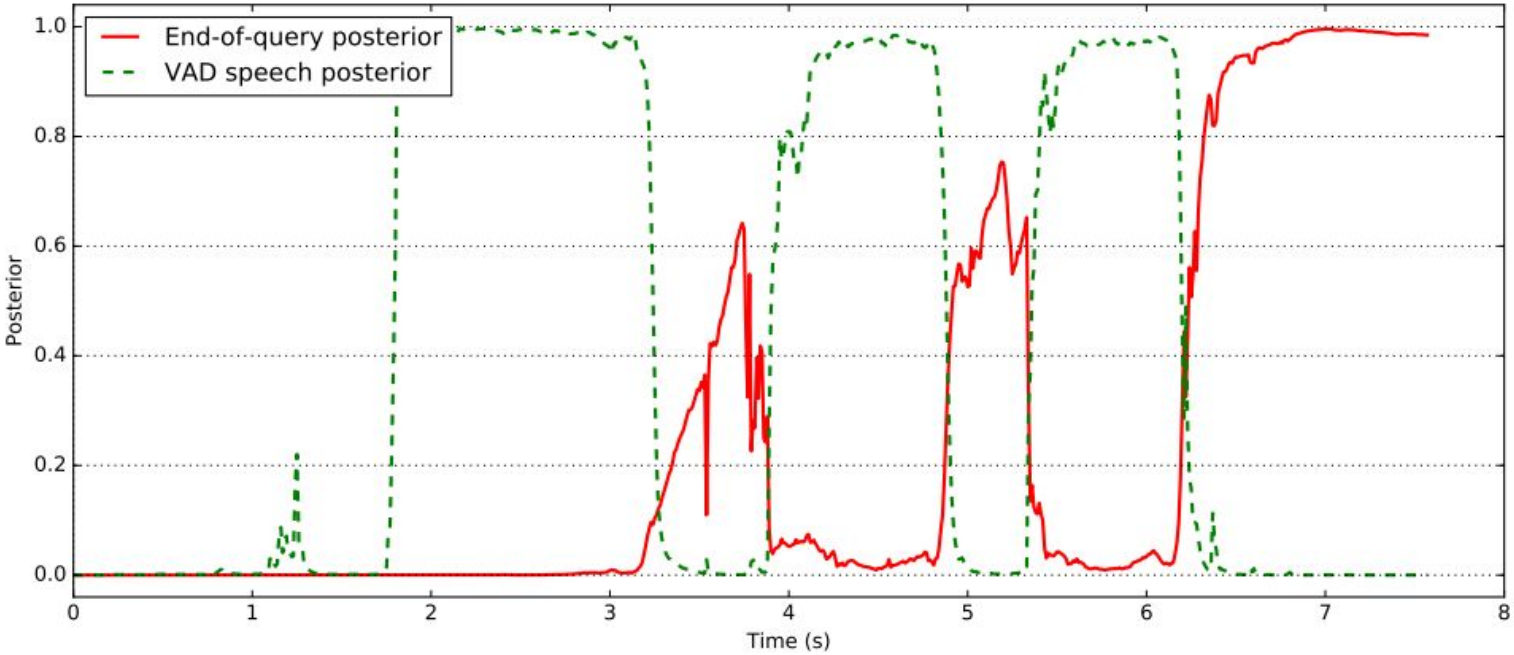
End-of-query detector

- Labels:
  - Speech (0)
  - intermediate non-speech (1) - between words
  - Initial non-speech (2)
  - **Final non-speech (end-of-query)** (3)



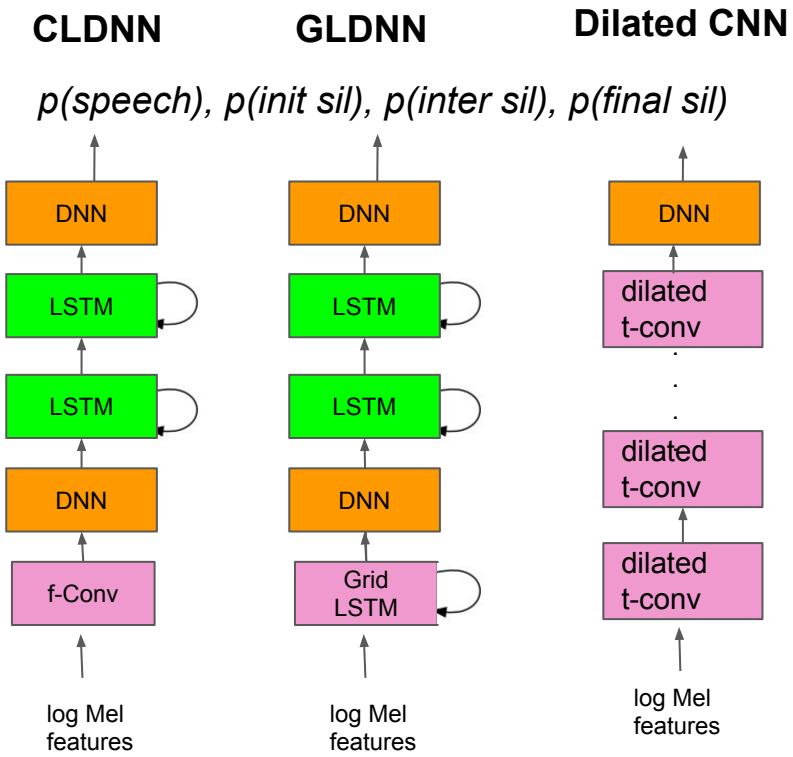
**Improved end-of-query detection for streaming speech recognition*
M. Shannon, G. Simko, S. Chang, and C. Parada Interspeech 2017

EOQ vs VAD



Endpointer Models

- LSTM based models [Chang et al., 2017]
  - Conv/Grid LSTM
  - Bottleneck layer
  - 2 lstm layers
  - 1 dnn layer
  - 1 softmax output layer
- Dilated CNN (WaveNet) [Chang et al., 2018]
  - temporal modeling with CNN



Summary

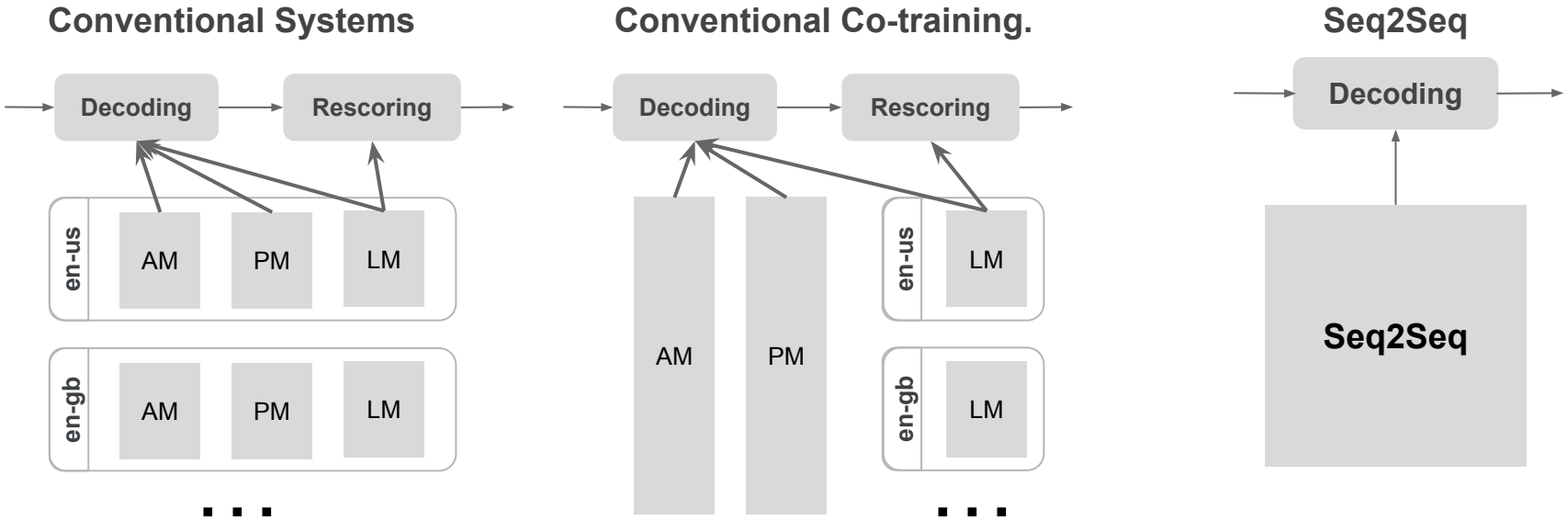
- VAD to EOQ: ~150 ms improvement
- Models: ~100 ms improvement
- More than acoustic information: E2E endpointer

Multi-Dialect ASR

Dialects

- British English (en-gb) vs American English (en-us)
    - Acoustic variation:
        - Cut: (en-gb): **/kʌt/** (en-us): **/kʌ/**
        - Work: (en-gb) **/wɜːk/**,(en-us): **/wɜ˞rk/**
    - Spelling variation:
        - (en-gb): centre, (en-us): center
        - (en-gb): colour, (en-us): color

E2E multi-dialect ASR



In conventional systems, languages/dialects, are handled with **individual AMs, PMs and LMs**. Upscaling is becoming challenging.

A single model for all.

Multi-Dialect LAS

- Modeling Simplicity
- Data Sharing
  - among dialects and model components
- Joint Optimization
- Infrastructure Simplification
  - a single model for all

Table: Resources required for building each system.

Conventional	Seq2Seq
data phoneme lexicon text normalization LM × N	data

Motivations

- We share the same interest:
  - *S. Watanabe, T. Hori, J.R. Hershey*; **Language independent end-to-end architecture for joint language identification and speech recognition**; ASRU 2017. MERL, USA.
    - English, Japanese, Mandarin, German, Spanish, French, Italian, Dutch, Portuguese, Russian.
  - *S. Kim, M.L. Seltzer*; **Towards language-universal end-to-end speech recognition**; submitted to ICASSP 2018. Microsoft, USA.
    - English, German, Spanish.

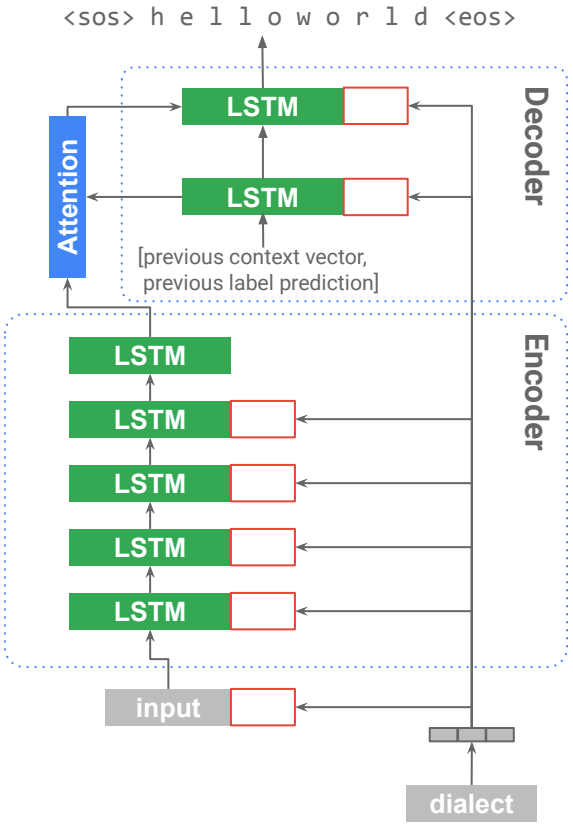
Dialect as Output Targets

- Multi-Task Learning: Joint Language ID (LID) and ASR
  - LID first, then ASR
    - "<sos> **<en-gb>** h e l l o ␣ w o r l d <eos>"
    - LID errors may affect ASR performance
  - ASR first, then LID
    - "<sos> h e l l o ␣ w o r l d **<en-gb>** <eos>"
    - ASR prediction is not dependent on LID prediction, not suffering from LID errors

Dialect as Input Features

- Passing the dialect information as additional features

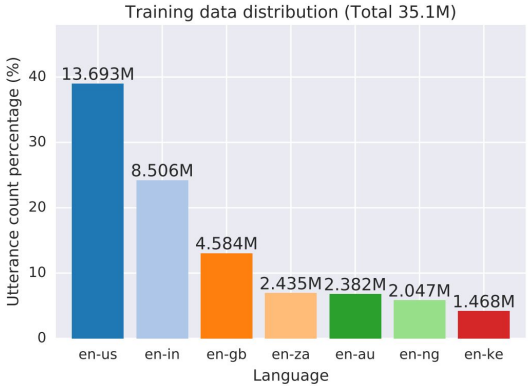
components		variations
encoders	→	acoustic
decoders	→	lexicon and language



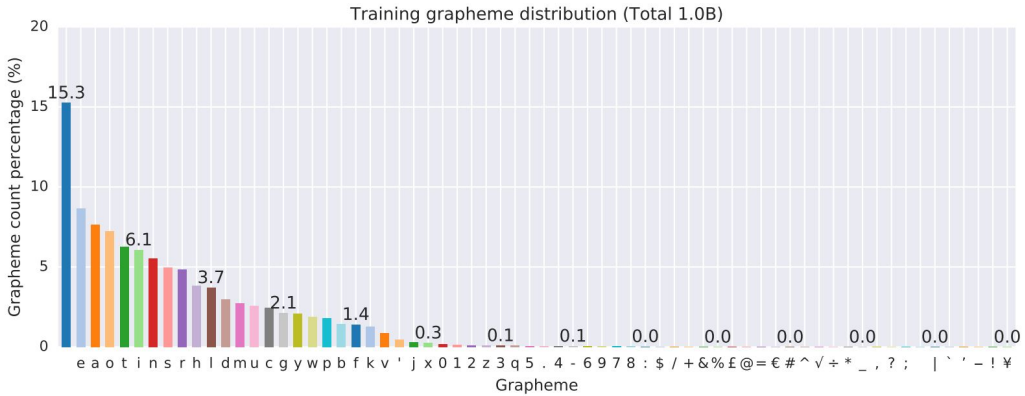
Experimental Evaluations

Task

- **7 English dialects:** US (America), IN (India), GB (Britain), ZA (South Africa), AU (Australia), NG (Nigeria & Ghana), KE (Kenya)



★ **unbalanced** dialect data



★ **unbalanced** target classes

LAS Co-training Baselines

Dialect	US	IN	GB	ZA	AU	NG	KE
dialect-ind.	10.6	18.3	12.9	12.7	12.8	33.4	19.2
dialect-dep.	**9.7**	**16.2**	**12.7**	**11.0**	**12.1**	33.4	**19.0**

★ dialect specific **fine-tuning still wins**

★ simply pooling the data is **missing** certain dialect specific variations

LAS With Dialect as Output Targets

Dialect	US	IN	GB	ZA	AU	NG	KE
Baseline (dialect-dep.)	9.7	**16.2**	12.7	11.0	12.1	33.4	19.0
LID first	9.9	16.6	12.3	11.6	12.2	33.6	18.7
ASR first	**9.4**	16.5	**11.6**	**11.0**	**11.9**	**32.0**	**17.9**

★ LID error **affects** ASR

★ **ASR first** is better

	Example target sequence
LID first	<sos> <en-gb> h e l l o ␣ w o r l d <eos>
ASR first	<sos> h e l l o ␣ w o r l d <en-gb> <eos>

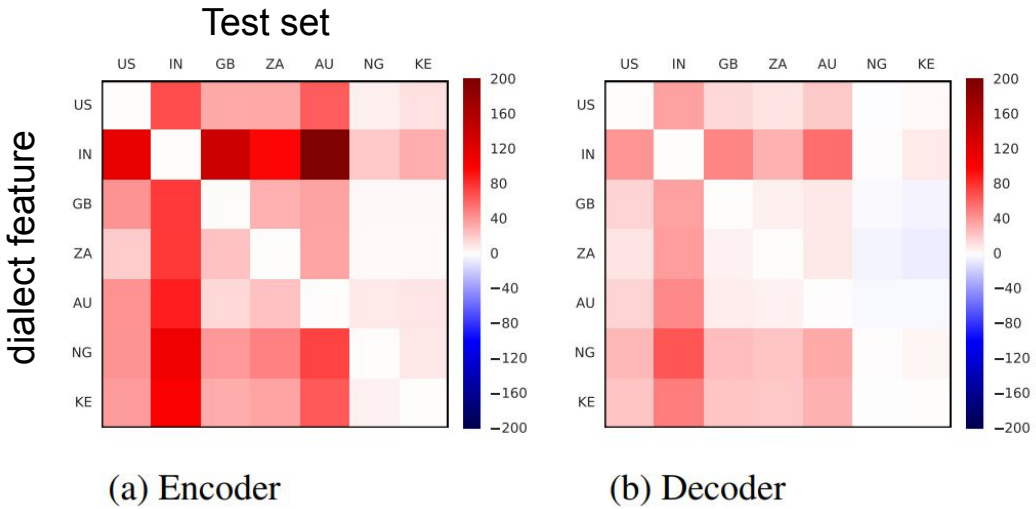
LAS With Dialect as Input Features

Dialect	US	IN	GB	ZA	AU	NG	KE
Baseline (dialect-dep.)	9.7	16.2	12.7	11.0	12.1	33.4	19.0
encoder	9.6	16.4	11.8	10.6	10.7	31.6	18.1
decoder	9.4	16.2	**11.3**	10.8	10.9	32.8	18.0
both	**9.1**	**15.7**	11.5	**10.0**	**10.1**	**31.3**	**17.4**

★ feeding dialect to **both encoder and decoder** gives the largest gains

LAS With Dialect as Input Features

Feeding different dialect vectors (rows) on different test sets (columns).



★ **encoder** is more sensitive to wrong dialects → large acoustic variations

★ for **low-resource** dialects (NG, KE), the model **learns to ignore** the dialect information

LAS With Dialect as Input Features

- The dialect vector does **both AM and LM adaptation**

*Table: The number of **color**/**colour** occurrences in hypotheses on the **en-gb** test data.*

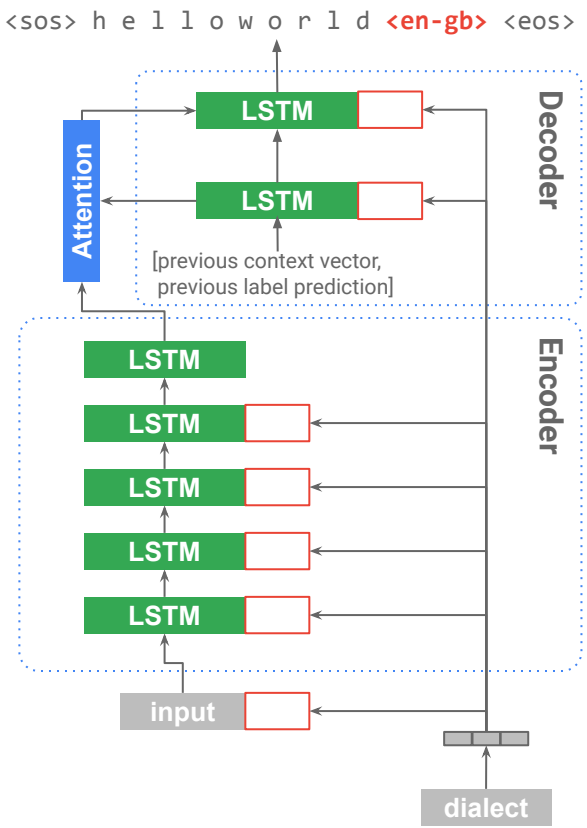
dialect vector	encoder	decoder	color (US)	colour (GB)
❌	❌	❌	1	22
<en-gb>: [0, 1, 0, 0, 0, 0, 0]	✓	❌	19	4
<en-gb>: [0, 1, 0, 0, 0, 0, 0]	❌	✓	0	25
<en-us>: [1, 0, 0, 0, 0, 0, 0]	❌	✓	24	0

★ dialect vector helps **encoder** to **normalize accent variations**

★ dialect vector helps **decoder** to **learn dialect-specific lexicons**

Final Multi-Dialect LAS

- output targets:
  - multi-task with ASR first
- input features:
  - feeding dialect to both encoder and decoder



Final Multi-Dialect LAS

Dialect	US	IN	GB	ZA	AU	NG	KE
Baseline (dialect-dep.)	9.7	16.2	12.7	11.0	12.1	33.4	19.0
output targets (ASR first)	9.4	16.5	11.6	11.0	11.9	32.0	17.9
input features (both)	**9.1**	**15.7**	11.5	10.0	**10.1**	**31.3**	**17.4**
final	**9.1**	16.0	**11.4**	**9.9**	10.3	31.4	17.5

★ **small gains** when combining input and output

★ the final system **outperforms** the dialect-dependent models by 3.1~16.5% relatively

Summary

Attention

- Attention-based end-to-end model achieves state-of-the-art performance on Google's Voice Search and Librispeech tasks.
  - wordpiece output targets
  - multihead attention
  - MWER optimization
  - synchronous training
  - scheduled sampling
  - label smoothing

Online Models

- Recurrent Neural Network Transducer (RNNT)
- Neural Transducer (NT)
- Monotonic Chunkwise Attention (MoChA)

Production

- Personalization (Biasing)
- Endpointer
- Multi-dialect/lingual

Open Questions

Question 1

How to handle long-tail problems?
Rare words/phrases, numeric entities etc have very low or no appearance in the training data. How can we make the end-to-end model recognize them correctly?

Question 2

Can we push the "end" further?
Can end-to-end model be built to directly transform speech to semantics, intents or actions?

Lingvo (tensorflow/lingvo)

A toolkit suited to build neural networks, particularly sequence models.

Machine Translation

Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.

... ...

Speech Recognition

State-of-the-art Speech Recognition With Sequence-to-Sequence Models.

... ...

Speech Synthesis

Hierarchical Generative Modeling for Controllable Speech Synthesis.

... ...

Language Understanding

Semi-Supervised Learning for Information Extraction from Dialogue.

... ...

Source: https://github.com/tensorflow/lingvo

Thank You

References

[Audhkhasi et al., 2017] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, D. Nahamoo “Direct Acoustics-to-Word Models for English Conversational Speech Recognition,” Proc. of Interspeech, 2017.

[Bahdanau et al., 2017] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, Y. Bengio, “An Actor-Critic Algorithm for Sequence Prediction,” Proc. of ICLR, 2017.

[Battenberg et al., 2017] E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, D. Seetapun, A. Sriram, Z. Zhu, “Exploring Neural Transducers For End-to-End Speech Recognition,” Proc. of ASRU, 2017.

[Chan et al., 2015] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, “Listen, Attend and Spell,” CoRR, vol. abs/1508.01211, 2015.

[Chang et al., 2017] S-Y. Chang, B. Li, T. N. Sainath, G. Simko, C. Parada, “Endpoint Detection using Grid Long Short-Term Memory Networks for Streaming Speech Recognition,” Proc. of Interspeech, 2017.

[Chang et al., 2018] S-Y. Chang, B. Li, G. Simko, T. N. Sainath, A. Tripathi, A. van den Oord, O. Vinyals, “Temporal Modeling Using Dilated Convolution and Gating for Voice-Activity-Detection,” Proc. of ICASSP, 2018.

[Chiu and Raffel, 2017] C.-C. Chiu, C. Raffel, “Monotonic Chunkwise Alignments,” Proc. of ICLR, 2017.

[Chiu et al., 2018] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, M. Bacchiani, “State-of-the-art Speech Recognition With Sequence-to-Sequence Models,” Proc. of ICASSP, 2018.

[Chorowski et al., 2015] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-Based Models for Speech Recognition,” in Proc. of NIPS, 2015.

[Graves et al., 2006] A. Graves, S. Fernandez, F. Gomez, J. Schmidhuber, “Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks,” Proc. of ICML, 2006.

[Graves, 2012] A. Graves, “Sequence Transduction with Recurrent Neural Networks,” Proc. of ICML Representation Learning Workshop, 2012.

References

[Graves et al., 2013] A. Graves, A. Mohamed, and G. Hinton, “Speech Recognition with Deep Neural Networks,” in Proc. ICASSP, 2013.

[Gulcehre et al., 2015] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, Y. Bengio, “On Using Monolingual Corpora in Neural Machine Translation”, CoRR, vol. abs/1503.03535, 2015.

[Hannun et al., 2014] A. Hannun, A. Maas, D. Jurafsky, A. Ng, “First-Pass Large Vocabulary Continuous Speech Recognition using Bi-Directional Recurrent DNNs,” CoRR, vol. abs/1408.2873, 2014.

[He et al., 2017] Y. He, R. Prabhavalkar, K. Rao, W. Li, A. Bakhtin and I. McGraw, "Streaming small-footprint keyword spotting using sequence-to-sequence models," Proc. of ASRU, 2017.

[He et al., 2018] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S-Y. Chang, K. Rao, A. Gruenstein, “Streaming End-to-end Speech Recognition For Mobile Devices,” CoRR, vol. abs/1811.06621, 2018.

[Jaitly et al., 2016] N. Jaitly, D. Sussillo, Q. V. Le, O. Vinyals, I. Sutskever, S. Bengio, “An Online Sequence-to-Sequence Model Using Partial Conditioning,” Proc. of NIPS, 2016.

[Kannan et al., 2018] A.Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," Proc. of ICASSP, 2018.

[Kim and Rush, 2016] Y. Kim and A. M. Rush, “Sequence-level Knowledge Distillation,” Proc. of EMNLP, 2016.

[Kim et al., 2017] S. Kim, T. Hori and S. Watanabe, “Joint CTC-attention based End-to-End Speech Recognition using Multi-Task Learning,” Proc. of ICASSP, 2017.

[Kingsbury, 2009] B. Kingsbury, “Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling,” Proc. of ICASSP, 2009.

[Li et al., 2018] B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y. Wu, K. Rao, “Multi-Dialect Speech Recognition With A Single Sequence-To-Sequence Model,” Proc. of ICASSP, 2018.

References

[Maas et al., 2015] A. Maas, Z. Xie, D. Jurafsky, A. Ng, “Lexicon-Free Conversational Speech Recognition with Neural Networks," Proc. of NAACL-HLT, 2015.

[McGraw et al., 2016] I. McGraw, R. Prabhavalkar, R. Alvarez, M. G. Arenas, K. Rao, D. Rybach, O. Alsharif, H. Sak, A. Gruenstein, F. Beaufays, C. Parada, “Personalized speech recognition on mobile devices”, Proc. of ICASSP, 2016

[Rabiner, 1989] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” Proc. of IEEE, 1989.

[Prabhavalkar et al., 2017] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, N. Jaitly, “A Comparison of Sequence-to-Sequence Models for Speech Recognition,” Proc. of Interspeech, 2017.

[Prabhavalkar et al., 2018] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.-C. Chiu, A. Kannan, “Minimum Word Error Rate Training for Attention-based Sequence-to-Sequence Models,” Proc. of ICASSP, 2018.

[Povey, 2003] D. Povey, “Discriminative Training for Large Vocabulary Speech Recognition”, Ph.D. thesis, Cambridge University Engineering Department, 2003.

[Pundak et al., 2018] G. Pundak, T. N. Sainath, R. Prabhavalkar, A. Kannan, D. Zhao, “Deep context: end-to-end contextual speech recognition,” Proc. of SLT, 2018.

[Rao et al., 2017] K. Rao, H. Sak, R. Prabhavalkar, “Exploring Architectures, Data and Units For Streaming End-to-End Speech Recognition with RNN-Transducer”, Proc. of ASRU, 2017.

[Ranzato et al., 2016] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, “Sequence level training with recurrent neural networks,” Proc. of ICLR, 2016.

[Sainath et al., 2018] T. N. Sainath, C.-C. Chiu, R. Prabhavalkar, A. Kannan, Y. Wu, P. Nguyen, Z. Chen, “Improving the Performance of Online Neural Transducer Models,” Proc. of ICASSP, 2018.

[Sak et al., 2015] Has¸im Sak, Andrew Senior, Kanishka Rao, Francoise Beaufays, “Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition,” Proc. of Interspeech, 2015.

References

[Sak et al., 2017] H. Sak, M. Shannon, K. Rao, and F. Beaufays, “Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping,” in Proc. of Interspeech, 2017.

[Schuster & Nakajima, 2012] M. Schuster and K. Nakajima, “Japanese and Korean Voice Search,” Proc. of ICASSP, 2012.

[Shannon, 2017] M. Shannon, “Optimizing expected word error rate via sampling for speech recognition,” in Proc. of Interspeech, 2017.

[Sim et al., 2017] K. Sim, A. Narayanan, T. Bagby, T. N. Sainath, and M. Bacchiani, “Improving the Efficiency of Forward-Backward Algorithm using Batched Computation in TensorFlow,” Proc. of ASRU, 2017.

[Sriram et al., 2018] A. Sriram, H. Jun, S. Satheesh, A. Coates, “Cold Fusion: Training Seq2Seq Models Together with Language Models,” Proc. of ICLR, 2018.

[Stolcke et al., 1997] A. Stolcke, Y. Konig, M. Weintraub, “Explicit word error minimization in N-best list rescoring,” Proc. of Eurospeech, 1997.

[Su et al., 2013] H. Su, G. Li, D. Yu, and F. Seide, “Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription,” Proc. of ICASSP, 2013.

[Szegedy et al., 2016] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” Proc. of CVPR, 2016.

[Toshniwal et al., 2018] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. Moreno, E. Weinstein, K. Rao, “Multilingual Speech Recognition With A Single End-To-End Model,” Proc. of ICASSP, 2018.

[Vaswani et al., 2017] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” Proc. of NIPS, 2017.

[Wiseman and Rush, 2016] S. Wiseman and A. M. Rush, “Sequence-to-Sequence Learning as Beam Search Optimization,” Proc. of EMNLP, 2016.