

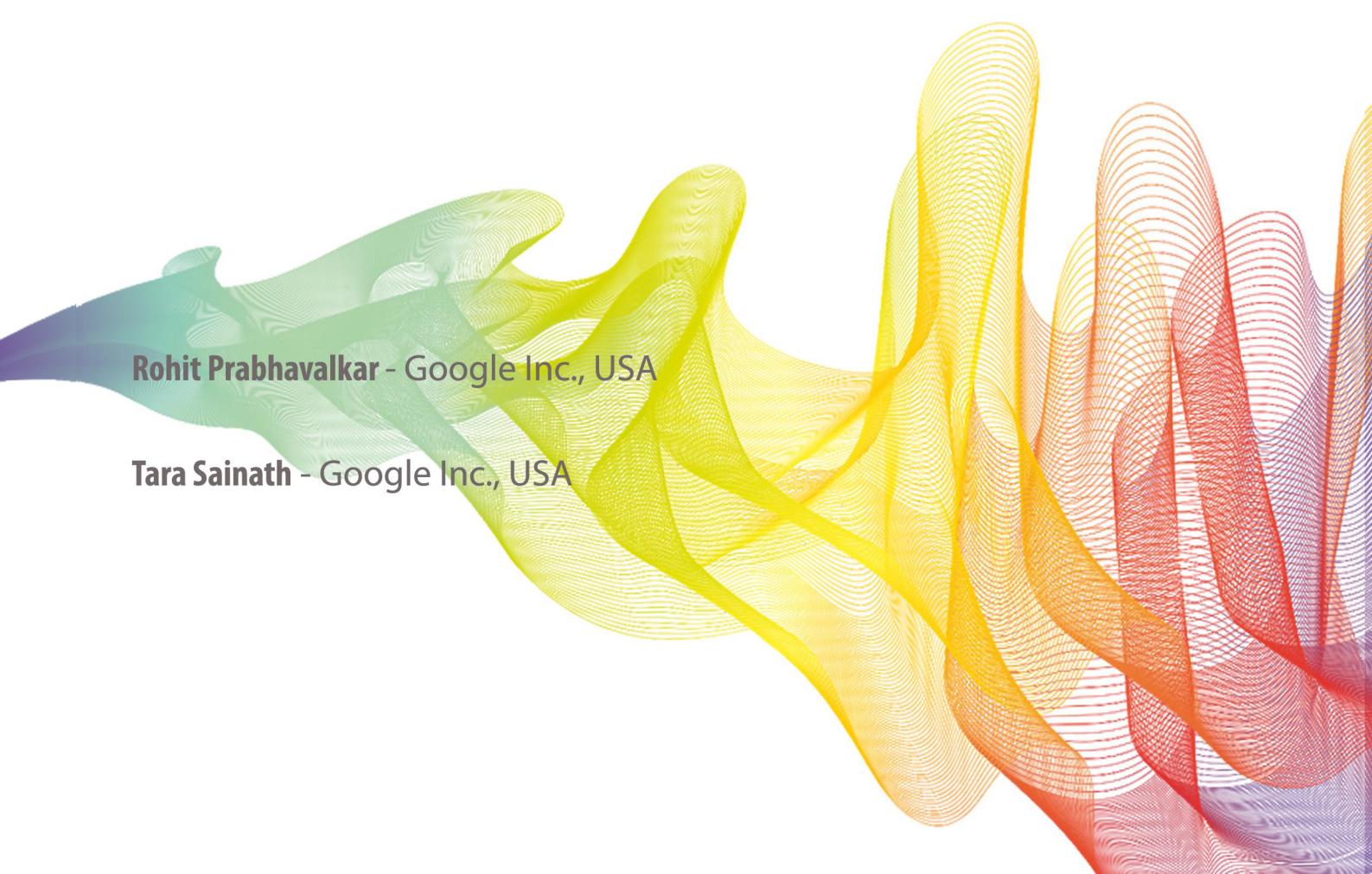


INTERSPEECH 2018
SEPTEMBER 2-6 | HYDERABAD, INDIA
HYDERABAD INTERNATIONAL CONVENTION CENTRE

Tutorial 2 - T2

Sunday Morning, Sep 2, 9:00AM - 12:30PM

End-To-End Models for Automatic Speech Recognition



Rohit Prabhavalkar - Google Inc., USA

Tara Sainath - Google Inc., USA

End-To-End Models for Automatic Speech Recognition

Traditional automatic speech recognition (ASR) systems are comprised of a set of separate components, namely an acoustic model (AM); a pronunciation model (PM); and a language model (LM). The AM takes acoustic features as input and predicts a distribution over subword units, typically context-dependent phonemes. The PM, which is traditionally a hand-engineered lexicon maps the sequence of subword units produced by the acoustic model to words. Finally, the LM assigns probabilities to various word hypotheses. In traditional ASR systems, these components are trained independently on different datasets, with a number of independence assumptions which are made for tractability.

Over the last several years, there has been a growing interest in developing end-to-end systems, which attempt to learn these separate components jointly in a single system. Examples of such system include attention-based models [1, 6], the recurrent neural transducer [2, 3], the recurrent neural aligner [4], and connectionist temporal classification with word targets [5]. A common feature of all of these models is that they are composed on a single neural network, which when given input acoustic frames directly outputs a probability distribution over graphemes or word hypotheses. In fact, as has been demonstrated in recent work, such end-to-end models can surpass the performance of a conventional ASR systems [6].

In this tutorial, we will provide a detailed introduction to the topic of end-to-end modeling in the context of ASR. We will begin by charting out the historical development of these systems, while emphasizing the commonalities and the differences between the various end-to-end approaches that have been considered in the literature. We will then discuss a number of recently introduced innovations that have significantly improved the performance of end-to-end models, allowing these to surpass the performance of conventional ASR systems. The tutorial will then describe some of the exciting applications of this research, along with possible fruitful directions to explore.

Finally, the tutorial will discuss some of the shortcomings of existing end-to-end modeling approaches and discuss ongoing efforts to address these challenges.

- [1] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, Attend and Spell," in Proc. ICASSP, 2016.
- [2] A. Graves, "Sequence transduction with recurrent neural networks," in Proc. of ICASSP, 2012.
- [3] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring Architectures, Data and Units For Streaming End-to-End Speech Recognition with RNN-Transducer," in Proc. ASRU, 2017.
- [4] 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. Interspeech, 2017.
- [5] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: acoustic-to-word LSTM model for large vocabulary speech recognition," in Proc. of Interspeech, 2017.
- [6] 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 and M. Bacchiani, "State-of-the-art Speech Recognition With Sequence-to-Sequence Models," in Proc. ICASSP, 2018.



Rohit Prabhavalkar received the B.E. degree in computer engineering from the University of Pune, India, in 2007 and the M.S. and Ph.D. degrees in computer science and engineering from The Ohio State University, USA, in 2012 and 2013, respectively. He is currently a research scientist at Google, having joined in 2013, where his research focuses on various aspects of acoustic modeling with the goal of improving automatic speech recognition technology. His other research interests include deep neural networks, natural language processing, and machine learning.



Tara Sainath received her PhD in Electrical Engineering and Computer Science from MIT in 2009. The main focus of her PhD work was in acoustic modeling for noise robust speech recognition. After her PhD, she spent 5 years at the Speech and Language Algorithms group at IBM T.J. Watson Research Center, before joining Google Research. She has served as a Program Chair for ICLR in 2017 and 2018. Also, she has co-organized numerous special sessions and workshops, including Interspeech 2010, ICML 2013, Interspeech 2016 and ICML 2017. In addition, she is a member of the IEEE Speech and Language Processing Technical Committee (SLTC) as well as the Associate Editor for IEEE/ACM Transactions on Audio, Speech, and Language Processing. Her research interests are mainly in acoustic modeling, including deep neural networks, sparse representations and adaptation methods.



Towards End-to-End Speech Recognition

Rohit Prabhavalkar and Tara N. Sainath

September 2, 2018

Acknowledgements



Google Brain

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

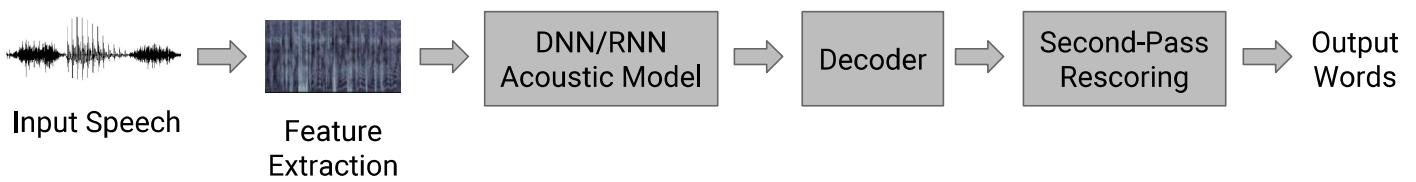
Speech Team

Michiel Bacchiani, Shuo-yiin Chang, Yanzhang He, Shankar Kumar, Bo Li, Golan Pundak, Kanishka Rao, David Rybach, Tara Sainath, Johan Schalkwyk, Vlad Schogol, Gabor Simko, Shubham Toshniwal, Ding Zhao

What is End-to-End ASR?

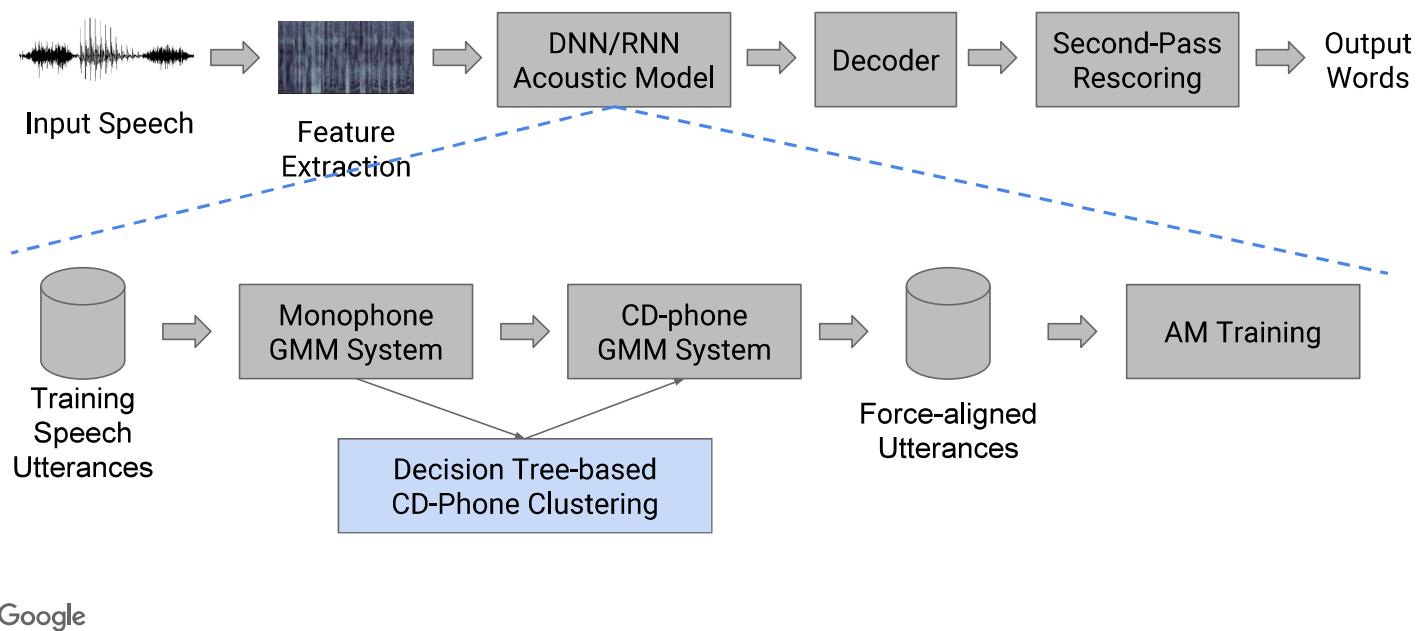
Google

Conventional ASR Pipeline

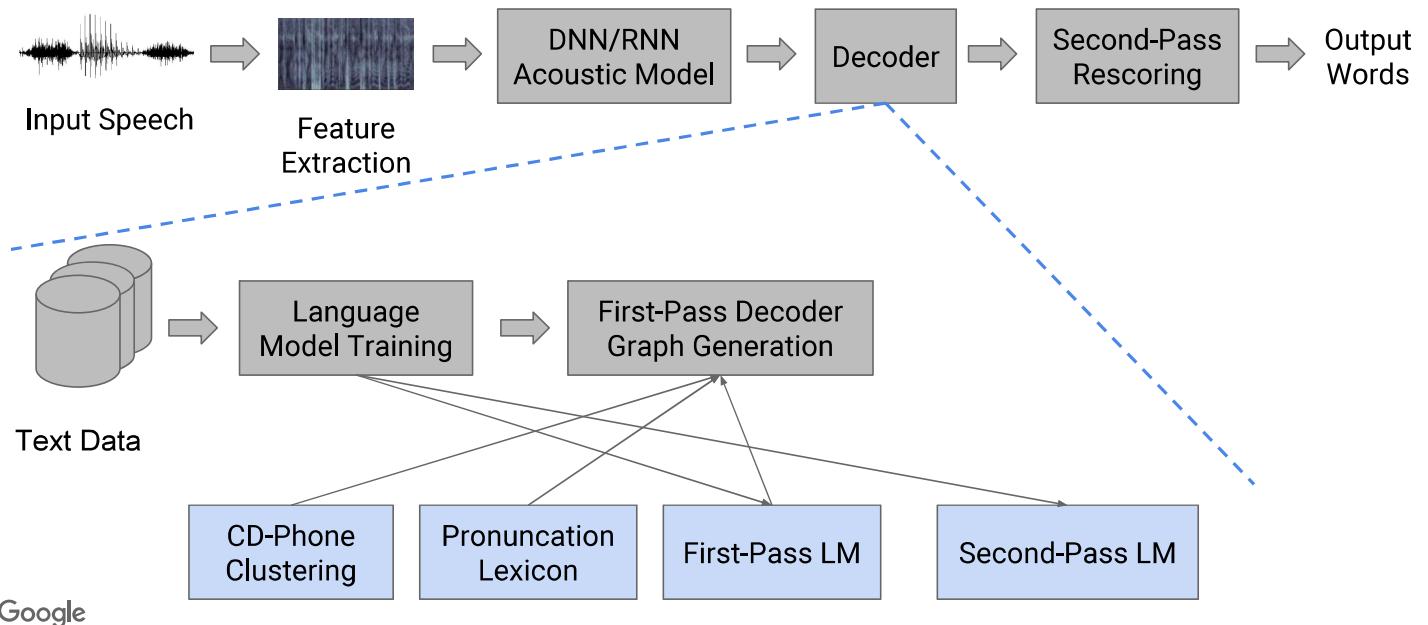


Google

Conventional ASR Pipeline: AM Training



Conventional ASR Pipeline: LM Training



Conventional ASR Pipeline

- Most ASR systems involve separately trained 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

Google

What is “End-to-End” ASR?

Google

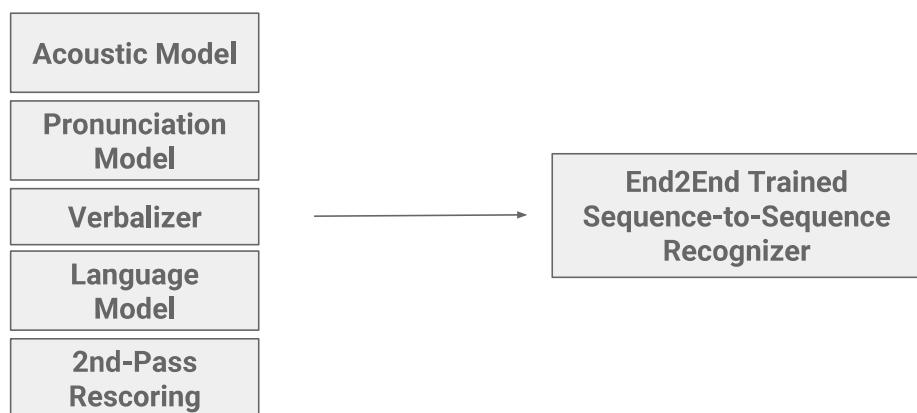
“A system which directly maps a sequence of input acoustic features into a sequence of graphemes or words.”

“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).”

Google

Motivation: End-to-End ASR

Typical Speech System



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

Google

Historical Development of End-to-End ASR

Google

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, word-level lagged behind ASR [Sak et al., 2015]

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

Alex Graves¹
 Santiago Fernández¹
 Faustino Gomez¹
 Jürgen Schmidhuber^{1,2}

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

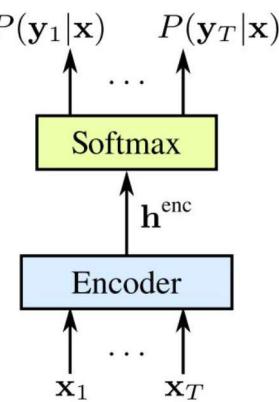
Many real-world sequence learning tasks require the prediction of sequences of labels from noisy, unsegmented input data. In

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

Google

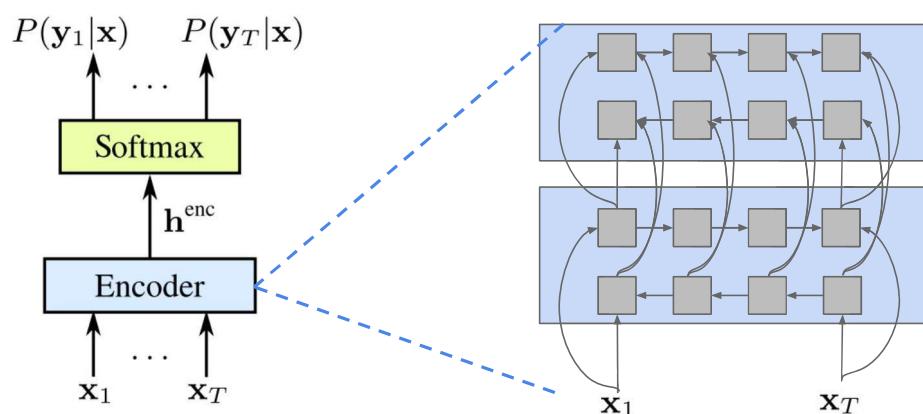
Connectionist Temporal Classification (CTC)



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

Google

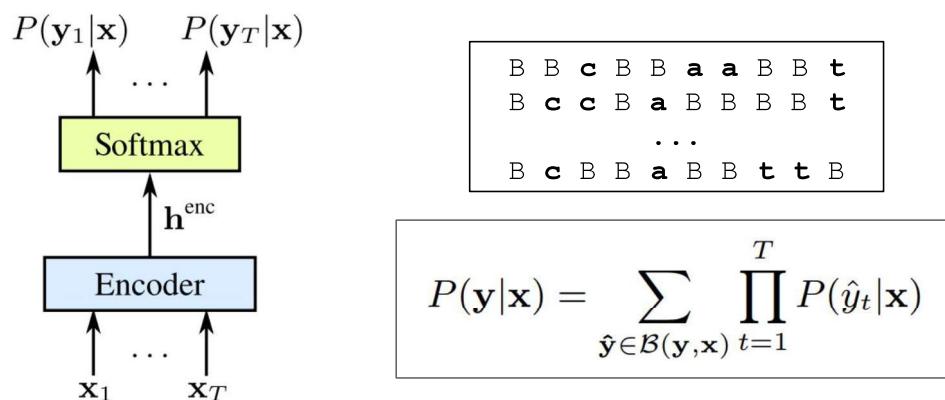
Connectionist Temporal Classification (CTC)



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

Google

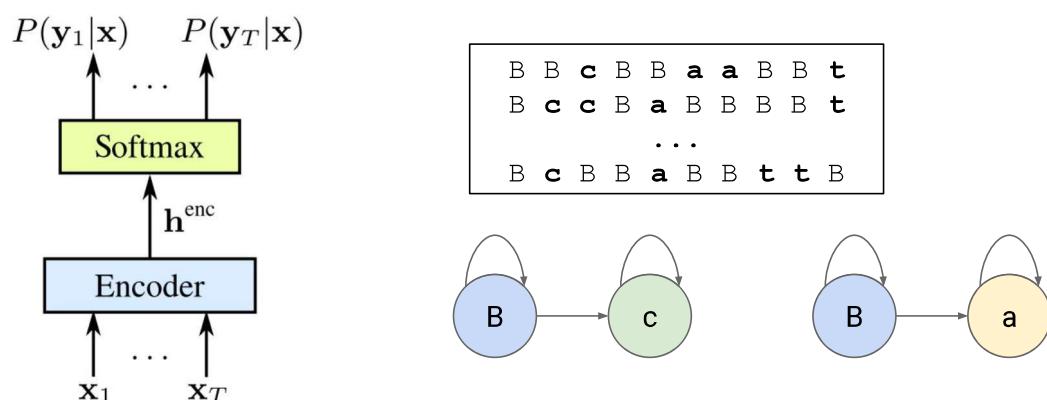
Connectionist Temporal Classification (CTC)



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

Google

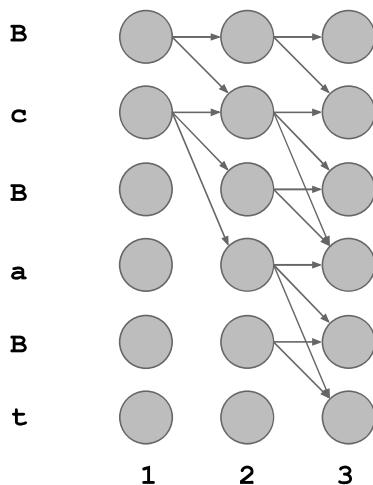
Connectionist Temporal Classification (CTC)



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

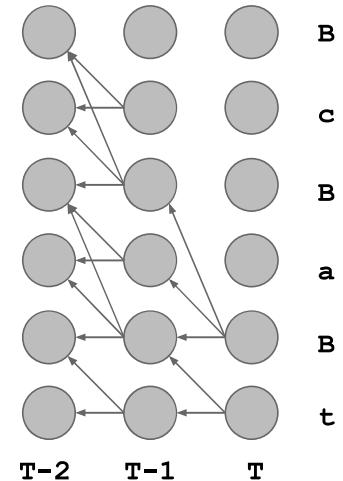
Google

Connectionist Temporal Classification (CTC)



Forward-Backward
Algorithm Computation

Frames, t



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

Google

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 rescoreing a baseline system.
- Also proposed minimizing expected transcription error [WSJ: 8.7% → 8.2%]

Google

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

Alex Graves
Google DeepMind, London, United Kingdom

GRAVES@CS.TORONTO.EDU

Navdeep Jaitly
Department of Computer Science, University of Toronto, Canada

NDJAITLY@CS.TORONTO.EDU

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

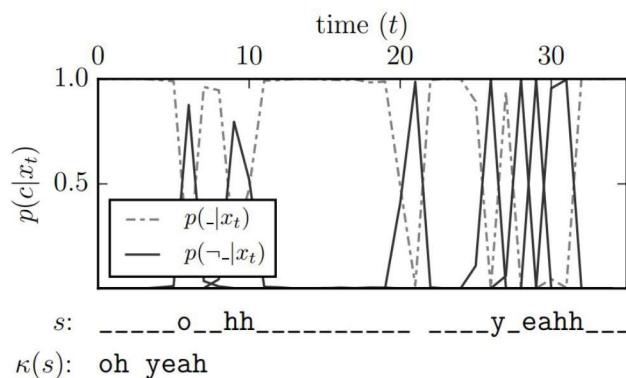
[Graves and Jaitly, 2014] ICML

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
 - [Soltan 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 ...

Google

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

Google

Reproduced from [Maas et al., 2015] NAACL

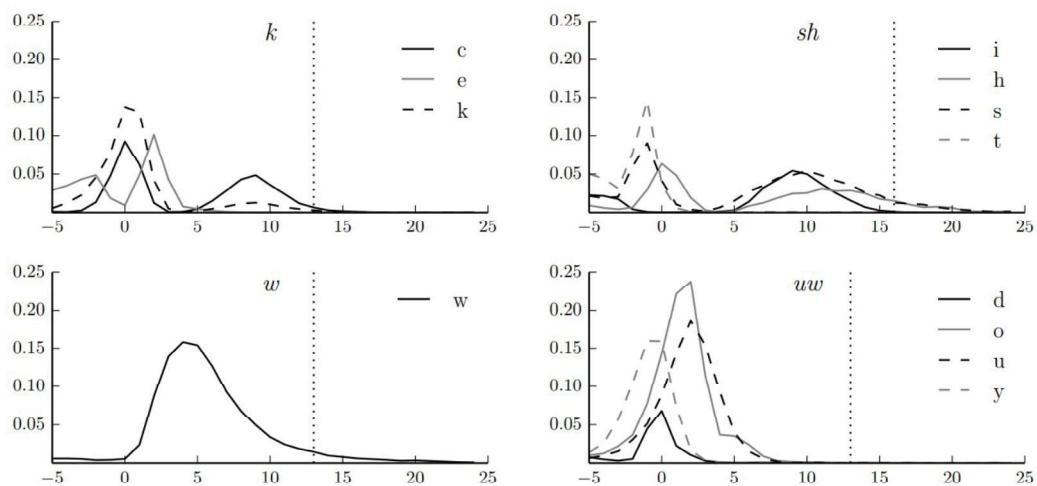
CTC-Based End-to-End ASR

#	Method	Transcription
(1)	Truth	yeah i went into the i do not know what you think of <i>fidelity</i> but
	HMM-GMM	yeah when the i don't know what you think of fidel it even them
	CTC+CLM	yeah i went to i don't know what you think of fidelity but um
(2)	Truth	no no speaking of weather do you carry a altimeter slash <i>barometer</i>
	HMM-GMM	no i'm not all being the weather do you uh carry a uh helped emitters last brahms her
	CTC+CLM	no no beating of whether do you uh carry a uh a time or less barometer
(3)	Truth	i would ima- well yeah it is i know you are able to stay home with them
	HMM-GMM	i would amount well yeah it is i know um you're able to stay home with them
	CTC+CLM	i would ima- well yeah it is i know uh you're able to stay home with them

Google

Reproduced from [Maas et al., 2015] NAACL

CTC-Based End-to-End ASR



Google

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

Google

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

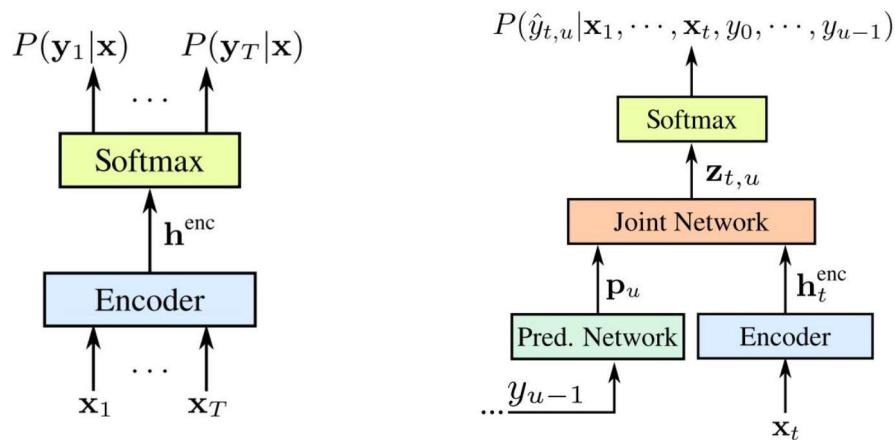
Recurrent neural networks (RNNs) are a powerful model for sequential data. End-to-end training methods such as Connectionist Temporal Classification make it possible to train RNNs for sequence labelling problems where the input-output alignment is unknown. The combination of these methods with

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

Google

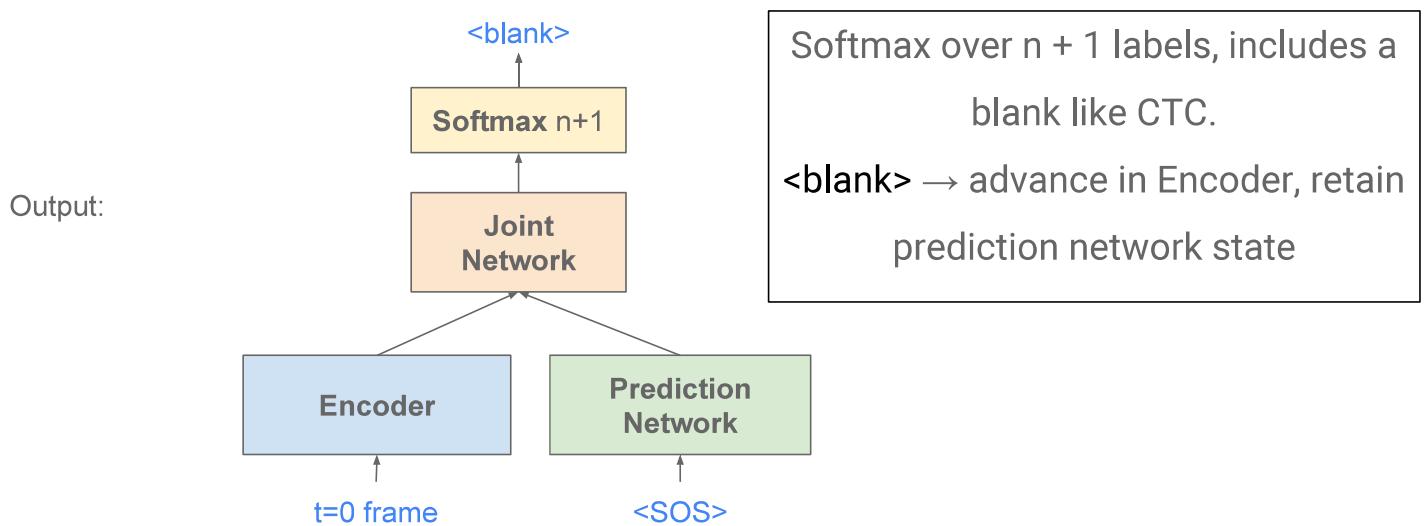
Recurrent Neural Network Transducer (RNN-T)



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

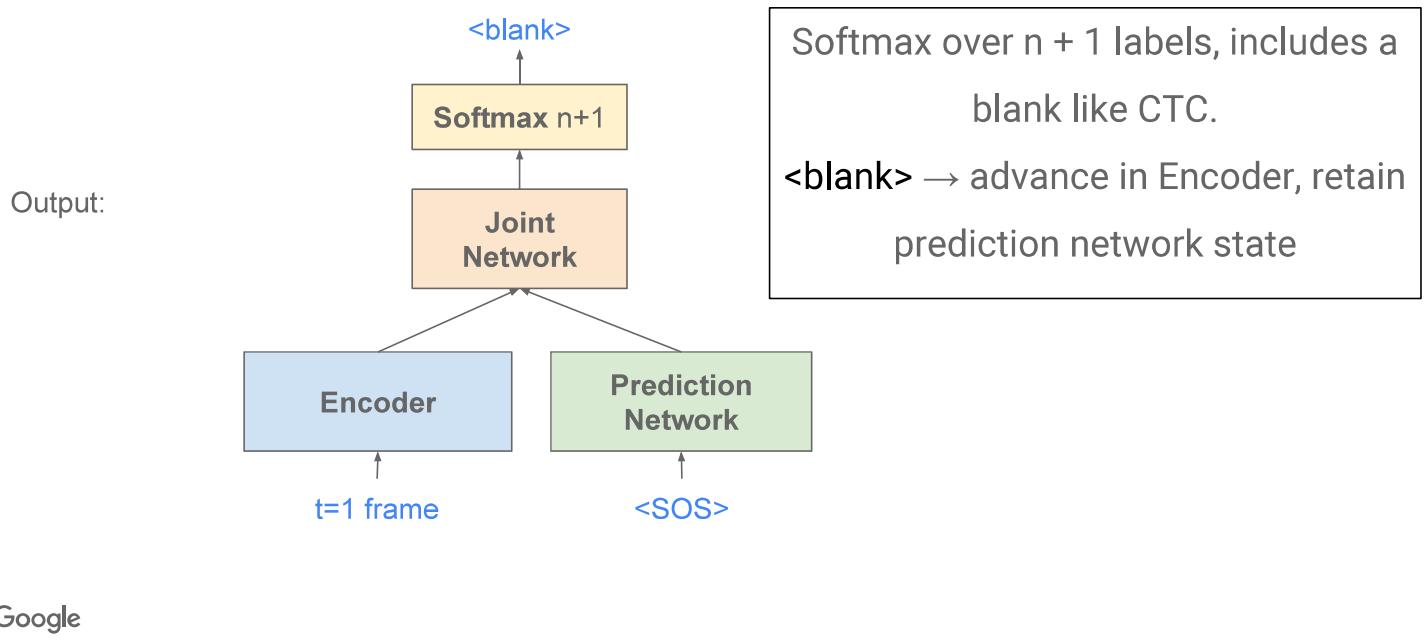
Google

Recurrent Neural Network Transducer (RNN-T)

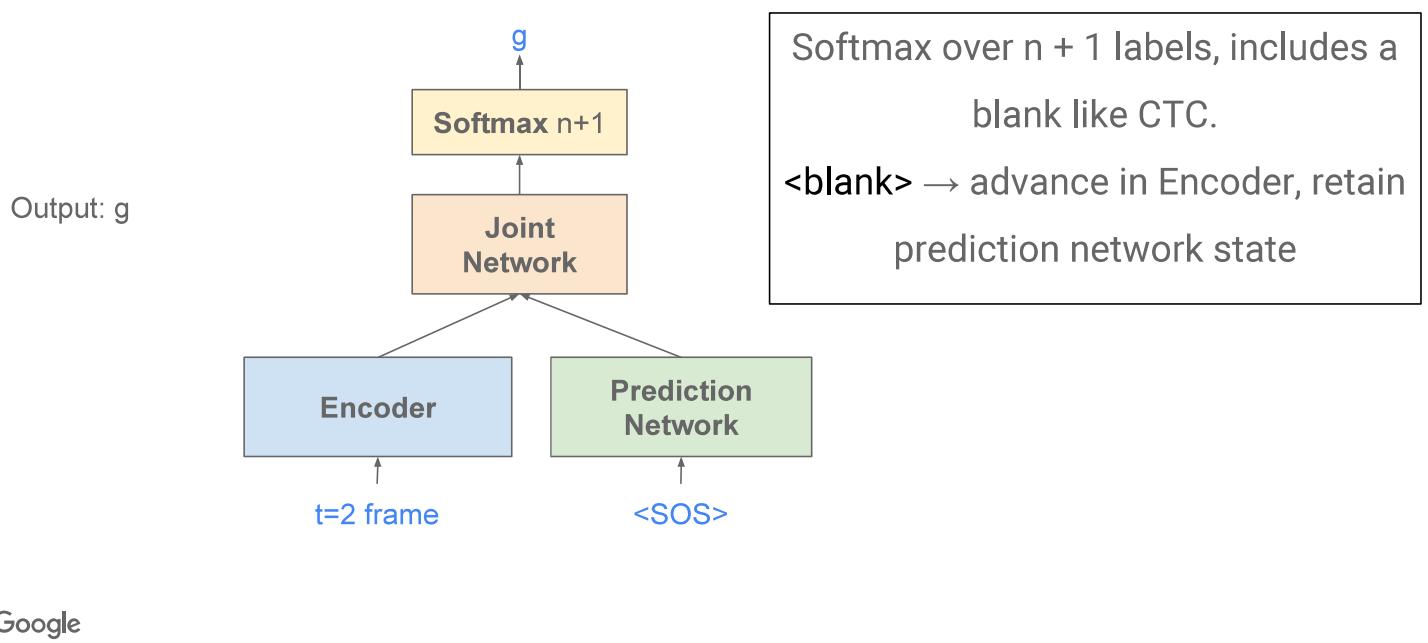


Google

Recurrent Neural Network Transducer (RNN-T)

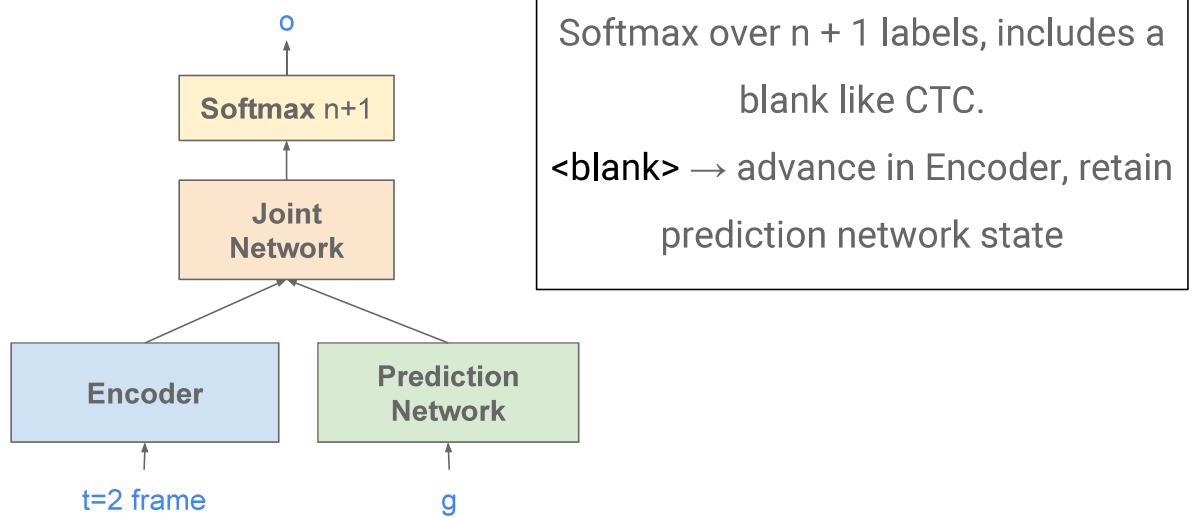


Recurrent Neural Network Transducer (RNN-T)



Recurrent Neural Network Transducer (RNN-T)

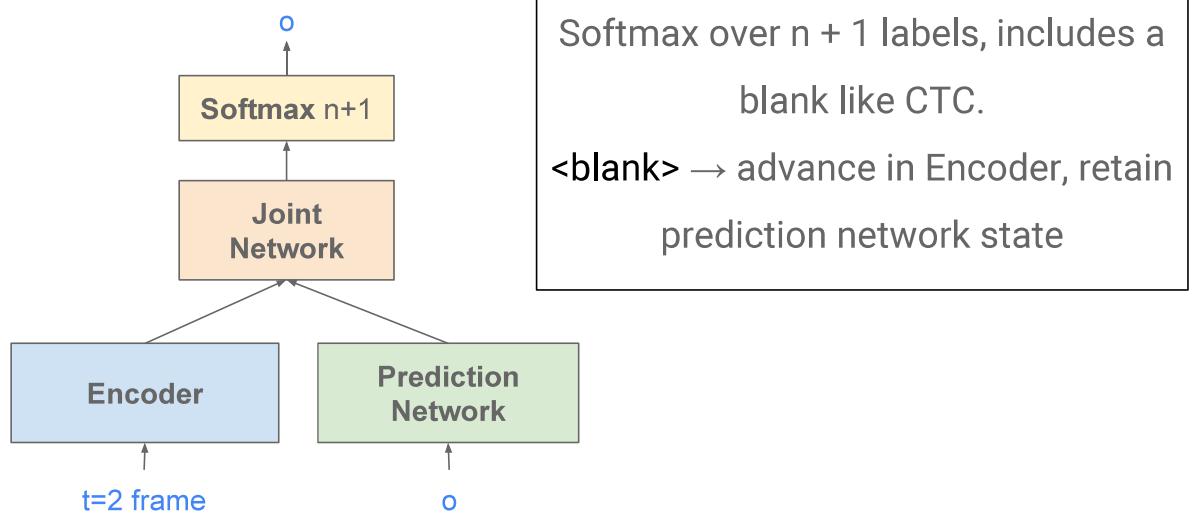
Output: go



Google

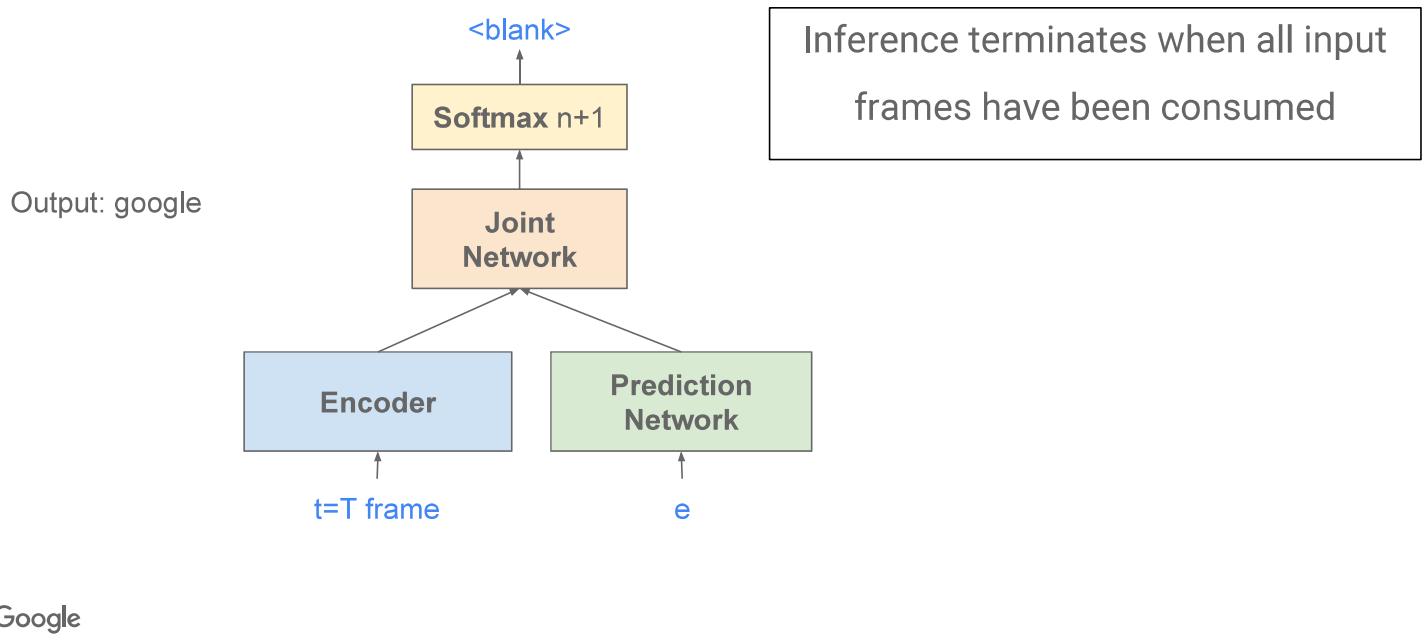
Recurrent Neural Network Transducer (RNN-T)

Output: goo



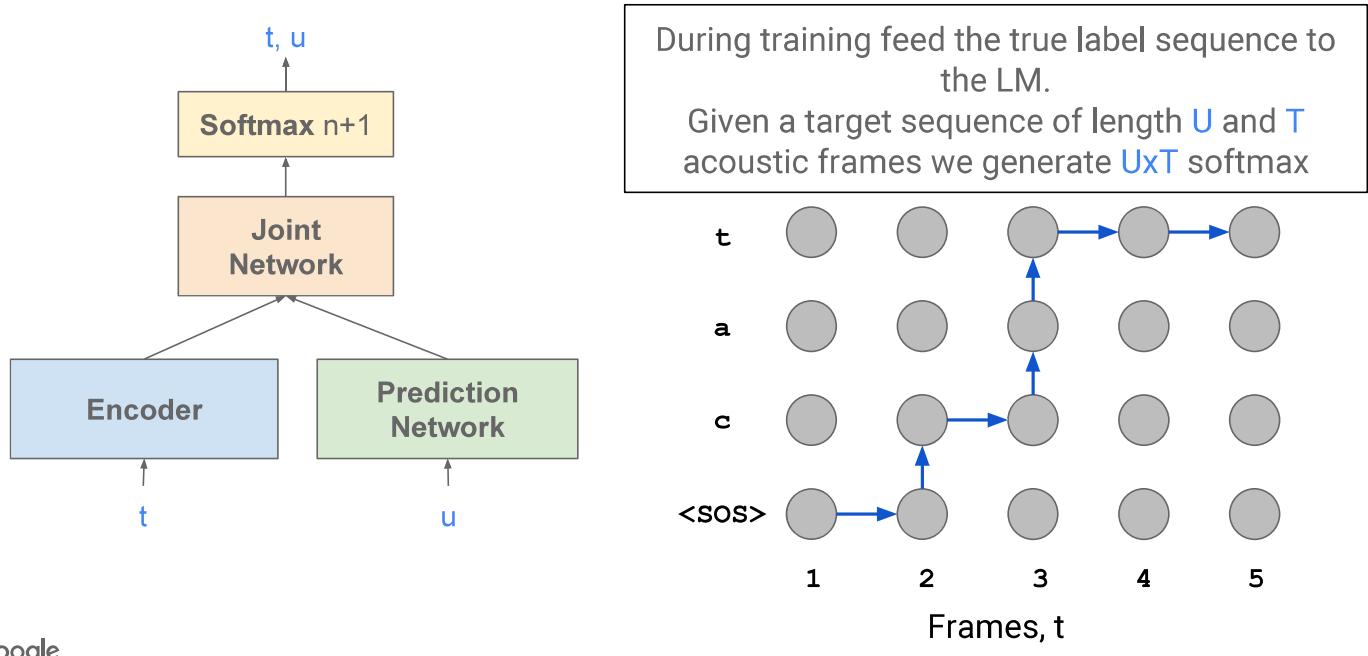
Google

Recurrent Neural Network Transducer (RNN-T)



Google

Recurrent Neural Network Transducer (RNN-T)



Recurrent Neural Network Transducer (RNN-T)

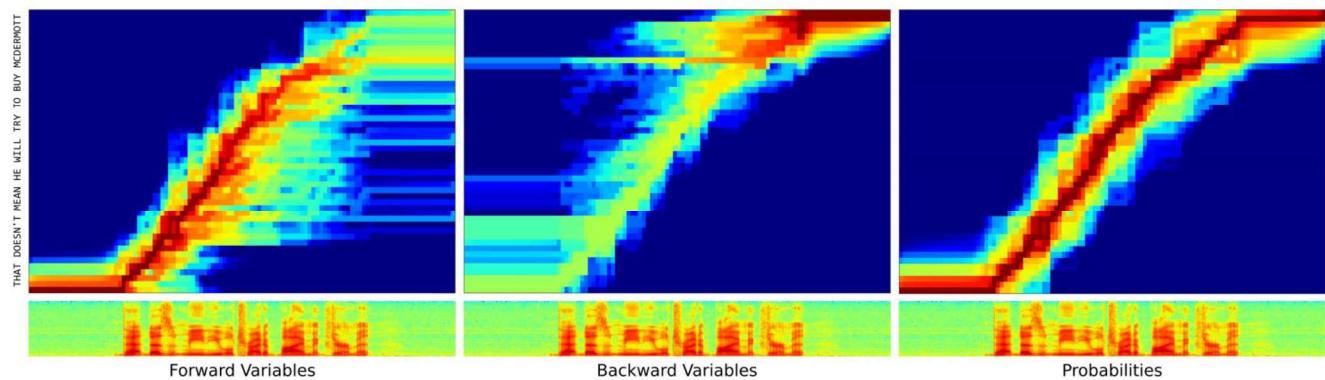


Figure 2. Forward-backward variables during a speech recognition task. The image at the bottom is the input sequence: a spectrogram of an utterance. The three heat maps above that show the logarithms of the forward variables (top) backward variables (middle) and their product (bottom) across the output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

Google

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.

Google

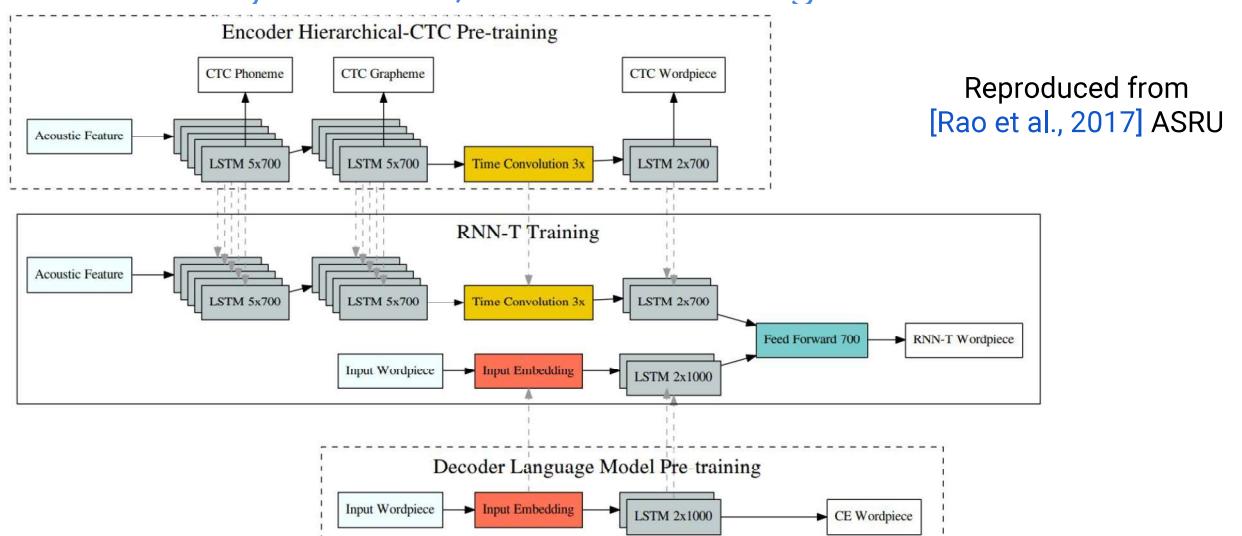
Reproduced from [Graves, 2013] ICASSP

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.

Google

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



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

Google

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

Google

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

Units	Layers		Pre-trained		Training Data Used			WER(%)		
	Encoder	Decoder	Encoder	Decoder	Acoustic	Pronunciation	Text	Params	VS	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%.

Google

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

Units	Layers		Pre-trained		Training Data Used			WER(%)		
	Encoder	Decoder	Encoder	Decoder	Acoustic	Pronunciation	Text	Params	VS	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 “prediction network” (i.e., prediction network) helps improve performance by ~5%.

Google

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

Units	Layers		Pre-trained		Training Data Used			WER(%)		
	Encoder	Decoder	Encoder	Decoder	Acoustic	Pronunciation	Text	Params	VS	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

Google

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

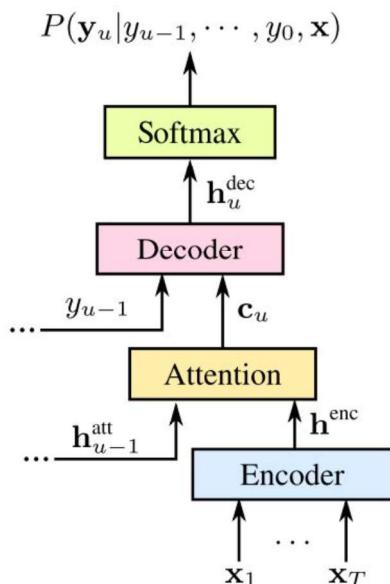
Dmitriy Serdyuk
Université de Montréal
Kyunghyun Cho
Université de Montréal

Yoshua Bengio
Université de Montréal
CIFAR Senior Fellow

[Chorowski et al., 2015]

Google

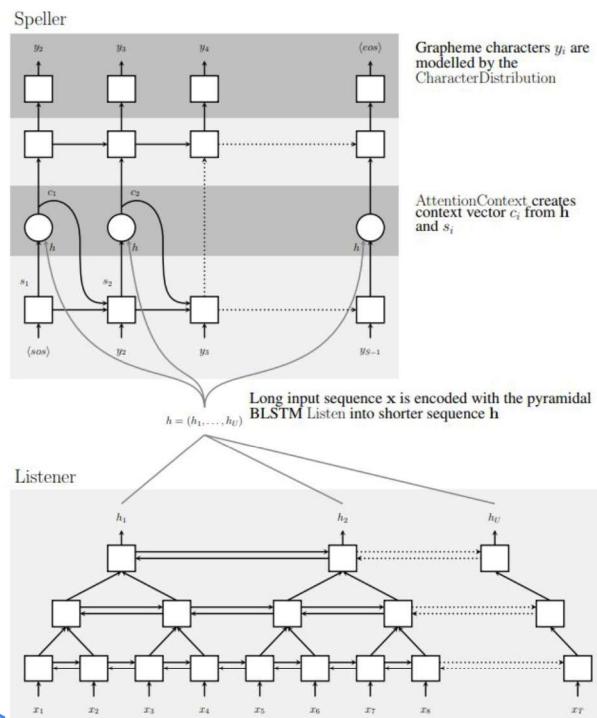
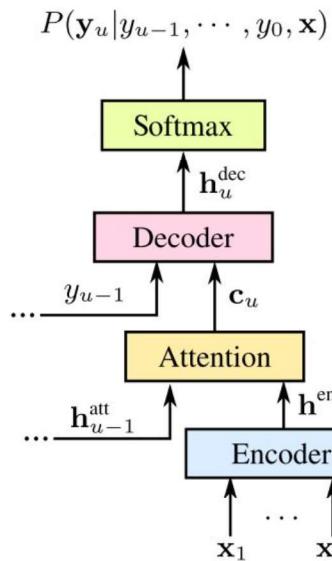
Attention-based Encoder-Decoder Models



- **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

Google

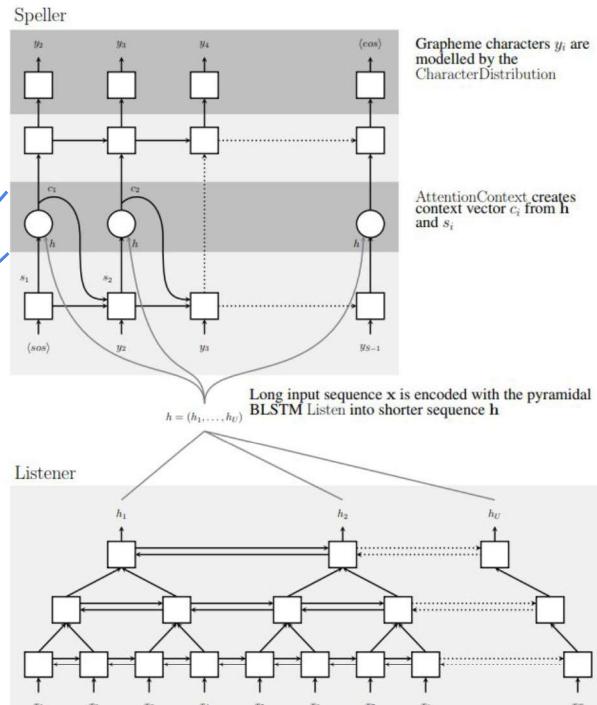
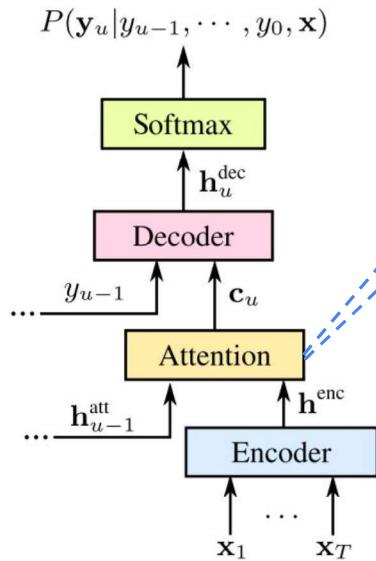
Attention-Based Models



Google

Reproduced from [Chan et al., 2015]

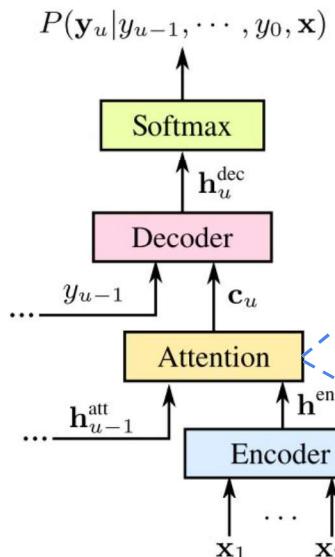
Attention-Based Models



Google

Reproduced from [Chan et al., 2015]

Attention-Based Models



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

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

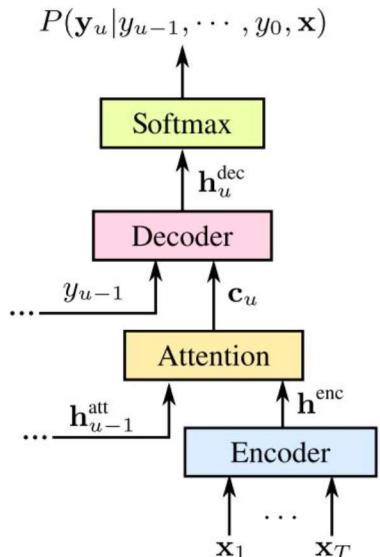
$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=1}^T \exp(e_{u,t'})}$$

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

Google

Attention-Based Models

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

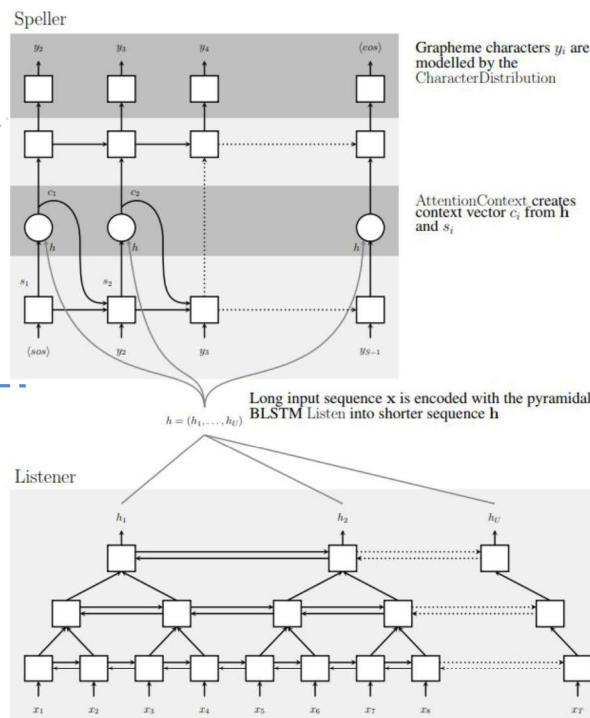
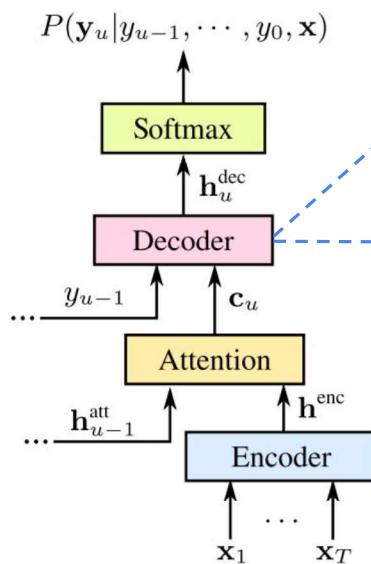


Additive Attention [Chorowski et al., 2015]

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

Google

Attention-Based Models



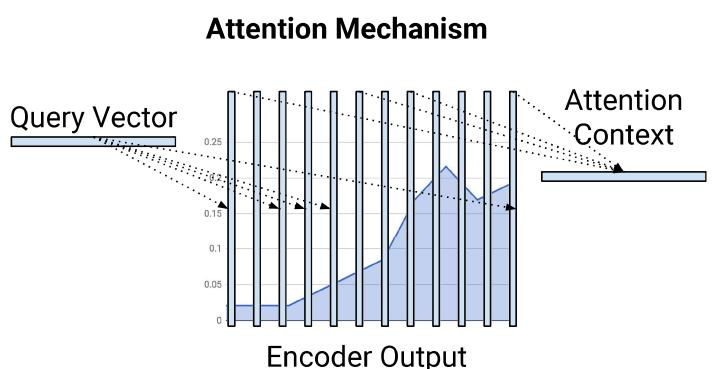
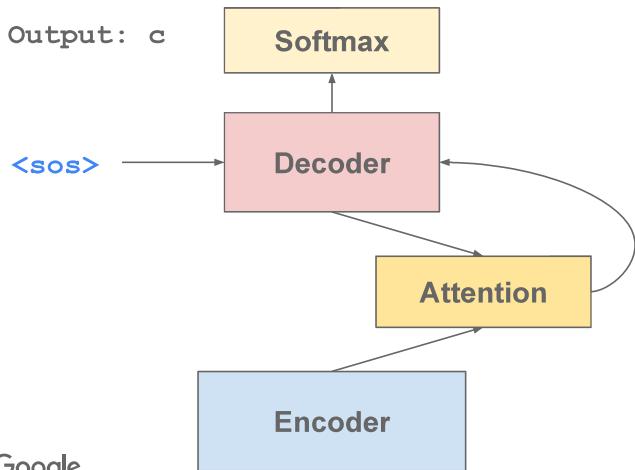
Google

Reproduced from [Chan et al., 2015]

Attention-based Models

$$\begin{aligned} P(a | \text{sos}, x) &= 0.01 \\ P(b | \text{sos}, x) &= 0.01 \\ \mathbf{P(c | \text{sos}, x)} &= 0.92 \\ &\dots \end{aligned}$$

Attention mechanism summarizes encoder features relevant to predict next label



Google

48

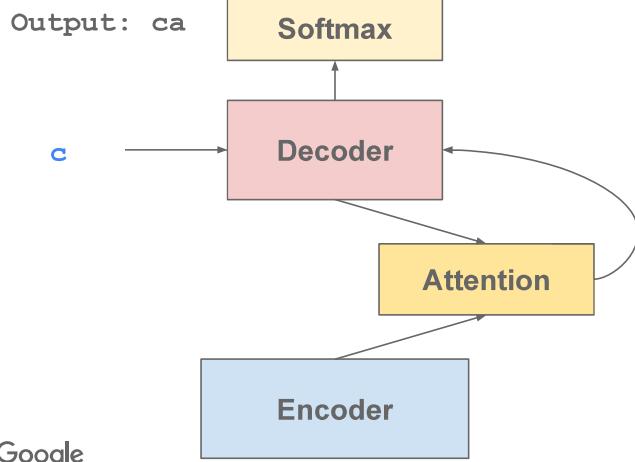
Attention-based Models

$$P(a|c, \text{<sos>, } x) = 0.95$$

$$P(b|c, \text{<sos>, } x) = 0.01$$

$$P(c|c, \text{<sos>, } x) = 0.01$$

...



49

Attention-based Models

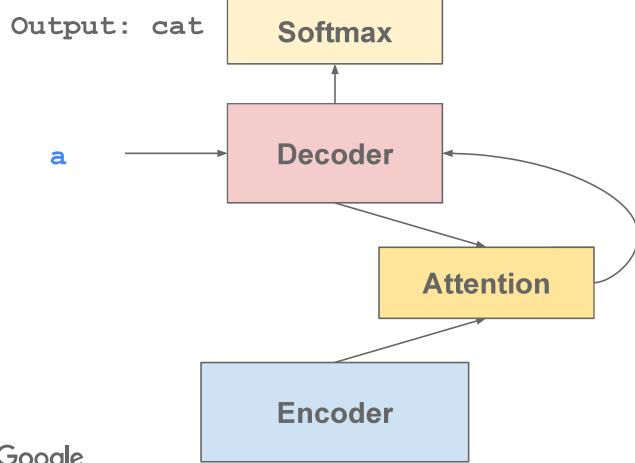
$$P(a|a, c, \text{<sos>, } x) = 0.01$$

$$P(b|a, c, \text{<sos>, } x) = 0.08$$

...

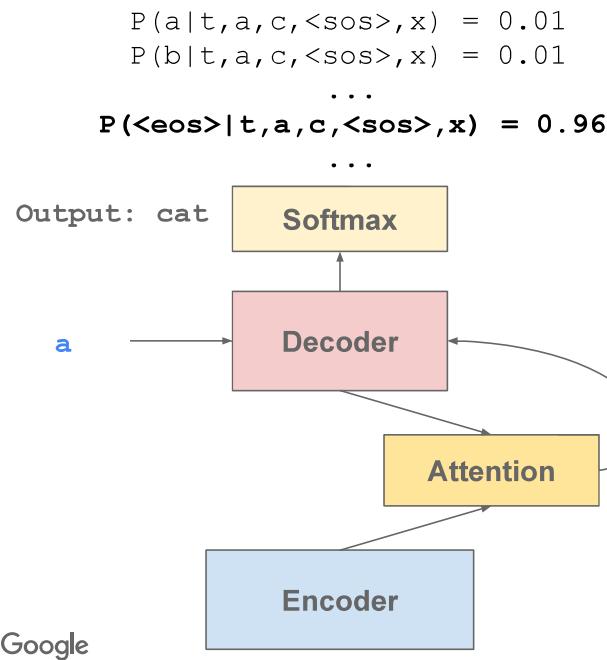
$$P(t|a, c, \text{<sos>, } x) = 0.89$$

...

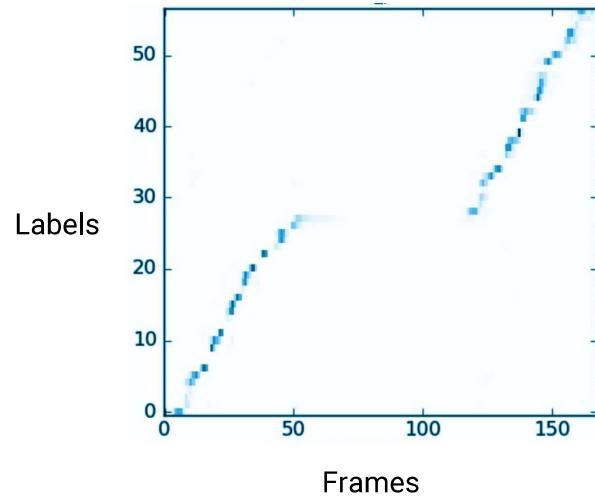


50

Attention-based Models



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



51

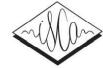
Comparing Various
Approaches:
Case-Study on a 12,500
hour Google Task

Google

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 Prabhavalkar¹, Kanishka Rao¹, Tara N. Sainath¹, Bo Li¹, Leif Johnson¹, Navdeep Jaitly^{2†}

¹Google Inc., U.S.A

²NVIDIA, 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]

Google

Experimental Setup: Model Configuration

- **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 in second pass
 - 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**

Google

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

Google

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

Decoding CTC-grapheme models without an LM performs poorly

Google

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
RNN-T	6.6	12.8

RNN-T which augments CTC with a neural LM significantly improves performance, and is close to the unidirectional baseline

Google

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
RNN-T	6.6	12.8
Attention-based Model	6.6	11.7

Attention-based model performs the best, but cannot be used for streaming speech recognition

Google

Comparison of End-to-End Approaches [Battenberg et al., 2017]

	Architecture	SWBD WER	CH WER	Model	Dev	Test
Published	Iterated-CTC [29]	11.3	18.7	CTC [4]		
	BLSTM + LF MMI [21]	8.5	15.3	Greedy decoding	23.03	-
	LACE + LF MMI ⁴ [28]	8.3	14.8	Beam search + LM (beam=2000)	15.9	16.44
	Dilated convolutions [25]	7.7	14.5	RNN-Transducer		
	CTC + Gram-CTC [17]	7.3	14.7	Greedy decoding	18.99	-
	BLSTM + Feature fusion[23]	7.2	12.7	Beam search (beam=32) + LM rescoring	17.41	-
Ours	CTC [17]	9.0	17.7		15.6	16.50
	RNN-Transducer			Attention		
	Beam Search NO LM	8.5	16.4	Greedy decoding	22.67	-
	Beam Search + LM	8.1	17.5	Beam search (beam=256)	18.71	-
	Attention			+ Length-norm weight	19.5	-
	Beam Search NO LM	8.6	17.8	+ Coverage cost	18.9	-
	Beam Search + LM	8.6	17.8	+ LM rescoring	16.0	16.48

Switchboard

DeepSpeech

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

Google

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)

Google

Turning Research Into Reality

Google

Moving From Research To Reality

- In order to use an end-to-end model for real-world applications, we need
 - Performance that matches that of a conventional model
 - Including MWER Training
 - Including External Language Model
 - More details in [\[Chiu et al., 2018\]](#)
 - Model must incorporate contextual biasing to long-tail words
 - Model must be streaming

MWER Training [Prabhavalkar et al., 2018]

Google

MWER Training of LAS Models: Motivation

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

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

- 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

MWER Training of LAS Models: Motivation

- Proposal: Minimize Expected 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]

MWER Training of LAS Models

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

Number of Word Errors

Minimizing expected WER directly is intractable since it involves a summation over all possible label sequences

MWER Training: Approximating expectation by sampling from the model

Google

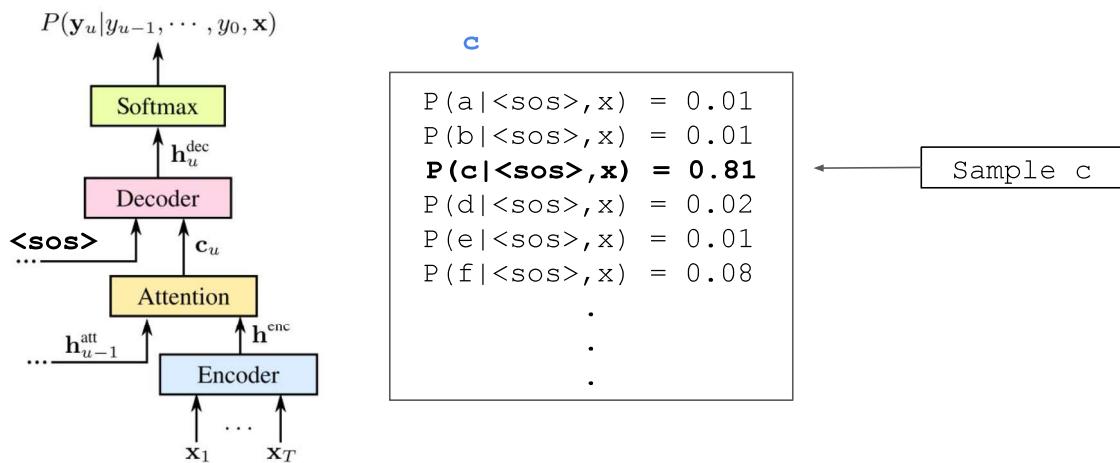
Approximation By Sampling [Shannon, 17]

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

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

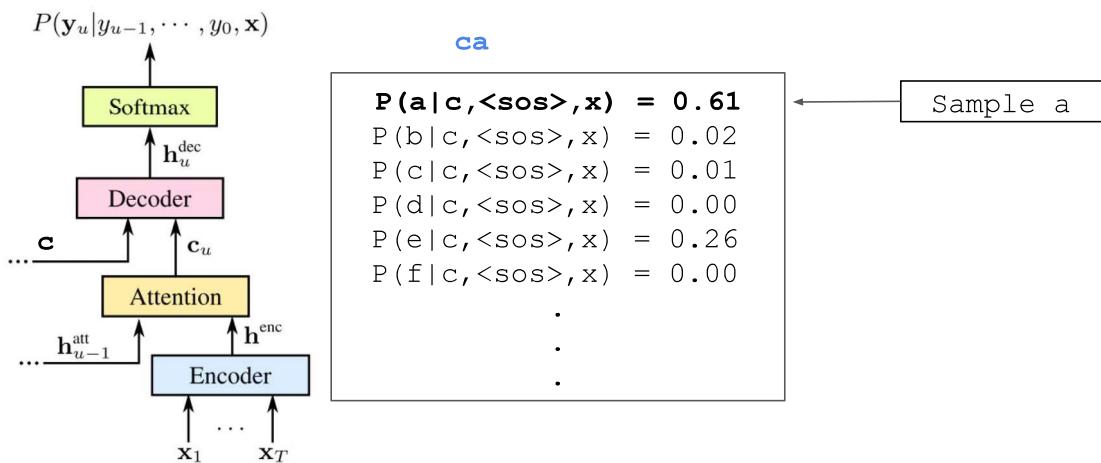
Approximate expectation using samples.

Approximation By Sampling [Shannon, 17]



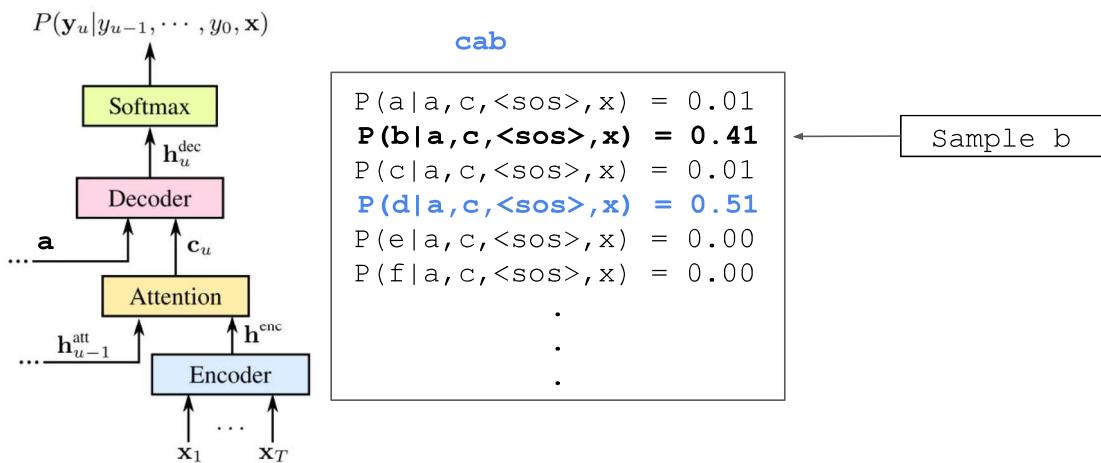
Drawing samples is particularly easy for locally-normalized models such as LAS!

Approximation By Sampling [Shannon, 17]



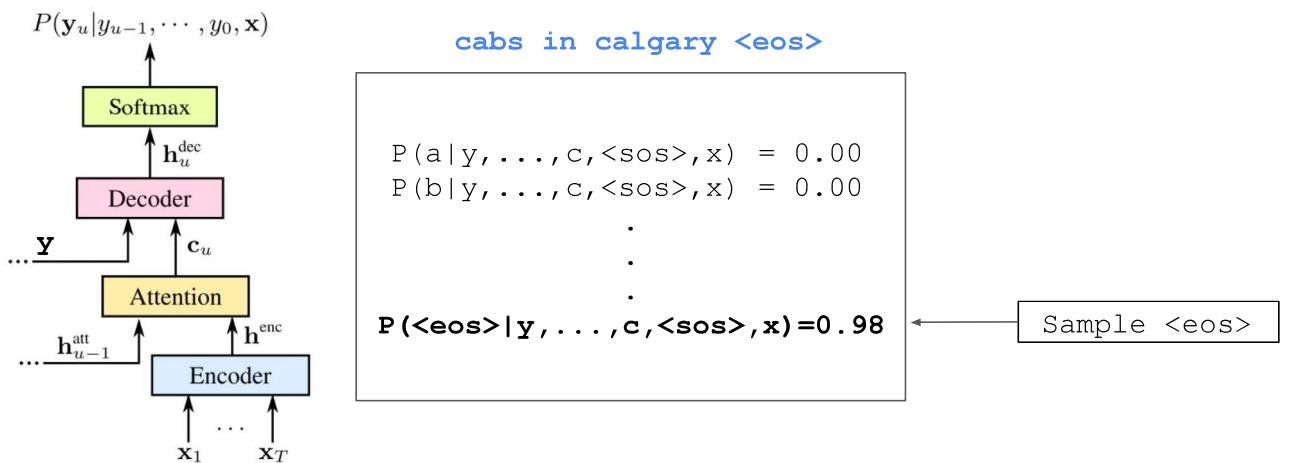
Drawing samples is particularly easy for locally-normalized models such as LAS!

Approximation By Sampling [Shannon, 17]



Drawing samples is particularly easy for locally-normalized models such as LAS!

Approximation By Sampling [Shannon, 17]



Drawing samples is particularly easy for locally-normalized models such as LAS!

Approximation By Sampling [Shannon, 17]

$$\begin{aligned}\nabla \mathcal{L}_{\text{werr}}^{\text{Sample}}(\mathbf{x}, \mathbf{y}^*) &= \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) [\mathcal{W}(\mathbf{y}, \mathbf{y}^*) - \mathbb{E}[\mathcal{W}(\mathbf{y}, \mathbf{y}^*)]] \nabla \log P(\mathbf{y}|\mathbf{x}) \\ &\approx \frac{1}{N} \sum_{\mathbf{y}_i \sim P(\mathbf{y}|\mathbf{x})} [\mathcal{W}(\mathbf{y}_i, \mathbf{y}^*) - \widehat{\mathcal{W}}] \nabla \log P(\mathbf{y}|\mathbf{x}) \quad (6)\end{aligned}$$

Gradient itself is an expectation, which can be approximated using samples!

Approximation By Sampling [Shannon, 17]

$$\begin{aligned}\nabla \mathcal{L}_{\text{werr}}^{\text{Sample}}(\mathbf{x}, \mathbf{y}^*) &= \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) [\mathcal{W}(\mathbf{y}, \mathbf{y}^*) - \mathbb{E}[\mathcal{W}(\mathbf{y}, \mathbf{y}^*)]] \nabla \log P(\mathbf{y}|\mathbf{x}) \\ &\approx \frac{1}{N} \sum_{\mathbf{y}_i \sim P(\mathbf{y}|\mathbf{x})} [\mathcal{W}(\mathbf{y}_i, \mathbf{y}^*) - \widehat{\mathcal{W}}] \nabla \log P(\mathbf{y}|\mathbf{x}) \quad (6)\end{aligned}$$

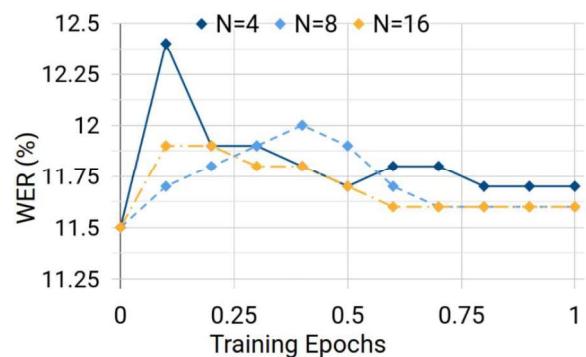
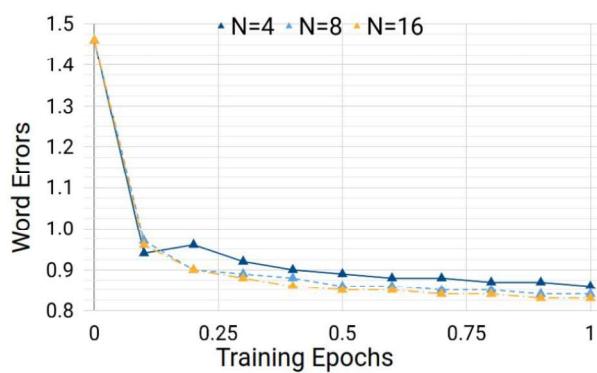
Increase the probability of sequences which have lower than average number of word errors!

Approximation By Sampling [Shannon, 17]

$$\mathcal{L}^{\text{Sample}} = \sum_{(\mathbf{x}, \mathbf{y}^*)} \mathcal{L}_{\text{werr}}^{\text{Sample}}(\mathbf{x}, \mathbf{y}^*) + \lambda \mathcal{L}_{\text{CE}}$$

Interpolate with CE-loss to stabilize training.
F-smoothing or H-criterion [Su et al., 2013]

Approximation By Sampling [Shannon, 17]



- Why doesn't the sampling-based approximation "work"?
 - Mismatch between decoding process during training (sampling) and decoding criterion (beam search) which focuses heavily on top hypotheses at each step [Kim and Rush, 2016]
 - In [Shannon, 17], paths are sampled from the lattice which corresponds to the most likely hypotheses, not from the space of all word sequences

MWER Training: Approximating expectation using decoded N-Best List

Google

Approximation using N-Best List [Stolcke+97][Povey,03]

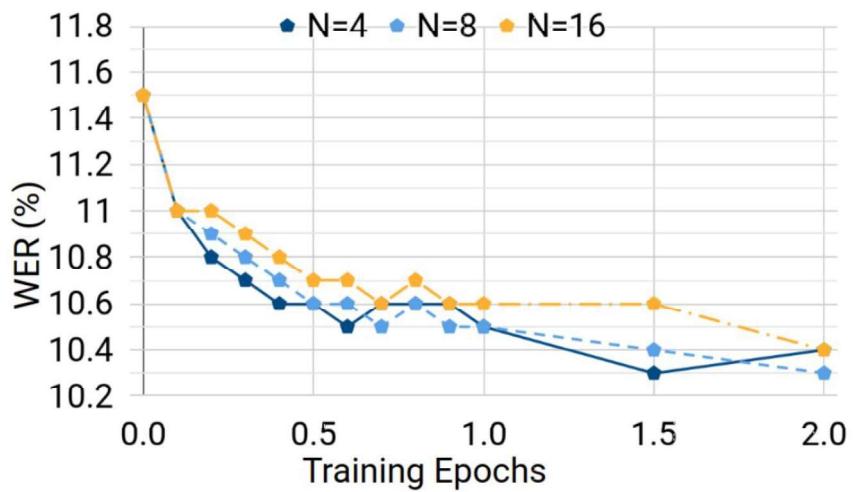
$$\mathcal{L}_{\text{werr}}(\mathbf{x}, \mathbf{y}^*) = \mathbb{E}[\mathcal{W}(\mathbf{y}, \mathbf{y}^*)] = \sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) \mathcal{W}(\mathbf{y}, \mathbf{y}^*)$$

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

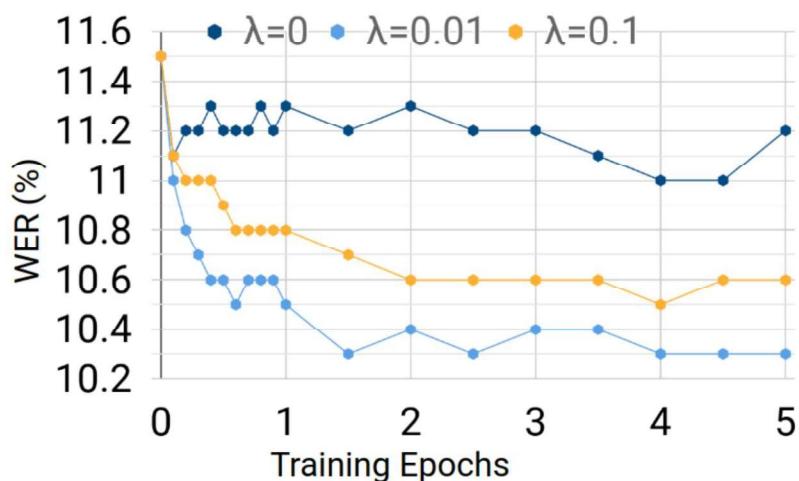
$$\widehat{P}(\mathbf{y}_i | \mathbf{x}) = \frac{P(\mathbf{y}_i | \mathbf{x})}{\sum_{\mathbf{y}_i \in \text{Beam}(\mathbf{x}, N)} P(\mathbf{y}_i | \mathbf{x})}$$

Assume that probability distribution is concentrated on top-N hypotheses

Approximation using N-Best List [Stolcke+97][Povey,03]



Approximation using N-Best List [Stolcke+97][Povey,03]



$$\mathcal{L}^{\text{N-best}} = \sum_{(\mathbf{x}, \mathbf{y}^*)} \mathcal{L}_{\text{werr}}^{\text{N-best}}(\mathbf{x}, \mathbf{y}^*) + \lambda \mathcal{L}_{\text{CE}}$$

Impact of interpolating MWER loss with CE loss during training.

Results: WER on Voice-Search Test Set

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

Results after direct decoding (beam size=8)

Results: WER on Voice-Search Test Set

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

Model + Second-Pass Rescoring	Uni-Directional Encoder	Bi-Directional Encoder
Baseline	7.3	6.6
+MWER Training	6.7 (8.2%)	6.2 (6.1%)

Results after N-best rescoring with Second-pass LM

MWER: Additional Comments

- 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

Results: LAS Model on Librispeech (960 hour task)

Model	Dev	DevOther	Test	TestOther
CE Baseline	5.8	16.1	6.2	16.4
MWER	5.3 (-8.8%)	15.2 (-5.7%)	5.7 (-8.4%)	15.4 (-6.0%)

Librispeech models trained on full 960 hour training data, with 16K word piece targets. Models are evaluated without an LM.

Language Model

Google

Motivation #1

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

Some Voice Search errors appear to be fixable with a good language model.

Google

Motivation #2

- 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 [\[Kannan et al., 2018\]](#)

Google

Shallow fusion

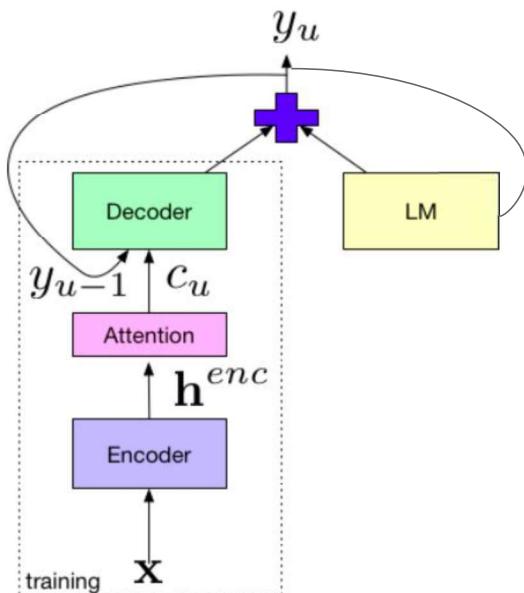
- Log-linear interpolation between language model and seq2seq model:

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

- 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.
- [\[Chorowski and Jaitly, 2017\]](#). [\[Kannan et al., 2018\]](#).

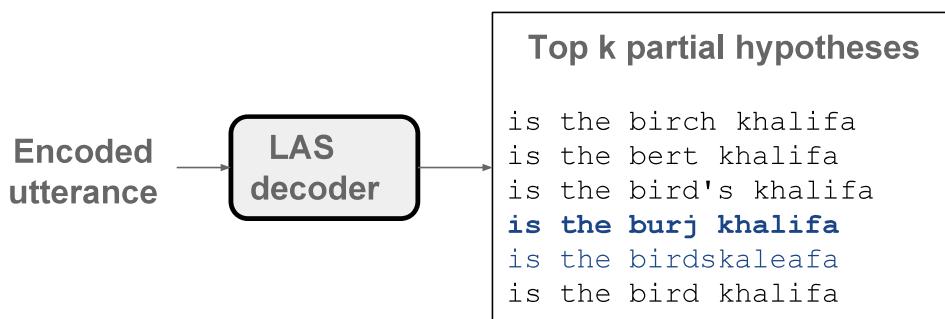
Google

Shallow fusion



Google

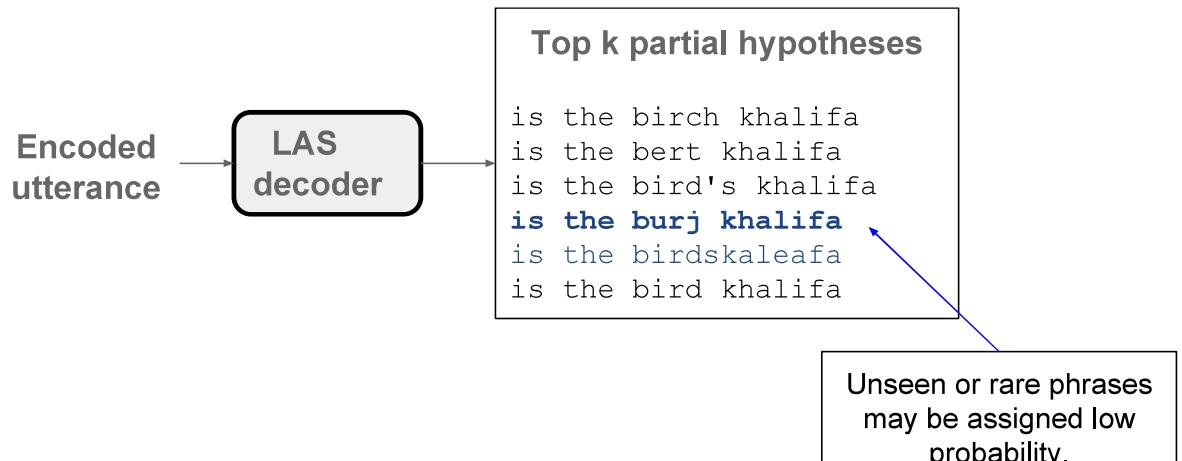
Baseline LAS Model



Baseline LAS model relies on LM learned from train data

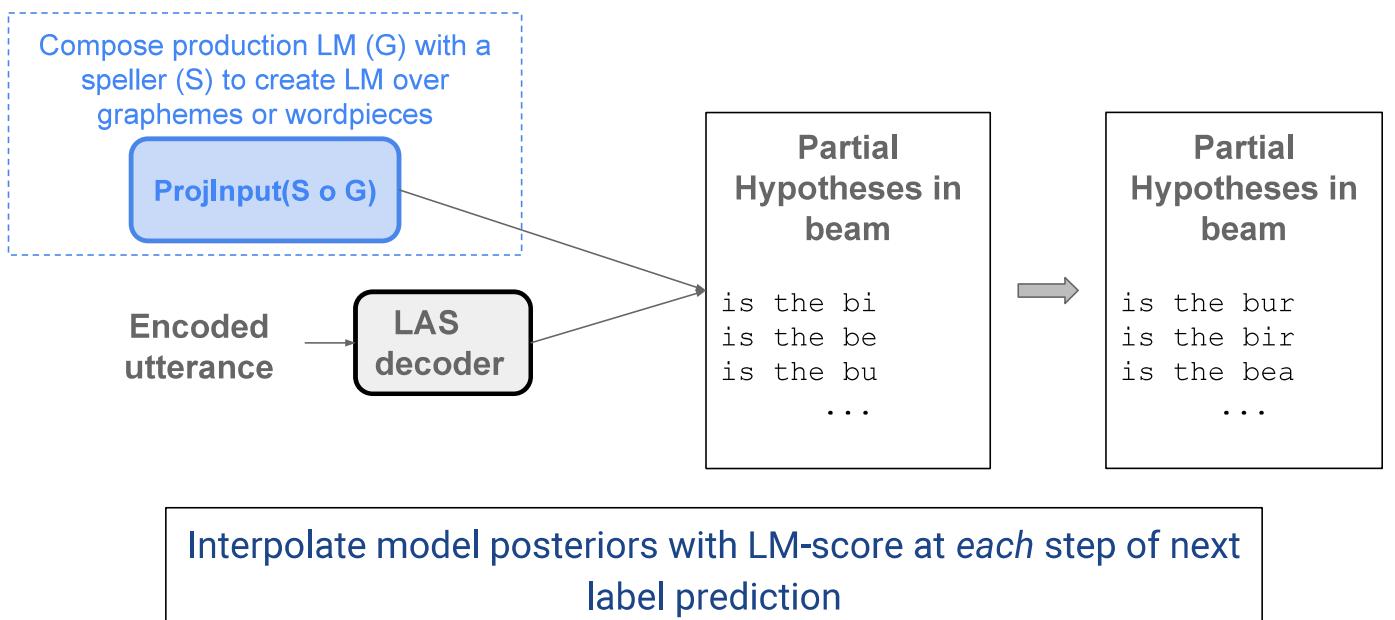
Google

Baseline LAS Model



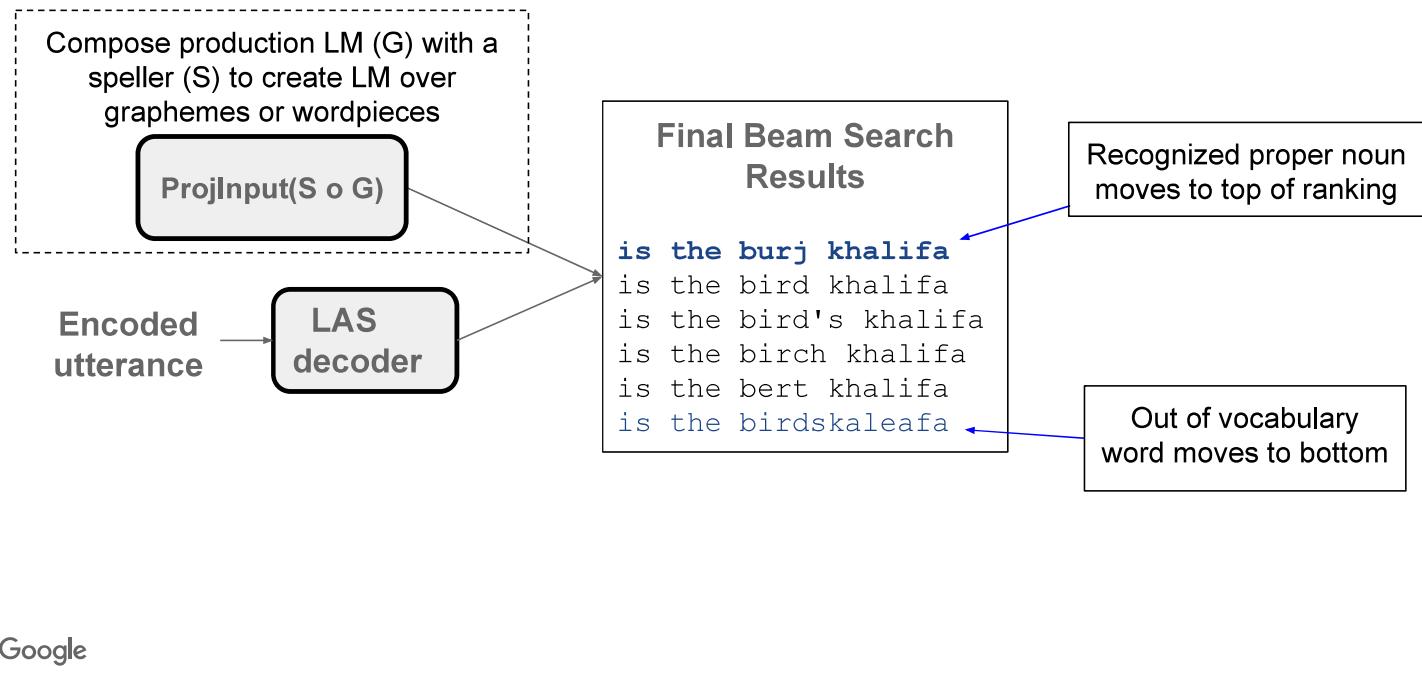
Google

Integration with FST LM in 1st pass



Google

Integration with FST LM in 1st pass



Results with FST LM

System	Dev WER	Test WER	LM Size
Baseline LAS	9.2%	7.7%	0 GB
LAS + FST LM in 1st pass	8.8%	7.4%	2 GB

Decoding with FST 1st pass production LMs into LAS system provides small improvement

Examples of LM wins

	Reference	Top 1 without LM	Top 1 with LM
Rare words	achondroplasia	acondra placia	achondroplasia
Proper nouns	st. isaac jogues mass schedule	st isaac jog's mass schedule	st isaac jogues mass schedule
	what causes high latency on a wi-fi connection?	what causes highlight and sienna wi-fi connection	what causes high latency on a wi-fi connection

Decoding with LM can correct errors early in decoding.
 In examples above, correct hypothesis does not appear in N-best without LM, so would not be possible to correct in second-pass with ProdLM.

Google

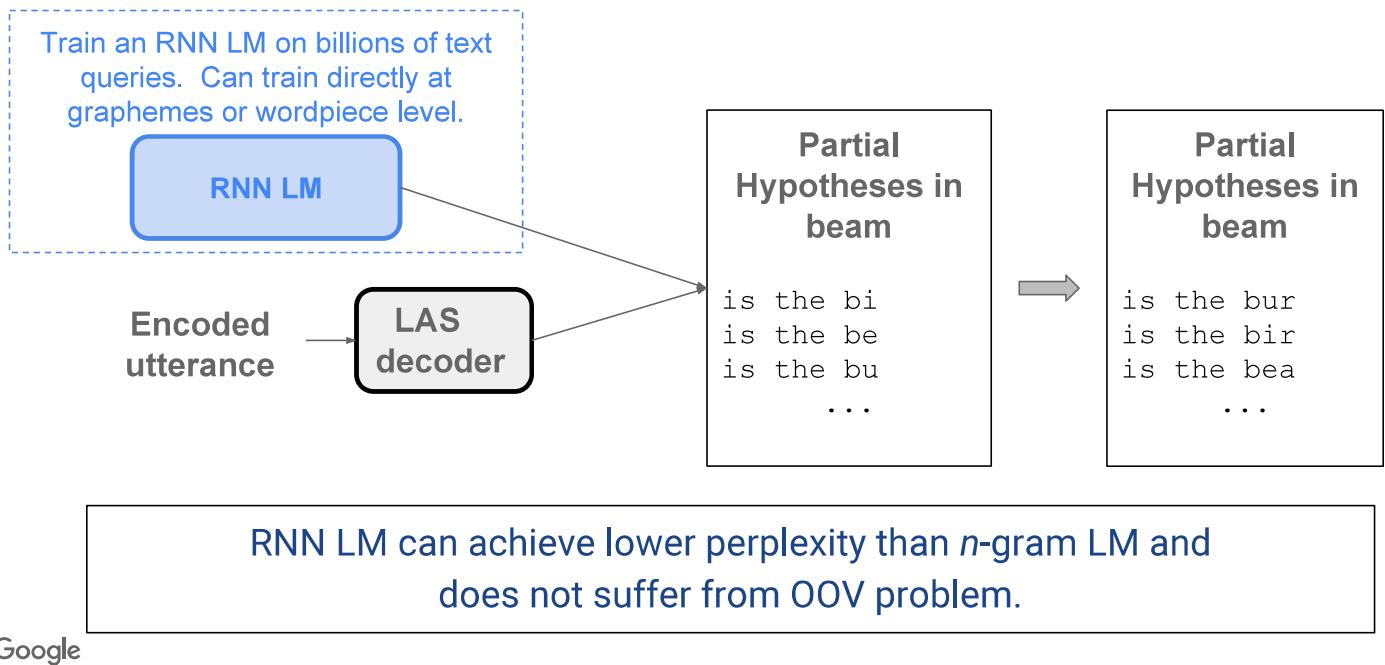
Examples of LM losses

	Reference	Top 1 without LM	Top 1 with LM
Out of vocab terms	urgent important unurgent unimportant	urgent important unurgent unimportant	urgent important un urgent unimportant
Websites (specific type of OOV)	mathfunbook.com product of a power property	mathfunbook.com product of a power property	math funbook com product of a power property
Grammatically incorrect language	why you not listening to me tonight	why you not listening to me tonight	why are you not listening to me tonight

LAS model can actually output words it has never seen before.
 Decoding with a language model removes this ability, costing about 0.2% absolute WER.

Google

Alternative: integrate with RNN LM in 1st pass



Google

Results with RNN-LM

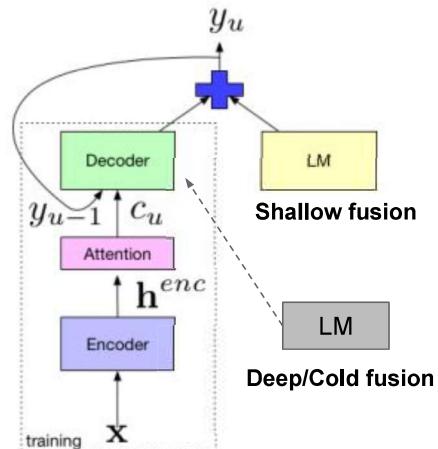
System	Dev WER	Test WER	LM Size
Baseline LAS	9.2%	7.7%	0 GB
LAS + FST LM in 1st pass	8.8%	7.4%	2 GB
LAS + RNN LM in 1st pass	8.4%	7.0%	1 GB

Decoding with RNN LM provides greater improvement at half the size!

Google

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., 2017]
 - Simple interface between a deep Im and the encoder
 - Allows to swap in task-specific LMs



Google

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

Google

Comparing E2E and Conventional Model

System	Test WER	LM Size (1st and 2nd pass)	Total Size
Conventional Model	6.7%	84.9 GB	87.2 GB
MWER LAS Model [1]	5.8%	0 GB	0.4 GB
LAS + RNN LM in 1st pass	5.5%	1 GB	1.4 GB

On our best system, we can reduce total model size by more than 50 times with a 20% WERR compared to a conventional model

Google

Handling Long Tail with Biasing

Google

What is “Biasing”?

“An attempt to adapt the priors baked into the speech models to better model information gained between training and inference (aka context).”

Google

Why Is Biasing Important

- Biasing can improve [WER](#) in domains by more than 10% relative

Test Set	WER, No Biasing	WER, Biasing
Contacts	15.0	2.8
Numeric	11.0	4.7
Yes-No-Cancel	18.8	10.4

Google

How To Bias E2E Models

- Two options for biasing
 - Bias externally (i.e., shallow-fusion)
 - Biasing within the model
 - Paper reference [\[Pundak et al., 2018\]](#)
 - In these experiments, we will evaluate on the following test sets
 - Contacts - “call Joe Doe, send a message to Jason Dean”
 - Songs - “play Lady Gaga, play songs from Jason Mraz”
 - Third Party - “text Jeanne, text John”

Google

(1) Biasing - Shallow Fusion

- General equation for shallow fusion during beam search

$$\vec{y}^* = \arg \max_{\vec{y}} \log P(\vec{y}|\vec{x}) + \lambda \log P_C(\vec{y})$$

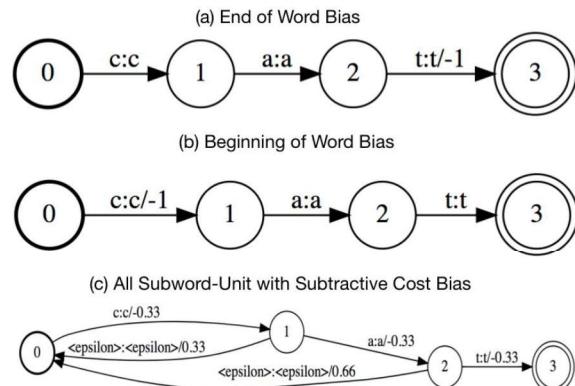
- Assumptions (for now)
 - Biasing is done at test time only
 - Tune interpolation weight λ per task

Google

Biasing - Where to Apply Scores?

- Best to apply to every unit (E3)

Experiment	Method (Grapheme)	WER - Songs
E0	No Bias (LAS)	20.9
E1	LAS + End of Word Bias	19
E2	LAS + Beginning of Word Bias	16.5
E3	LAS + Every Subword Unit w/ Subtractive Cost Bias	13.4



Google

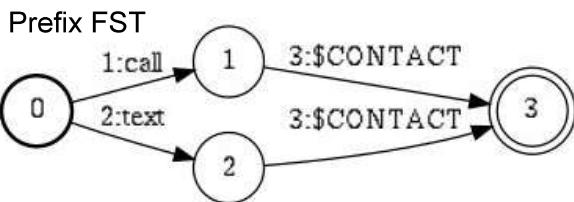
Improving Biasing Further

- Biasing FST should be applied before pruning the beam candidates, not rescoring a pruned beam (E4)
- Biasing at the WPM level is more effective than grapheme (E5)

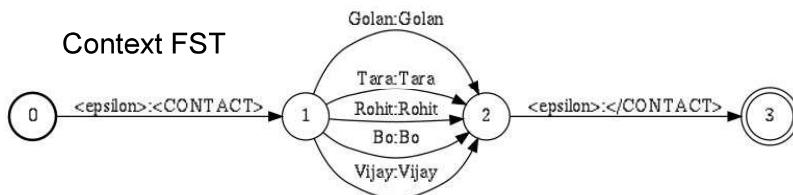
Experiment	Method (Grapheme)	WER - Songs
E0	No Bias (LAS)	20.9
E3	Grapheme Biasing	13.4
E4	Removing OTF Rescoring	9.4
E5	4K Word Piece Model LAS Biasing	6.9

Google

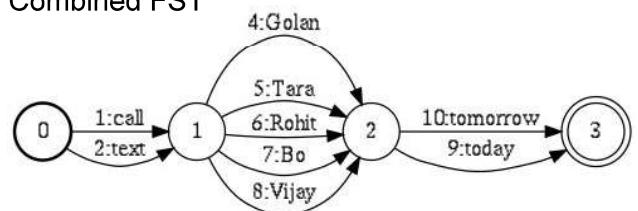
Prefixes & Suffixes



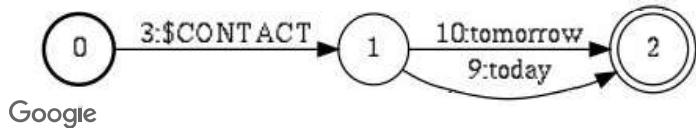
- Since E2E model cannot predict \$CONTACT, we combine all FSTs into one



Combined FST



Suffix FST



Prefixes & Suffixes

- Using this makes a large difference for biasing

Experiment	Method (Grapheme)	WER - Songs
E0	No Bias (LAS)	20.9
E5	4K Word Piece Model LAS Biasing	6.9
E6	+ Prefix and Suffix	5.6

Shallow Fusion Biasing Summary

- Biasing E2E Models similar quality to conventional model

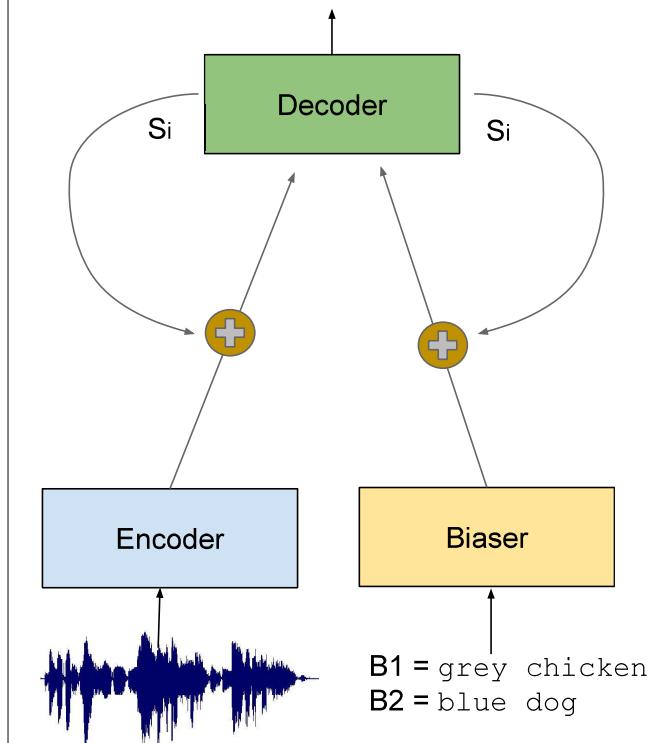
Method	CONTACTS	Songs	THIRD PARTY
Conventional Model No Biasing	36.1	26.5	-
Conventional Model Biasing	10.0	3.8	-
LAS No Biasing	26.9	16.8	10.5
LAS + Shallow Fusion, WPM 4K	7.1	5.6	3.9

Google

(2) Biased LAS Model (CLAS)

- Fixed-length embedding of bias phrases
- Attention over the embeddings, producing a **bias-dependent per-step** context vector
- Attention also includes a **N/A option** - don't apply bias

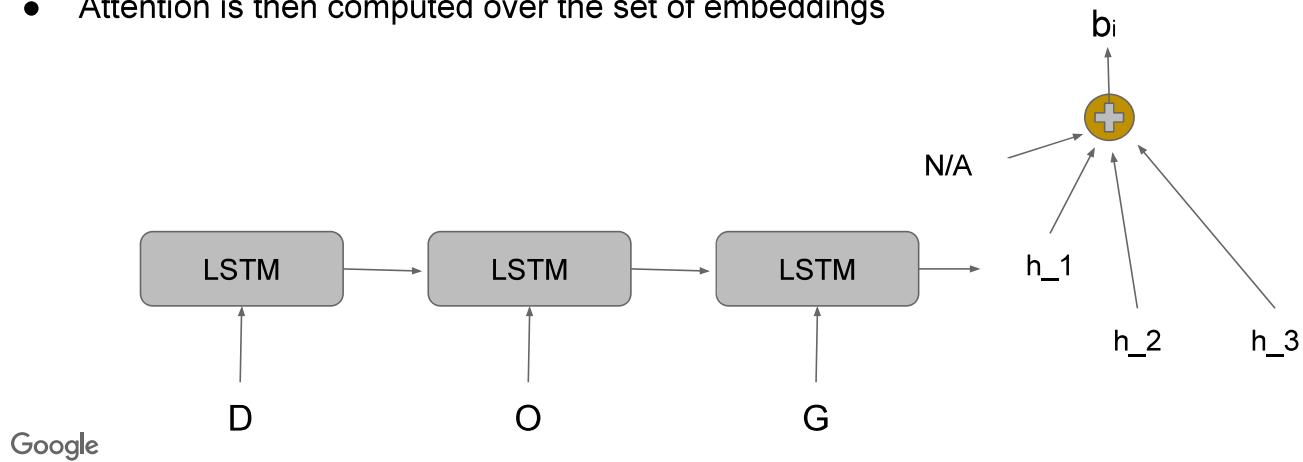
the grey chicken</bias> jumps



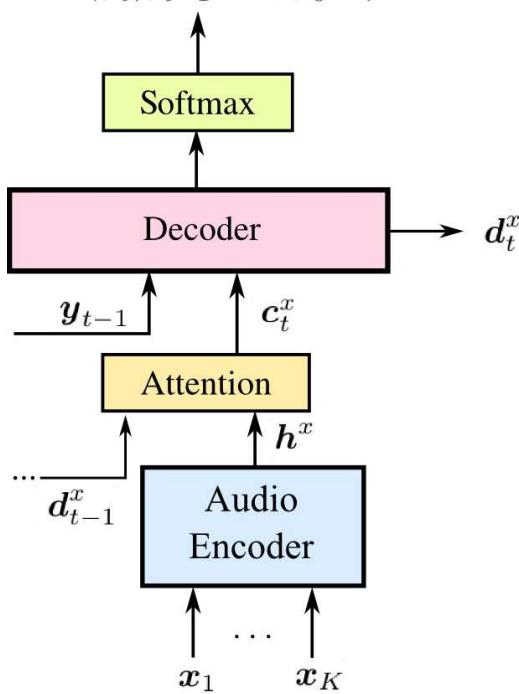
Google

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



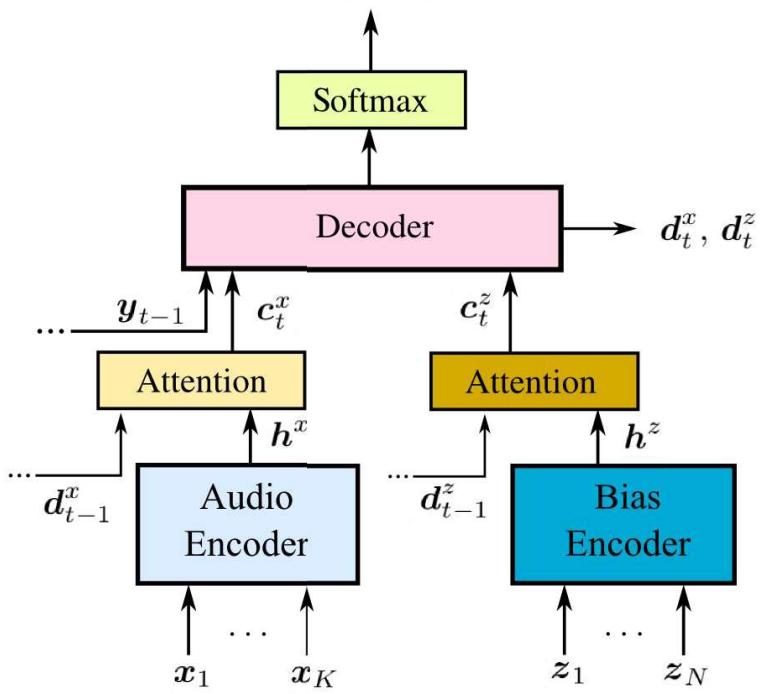
$$P(\mathbf{y}_t | \mathbf{y}_{t-1}, \dots, \mathbf{y}_0; \mathbf{x})$$



Google

(a.) LAS Model

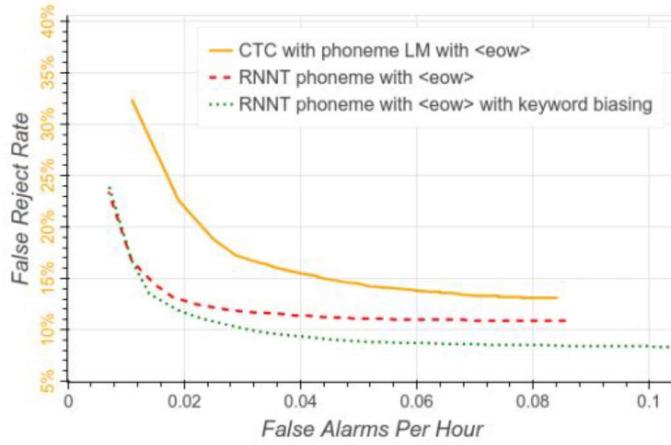
$$P(\mathbf{y}_t | \mathbf{y}_{t-1}, \dots, \mathbf{y}_0; \mathbf{x}; \mathbf{z})$$



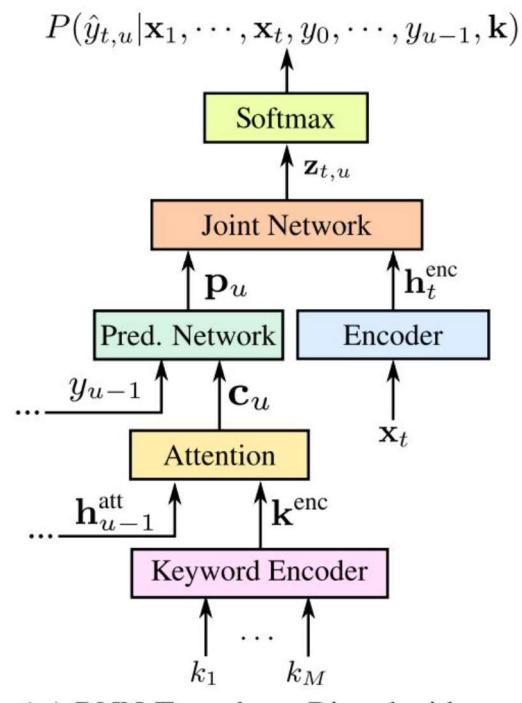
(b.) BLAS Model

Prior work: Keyword spotting with RNNT

- “Streaming Small-Footprint Keyword Spotting using Sequence-to-Sequence Models” [He et al., 2017]



Google

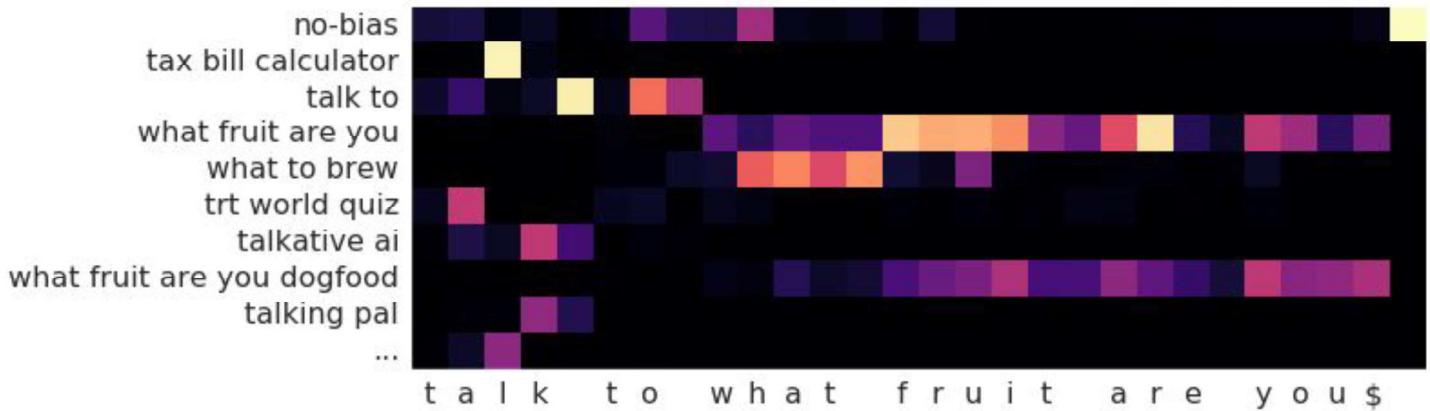


CLAS training

- 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 B and replace it with another bias phrase from the same batch
- Augment with additional N-1 more bias phrases from other references in the batch (distractors)
- Present the model the set of N (shuffled) bias phrases:
 - 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**

Google

Biasing Example



Google

Key aspects of CLAS

- Biasing is viewed as a keyword detection task which relates to both audio and LM (cf. beam search biasing)
- CLAS embeds “long” var-length bias sequences into fixed-length vectors
- CLAS computes attention over a set of phrases
- The model can take any list of bias phrases in inference time (including OOVs)
 - In training the bias phrases list is randomized for each batch
 - The number and content of bias phrases can be changed from training to inference

Google

Biasing Summary

- CLAS model performs similar to biasing of conventional model

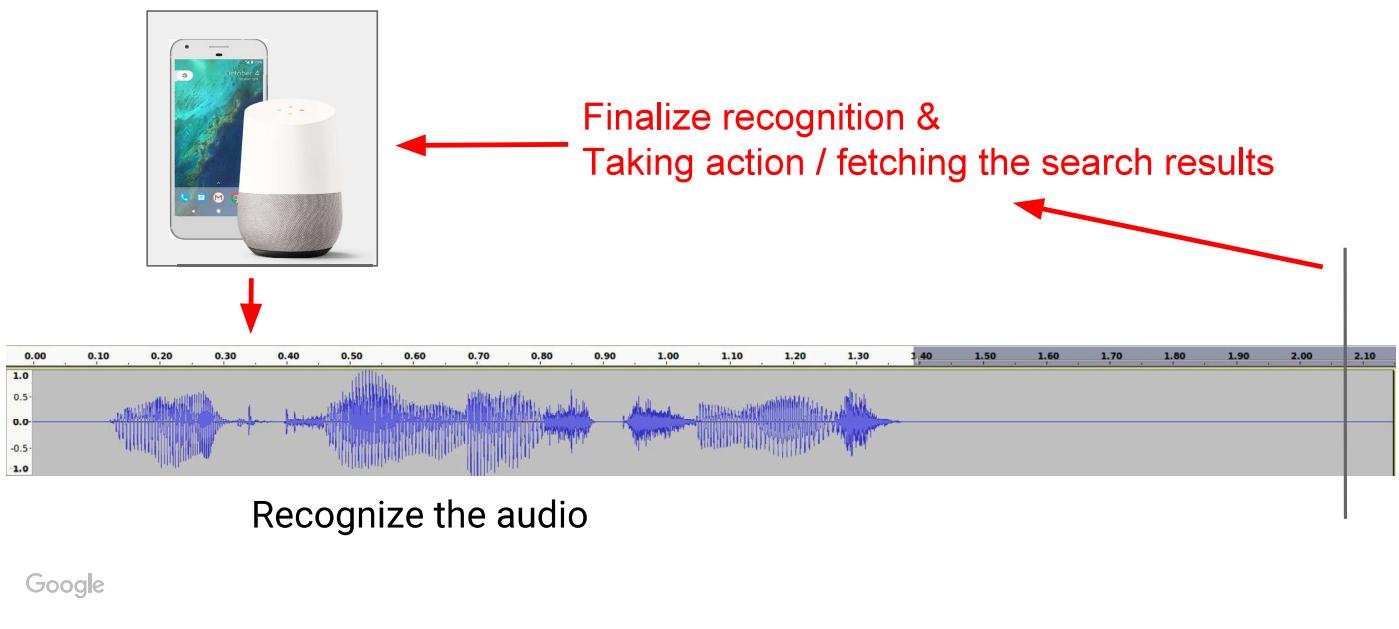
Method	CONTACTS	Songs	THIRD PARTY
Conventional Model No Biasing	36.1	26.5	-
Conventional Model Biasing	10.0	3.8	-
LAS No Biasing	26.9	16.8	10.5
CLAS + Shallow Fusion, Grapheme	7.5	5.7	5.6

Google

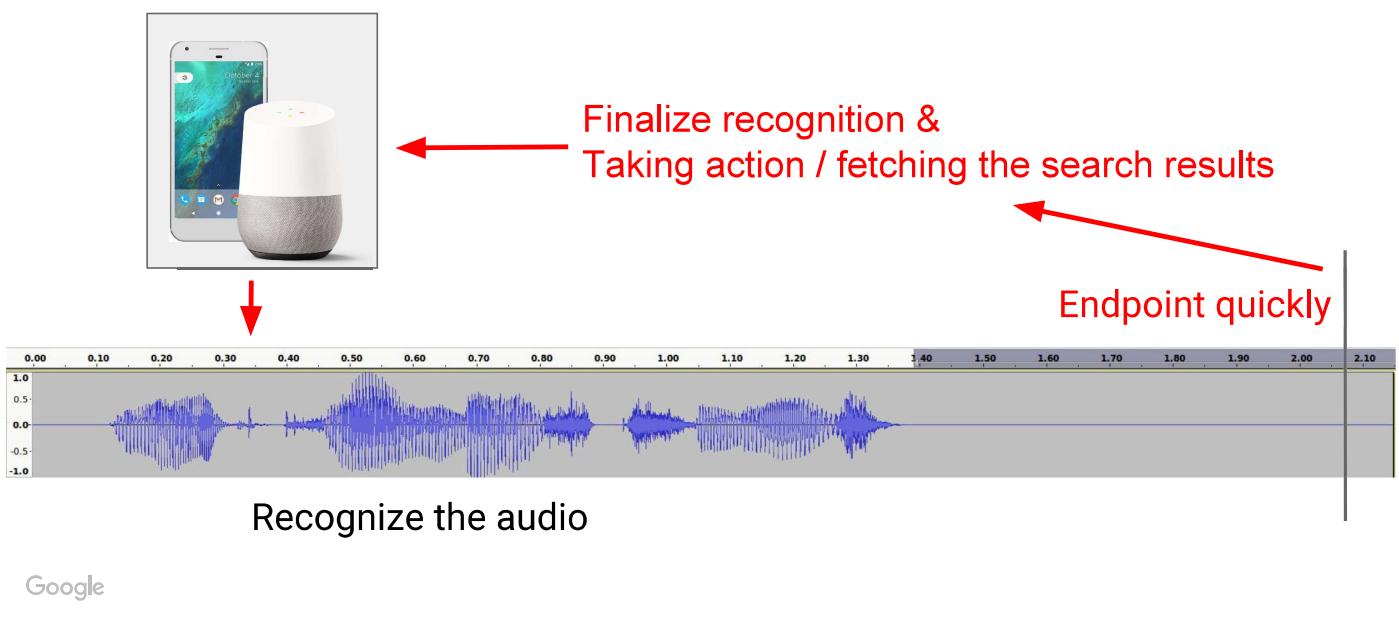
Online Models

Google

Streaming speech recognition



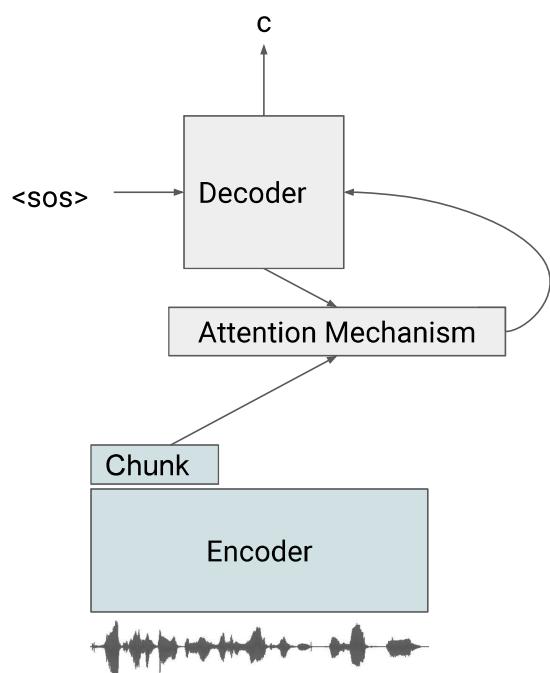
Streaming speech recognition



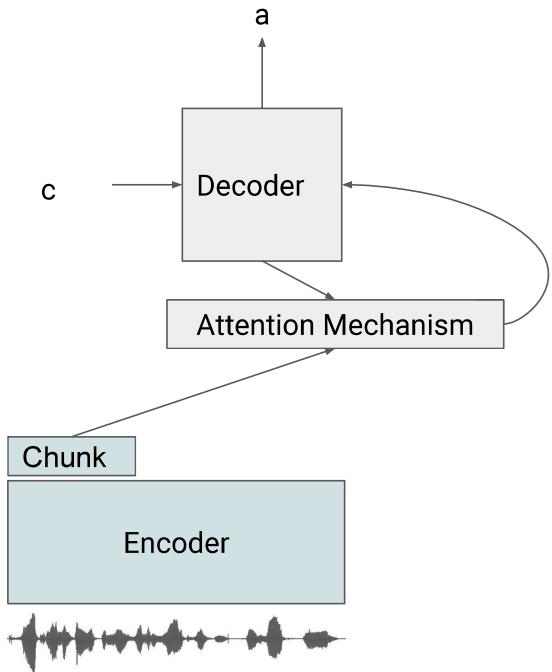
Online Models

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

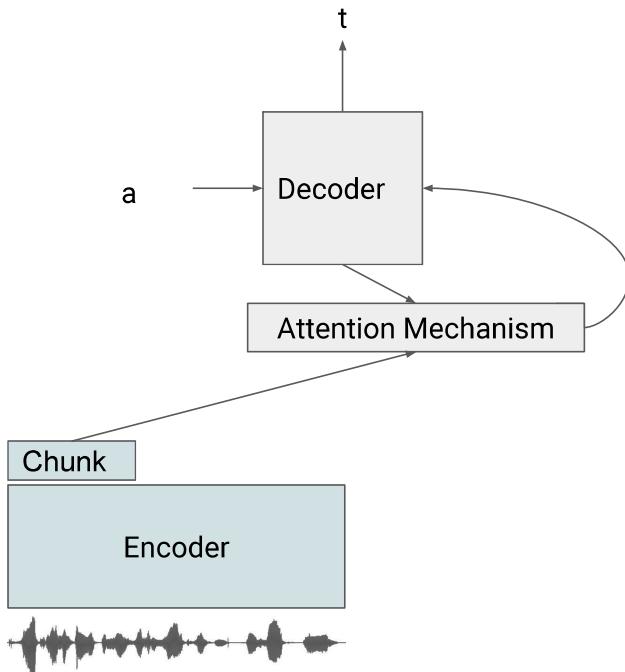
(1) Neural Transducer: “Online” Attention Models



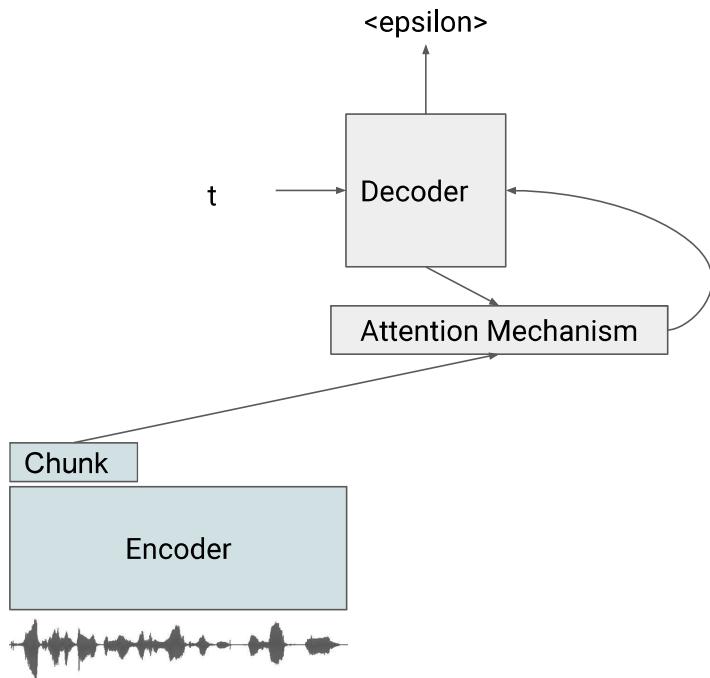
(1) Neural Transducer: “Online” Attention Models



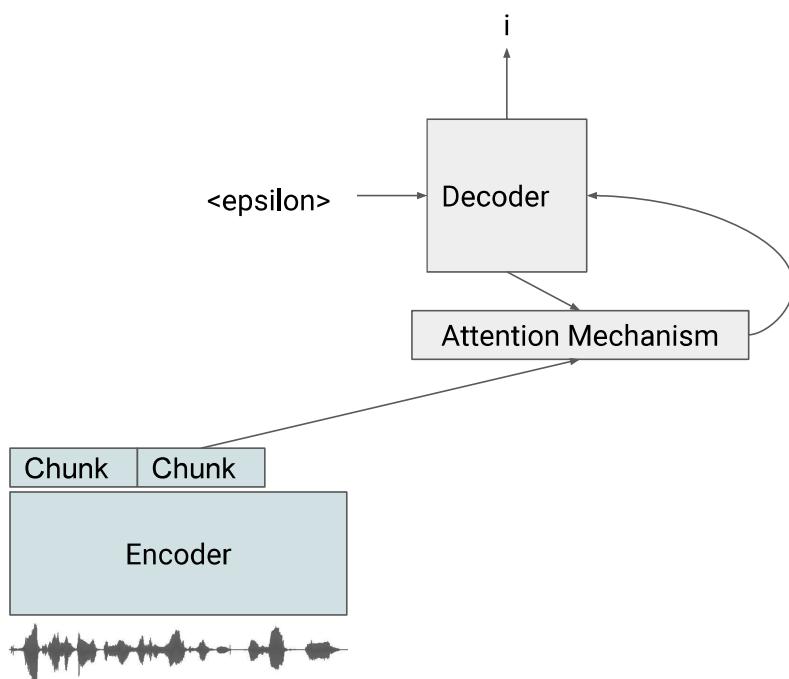
(1) Neural Transducer: “Online” Attention Models



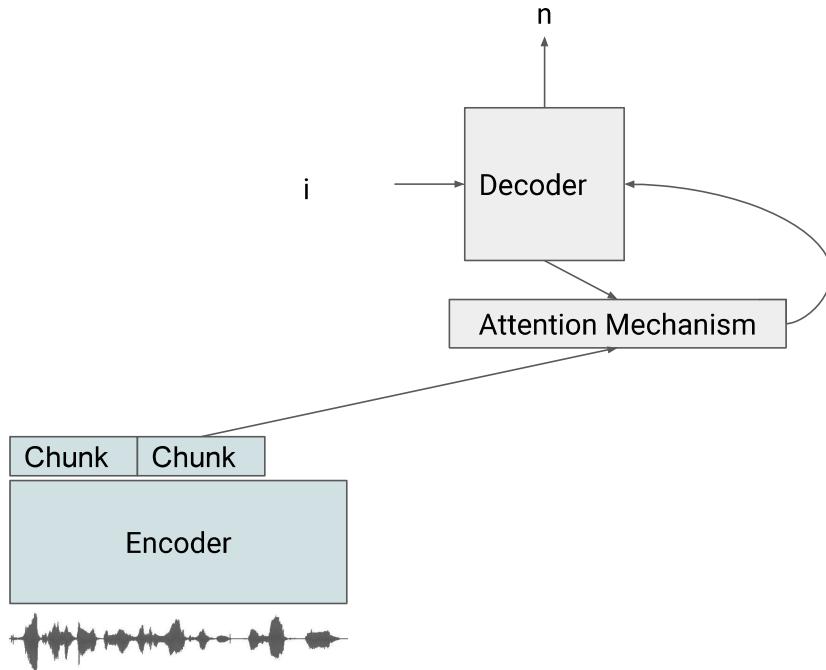
(1) Neural Transducer: “Online” Attention Models



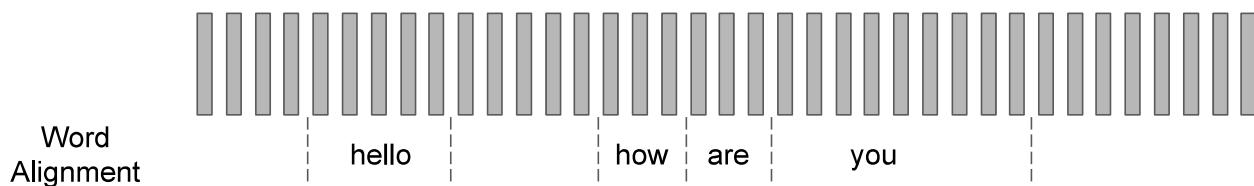
(1) Neural Transducer: “Online” Attention Models



(1) Neural Transducer: “Online” Attention Models

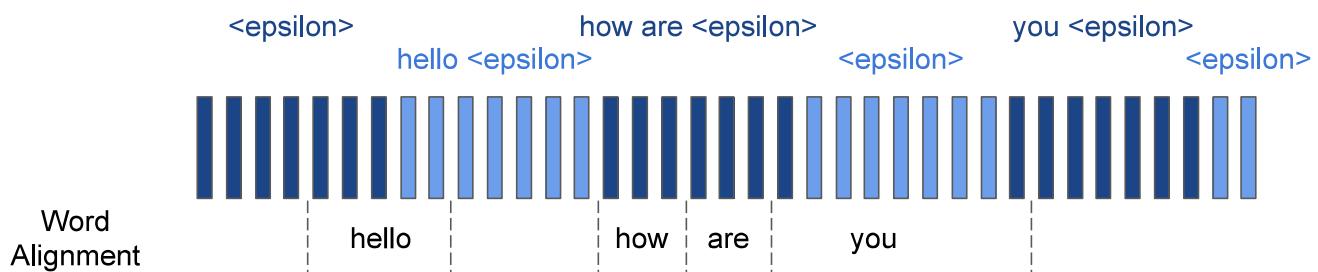


Training Data for Neural Transducer



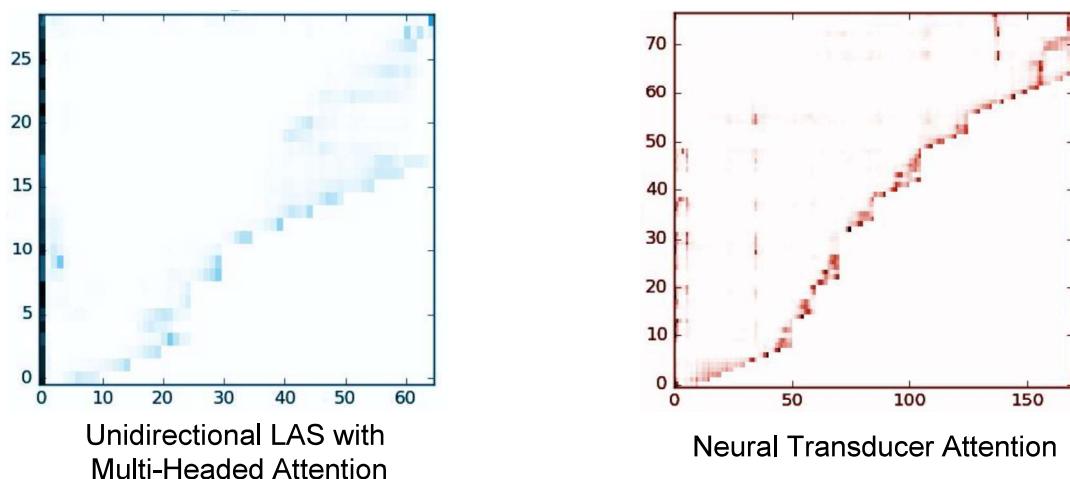
- Online methods like RNN-T, Policy Gradient learn alignment jointly with model
- We train neural transducer with a pre-specified alignment, so don't need to re-compute alignments (e.g., forward-backward) during training, which slows things down on GPU

Training Data for Neural Transducer



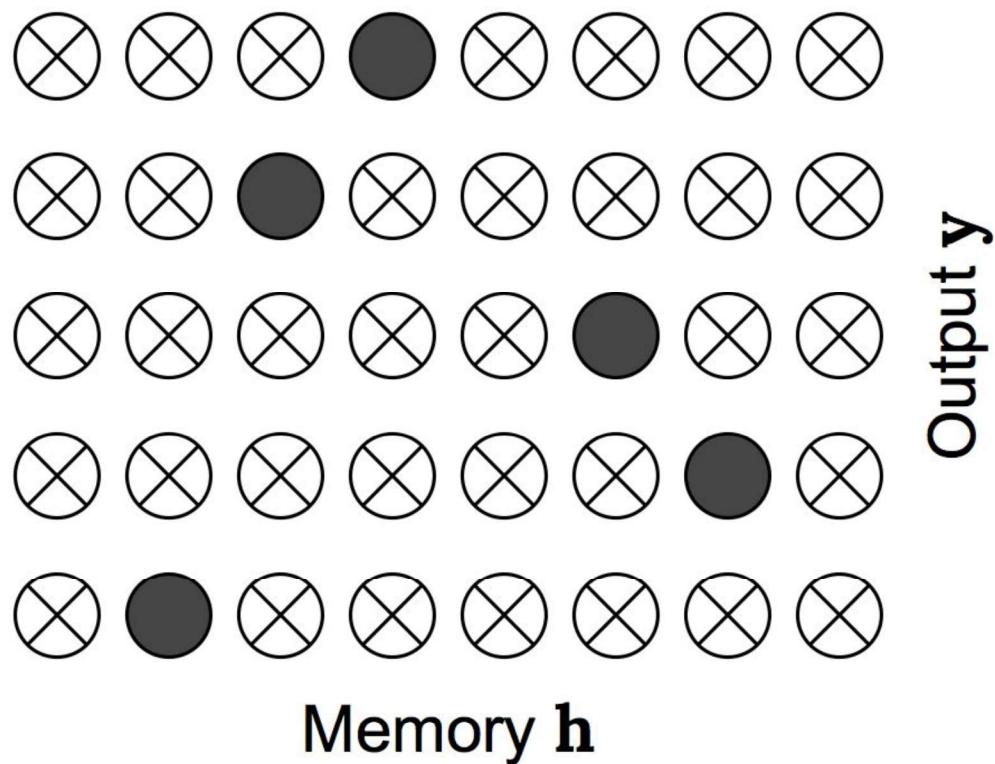
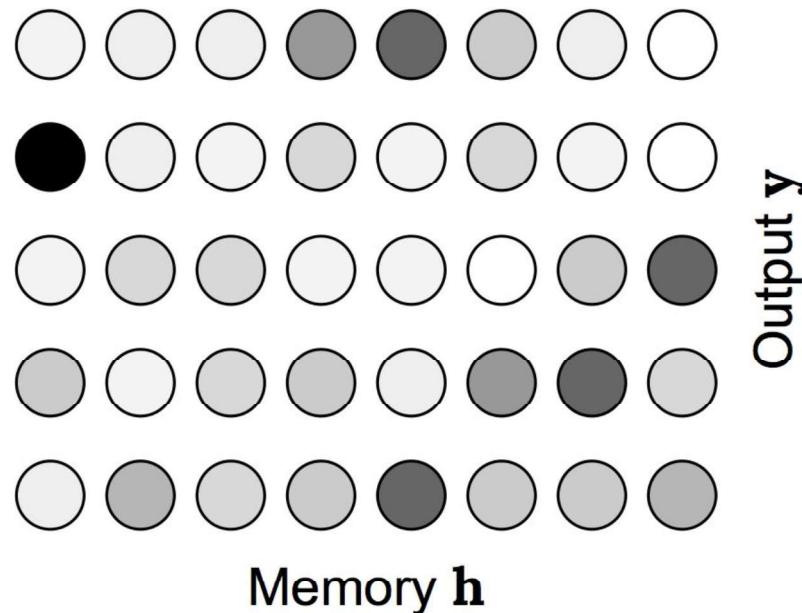
- <epsilon> signals end-of-chunk
- Since we don't have grapheme-level alignments, we wait till the end of the word to emit the entire word's graphemes

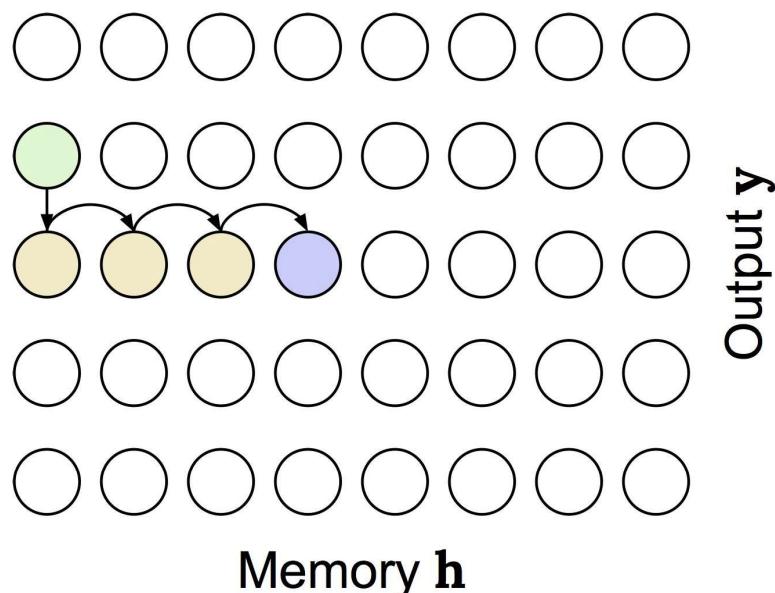
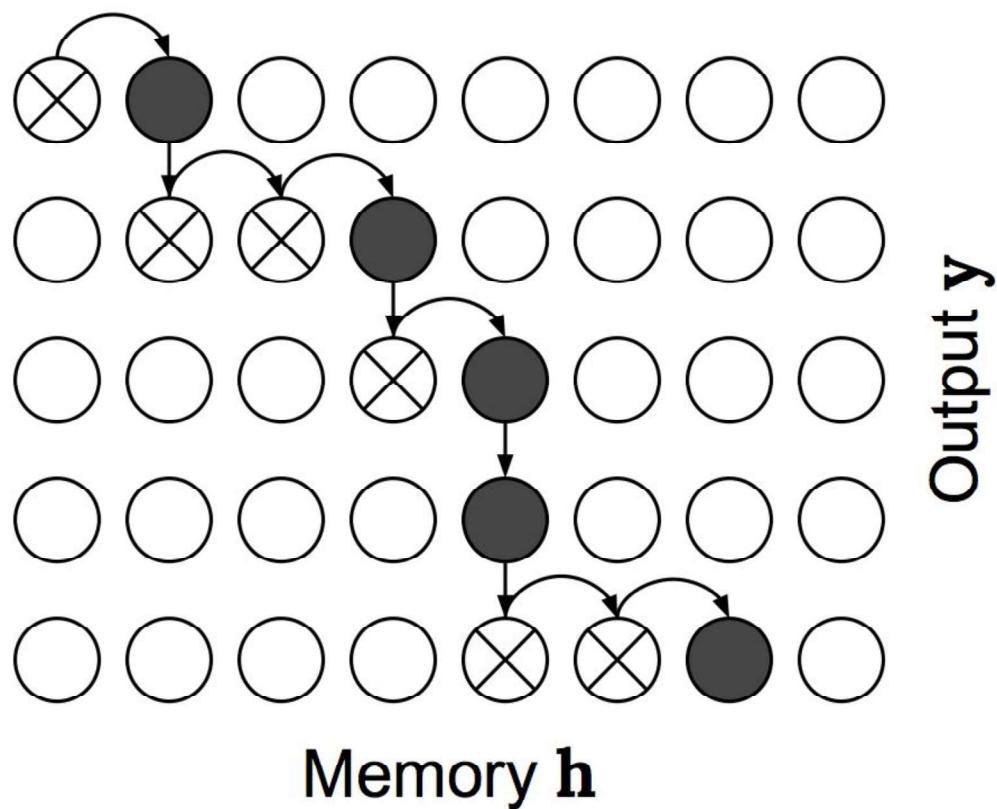
Neural Transducer Attention Plot



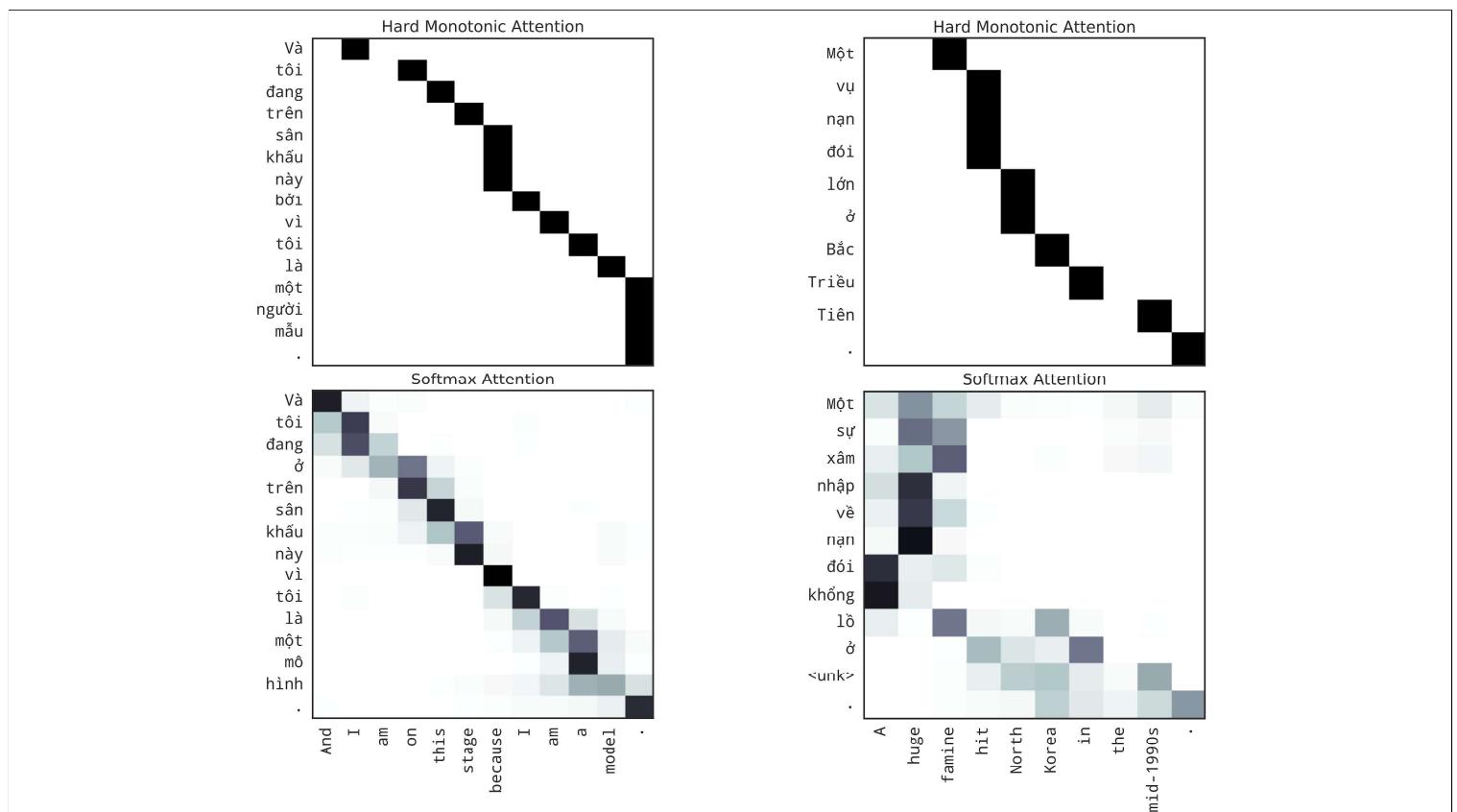
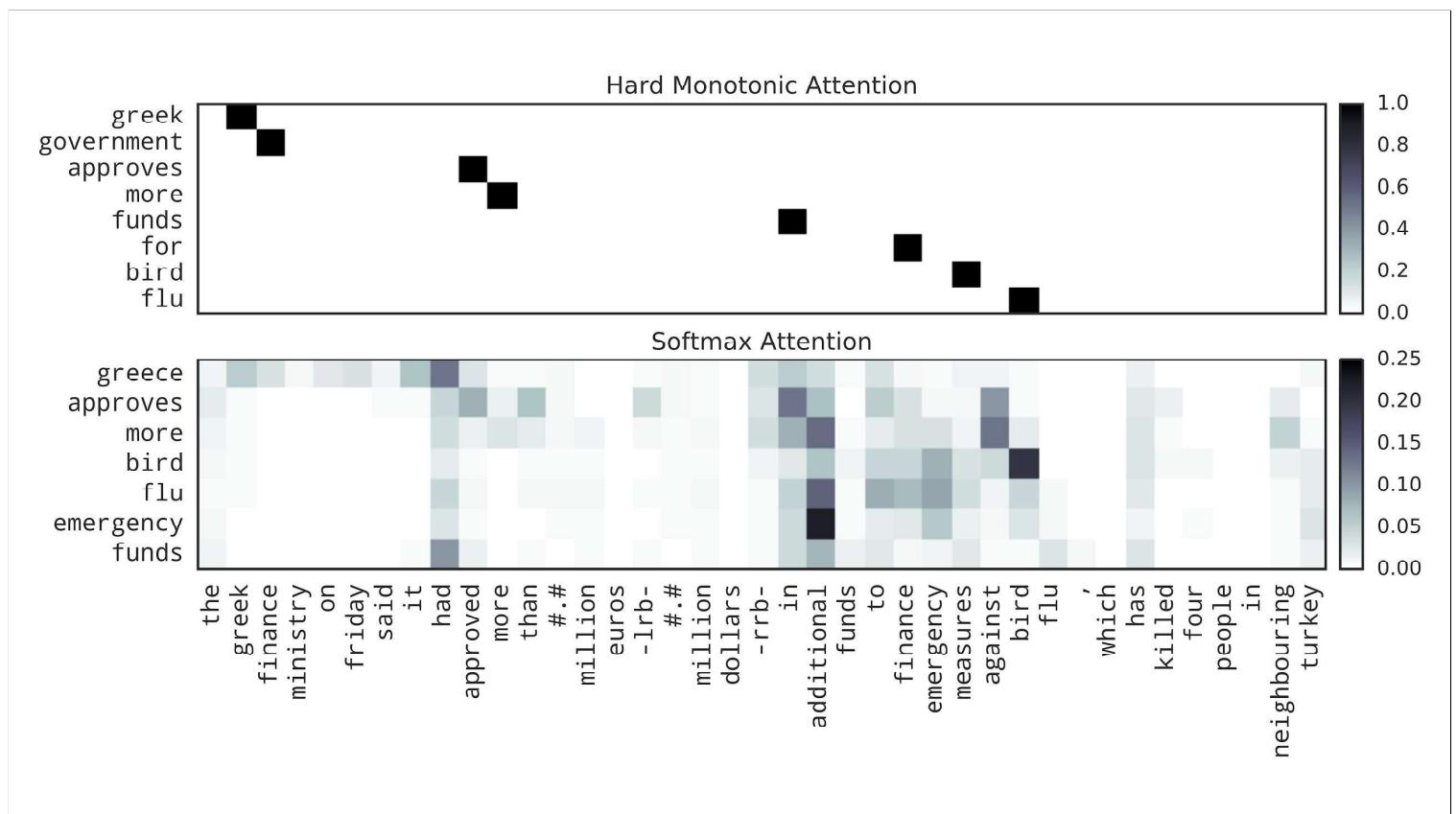
NT model examines previous frames without looking beyond the current chunk

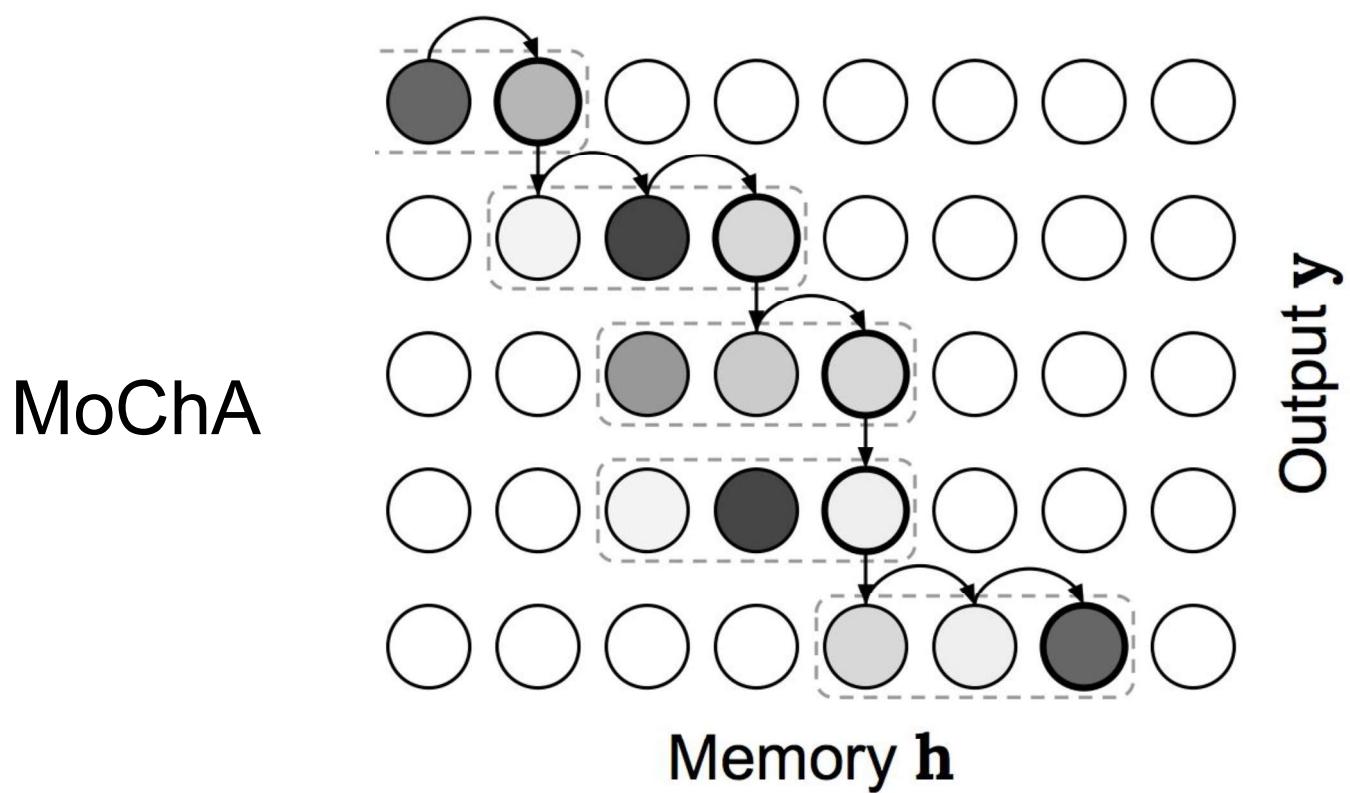
Monotonic Attention



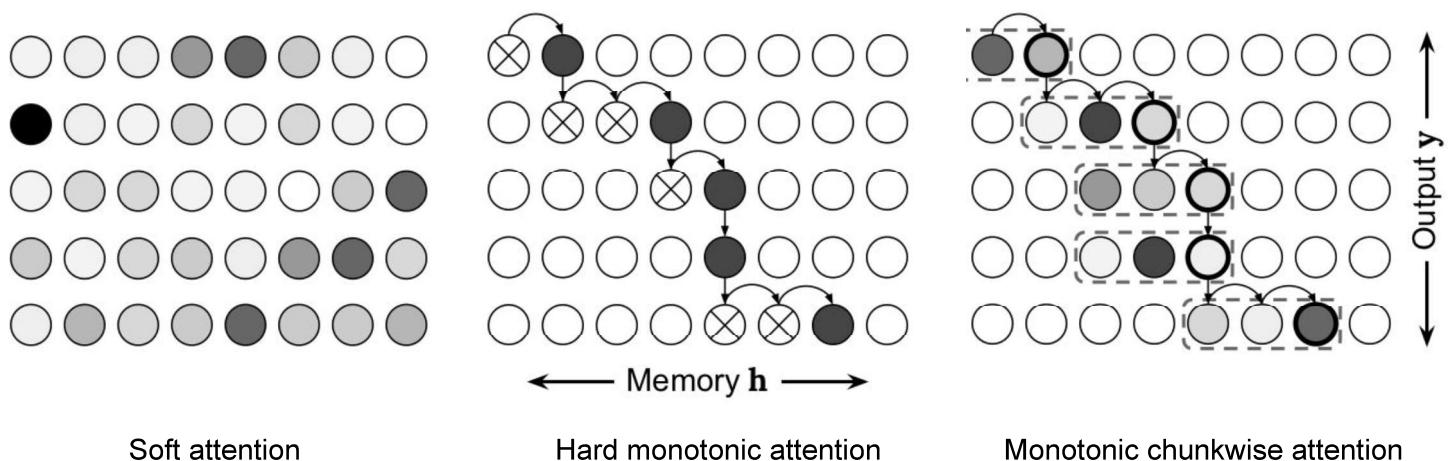


$$\text{attention}_{i,j} = \text{select}_{i,j} \sum_{k=1}^j \left(\text{attention}_{i-1,k} \prod_{l=k}^{j-1} (1 - \text{select}_{i,l}) \right)$$





(2) Monotonic chunkwise attention (MoChA)

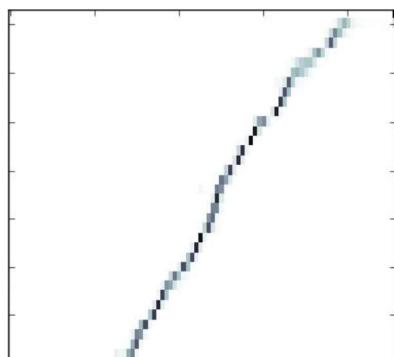


[Chiu and Raffel, 2018]

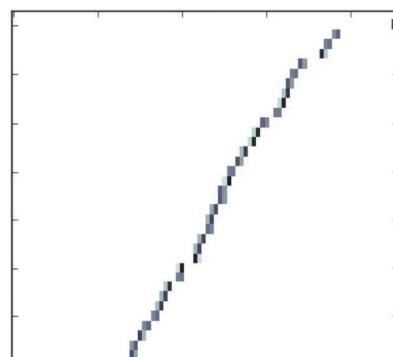
Google

Training monotonic chunkwise attention

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



Google



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

Google

Endpointer

Google

Why is Endpointing hard?

1. Latency vs WER tradeoff

(early trigger - affects WER) (late trigger - affects latency)

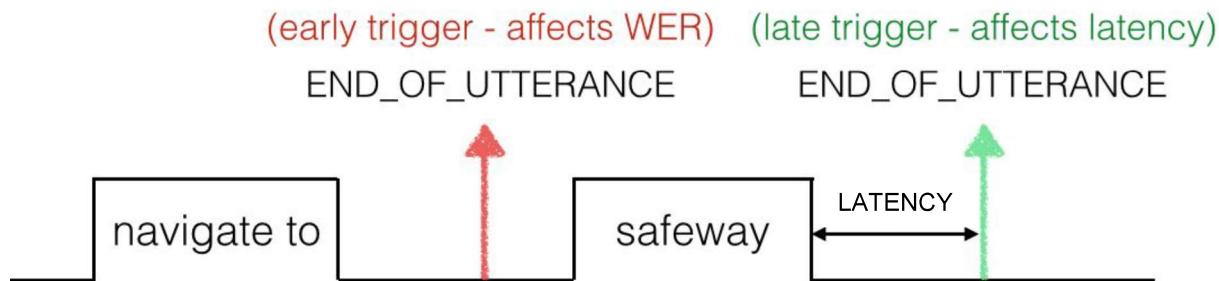
END_OF_UTTERANCE

END_OF_UTTERANCE



Why is Endpointing hard?

1. Latency vs WER tradeoff

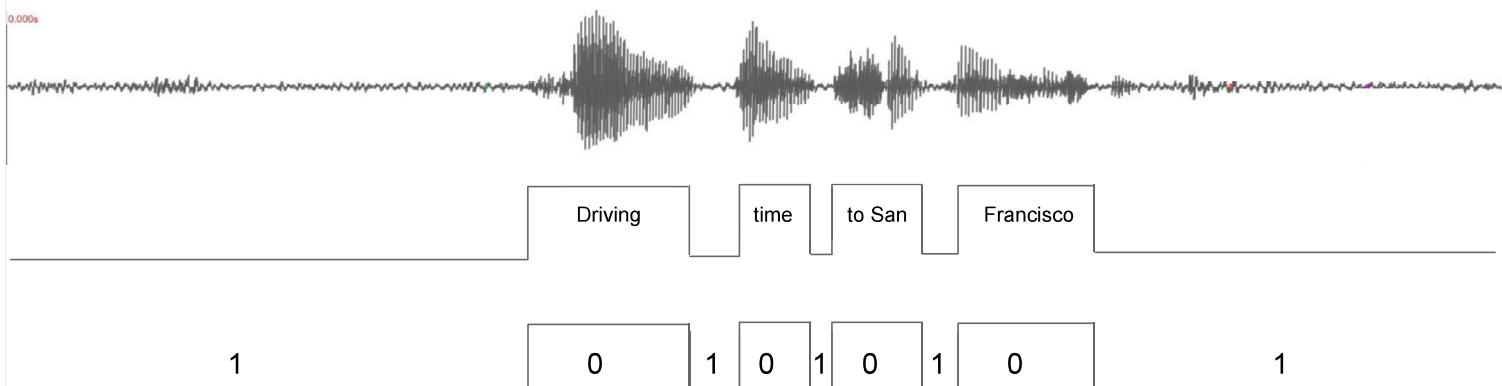


2. Noisy conditions



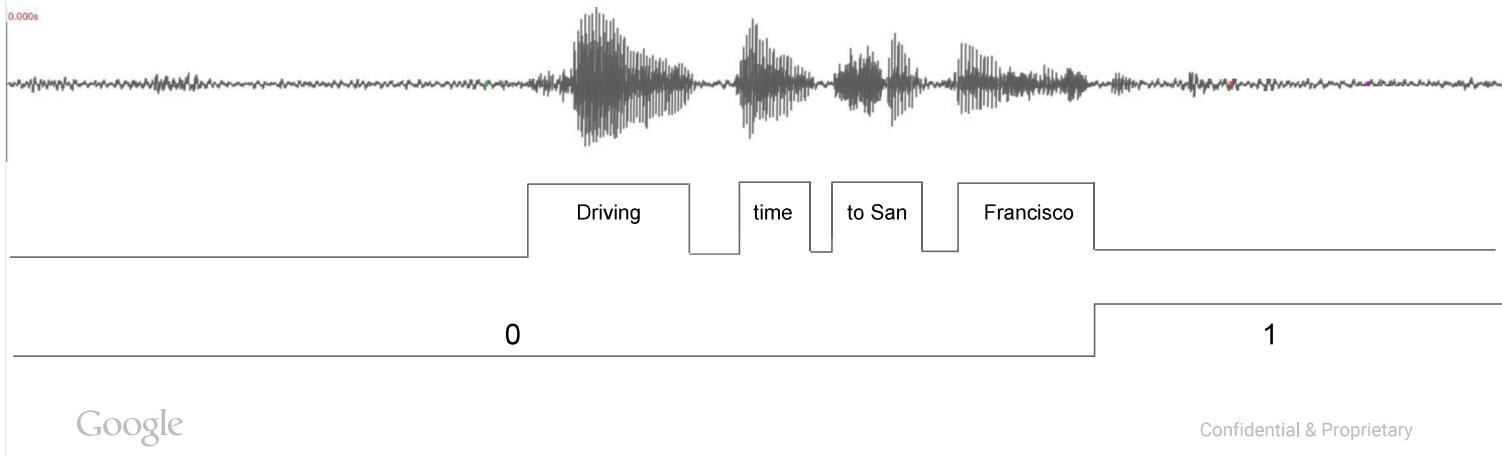
VAD based endpointer

- Use forced alignment to find the timing of the utterance
- Based on the timing mark each frame as SPEECH (0) or NON-SPEECH (1)
- Made mic closing decision when a fixed amount of silence is detected



End-of-utterance (EOU) detector

- Use forced alignment to find the timing of the utterance
- Based on the timing mark each frame as KEEP_OPEN (0) or CLOSE (1)
- Posteriors of CLOSE are thresholded to obtain a mic closing decision.

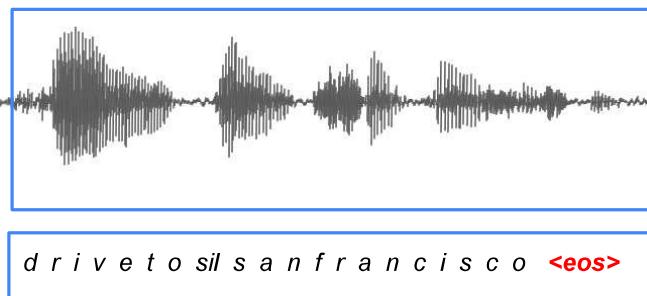


Google

Confidential & Proprietary

E2E endpointer

- An unified model does endpointing and ASR
- Add an <eos> symbol to the end of each target transcript.
- If top-1 hypothesis in the beam outputs a <eos> for a frame **then close mic.**



Google

Confidential & Proprietary

E2E endpointer

Parameters to precise control E2E endpointing:

- Cost penalty: scale up for **<eos>** cost
- Pruning: max cost of **<eos>** allowed in beam search

Similar to thresholds for EOU detector or VAD

Google

Confidential & Proprietary

Evaluating an Endpointer



- Measure endpointer latency
 - Use forced alignment to find the time of the 'sil' that's not followed by speech.
 - Compare that to the timestamp of the END_OF_UTTERANCE.
- Metrics:
 - median latency
 - 90th percentile latency
 - WER

Google

Confidential & Proprietary

Results summary

- VAD baseline: 900 ms median latency
- VAD -> EOU detector: ~100 ms improvement
- VAD -> E2E endpointer: **up to ~700 ms improvement!!**

	WER	Median latency	90th percentile latency
(1) CTC AM	14.5	890	960
(2) RNNT no EP	8.4	-	-
(3) RNNT + VAD EP	8.8	900	1030
(4) E2E RNNT EP	8.8	210	1010

Google

Confidential & Proprietary

Extensions of E2E Models

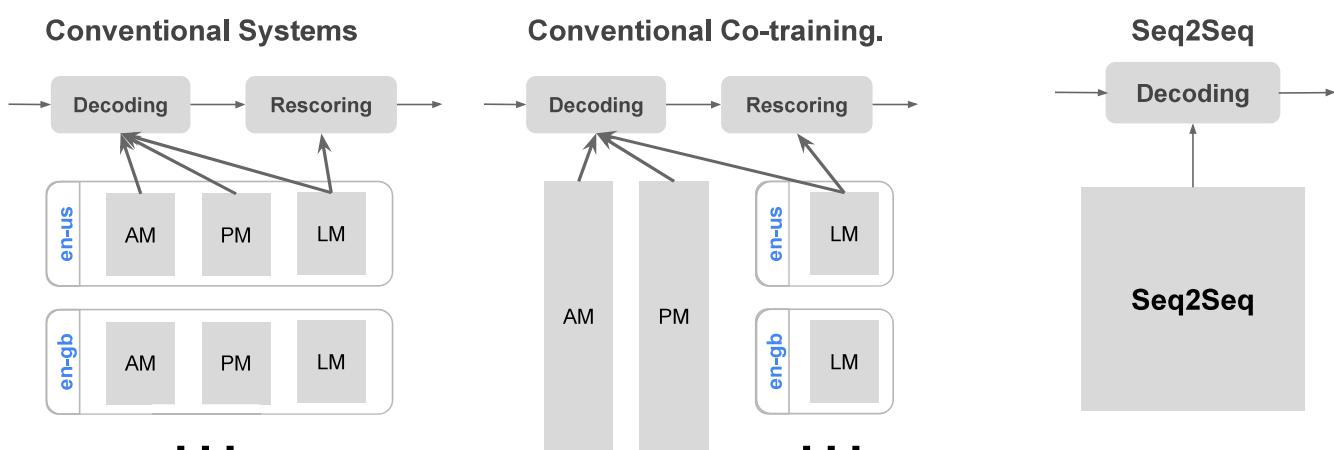
Google



Multi-Dialect Speech Recognition With A Single Seq2Seq Model

[Li et al., 2018]
[\[Toshniwal et al., 2018\]](#)

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 <div style="border: 1px solid black; padding: 5px;"> phoneme lexicon text normalization LM </div>	data × N

Google

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.

Google

Multi-Dialect LAS

Google

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 U w o r l d <eos>"
 - LID errors may affect ASR performance
 - ASR first, then LID
 - "<sos> h e l l o U w o r l d <en-gb> <eos>"
 - ASR prediction is not dependent on LID prediction, not suffering from LID errors

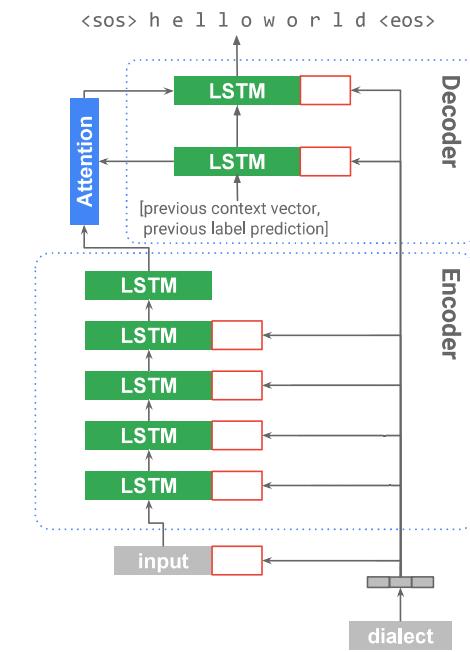
Google

Dialect as Input Features

- Passing the dialect information as additional features

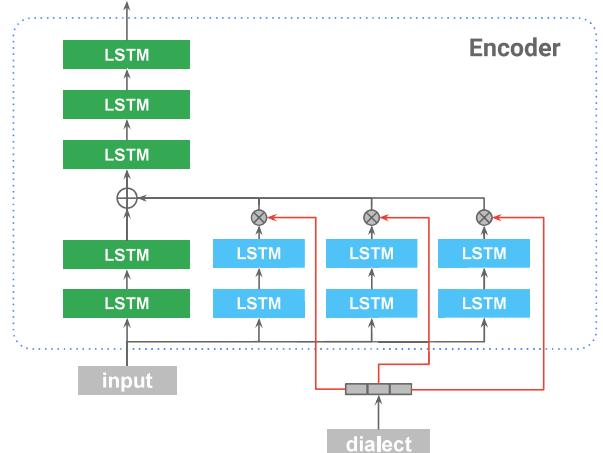
components	variations
encoders	→ acoustic
decoders	→ lexicon and language

Google



Dialect Information as Cluster Coefficients

- Cluster Adaptive Training (CAT) [1] coefficients
 - more flexible model architectures
 - larger capacity in variation modeling
 - but increased model parameters

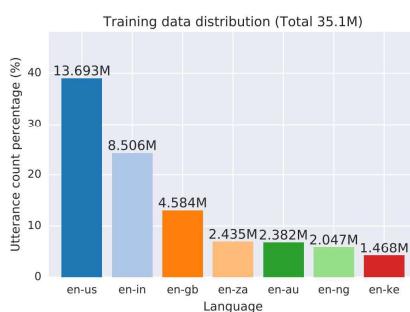


Experimental Evaluations

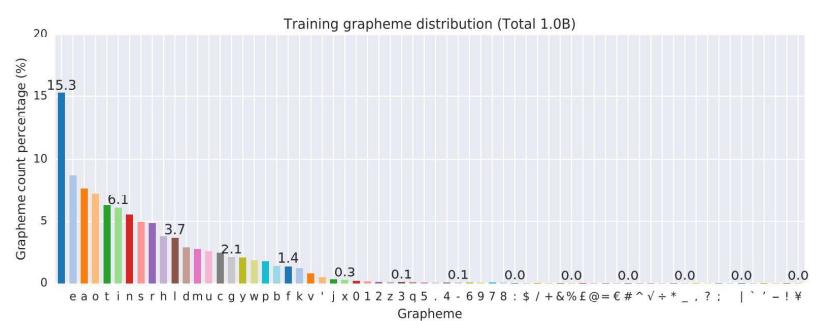
Google

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

Google

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

Google

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

Example target sequence

★ **ASR first** is better

LID first	<sos> <en-gb> h e l l o U w o r l d <eos>
ASR first	<sos> h e l l o U w o r l d <en-gb> <eos>

Google

LAS With Dialect as Input Features

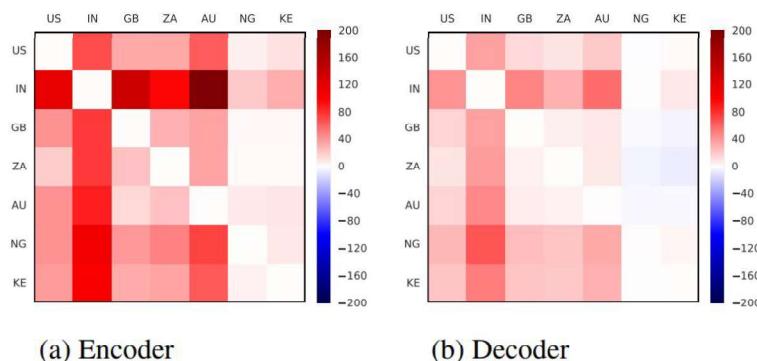
Dialect	US	IN	GB	ZA	AU	NG	KE
Baseline (dialect-dep.)	9.7	16.2	12.7	11.0	12.1	33.4	19.0
encoder	1-hot	9.6	16.4	11.8	10.6	10.7	31.6
	emb.	9.6	16.7	12.0	10.6	10.8	32.5
decoder	1-hot	9.4	16.2	11.3	10.8	10.9	32.8
	emb.	9.4	16.2	11.2	10.6	11.1	32.9
both	1-hot	9.1	15.7	11.5	10.0	10.1	31.3
							17.4

- ★ dialect 1-hot and embedding (emb.) performs **similarly**
- ★ feeding dialect to **both encoder and decoder** gives the largest gains

Google

LAS With Dialect as Input Features

Figure: Feeding different dialect vectors (rows) to the LAS encoder and decoder 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

Google

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)
x	x	x	1	22
<en-gb>: [0, 1, 0, 0, 0, 0, 0]	✓	x	19	4
<en-gb>: [0, 1, 0, 0, 0, 0, 0]	x	✓	0	25
<en-us>: [1, 0, 0, 0, 0, 0, 0]	x	✓	24	0

★ dialect vector helps encoder to normalize accent variations

★ dialect vector helps decoder to learn dialect-specific lexicons

Google

LAS With Dialect as CAT coefficients

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
input features (encoder)	1-hot	9.6	16.4	11.8	10.6	10.7	31.6
CAT coeff.	1-hot emb.	9.9 9.4	17.0 16.1	12.1 11.7	11.0 10.6	11.6 10.6	32.5 32.9

★ dialect as CAT coefficients is much better than as inputs

★ but with large model params increase (160K vs. 3M)

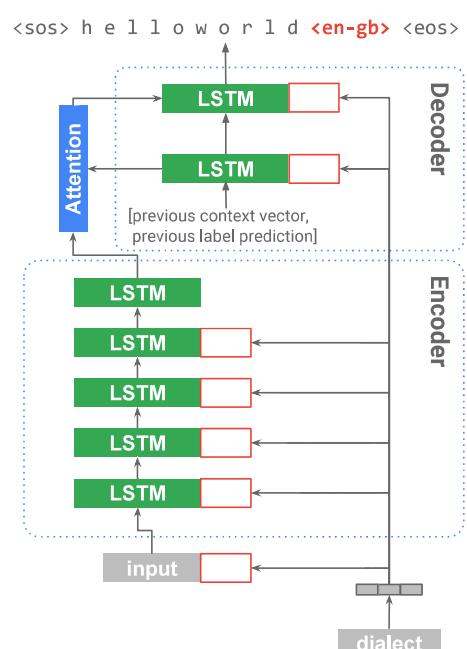
Google

Final Multi-Dialect LAS

Google

Final Multi-Dialect LAS

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



Google

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

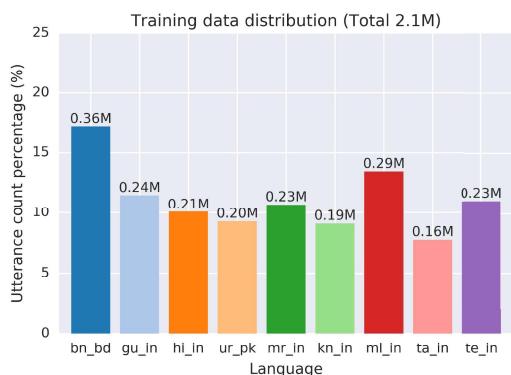
★ small gains when combining input and output

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

Google

IndicX - Task

- **9 Indian languages:** bn_bd, gu_in, hi_in, ur_pk, mr_in, kn_in, ml_in, ta_in, te_in
- **large script variations**

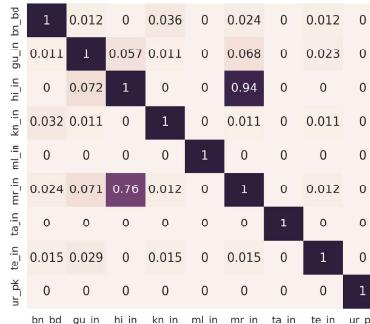


- Bengali (bn_bd) - বাঙালি ভাষার উদ্দেশকে বলতেন
- Gujarati (gu_in) - ગુજરાતી અંદર ન માટે અને બહાર પણ ન માટે
- Hindi (hi_in) - पहले वीडियोग्राफी होनी
- Kannada (kn_in) - ಮುಖದ ಮಧ್ಯದಲ್ಲಿರು ಹಿಂದೆ
- Malayalam (ml_in) - മലായിൽക്കും അവരുടെ വാക്കുകളിലുടെ അവരെ
- Marathi (mr_in) - श्रीकृष्णाच्या गोकुळातल्या
- Tamil (ta_in) - இது ஒரு நகராட்சியாகும்
- Telugu (te_in) - ఈ పేజీని 'తరువ්వ' చేయకമుందు ഇവිலో పెడదామా
- Urdu (ur_pk) - شیخ عبدالرحیم گرہوڑی جو کلام مصنف -

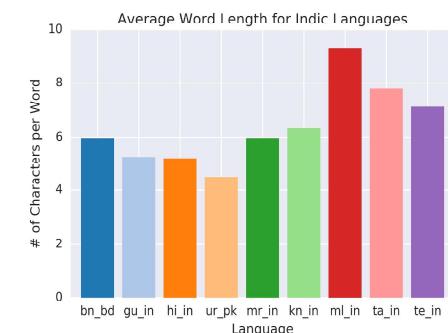
Google

[1] S. Toshniwal, T.N. Sainath, R.J. Weiss, B. Li et.al [Multilingual speech recognition with a single end-to-end model](#), ICASSP 2018

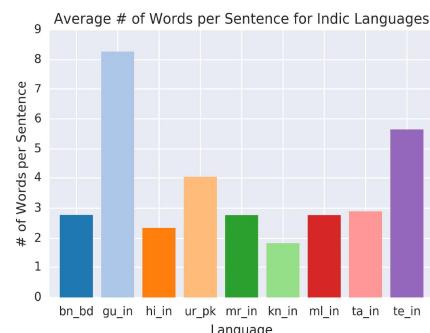
IndicX - Task



★ large variations in graphemes
★ totally 964 unique graphemes



★ lexicon variations

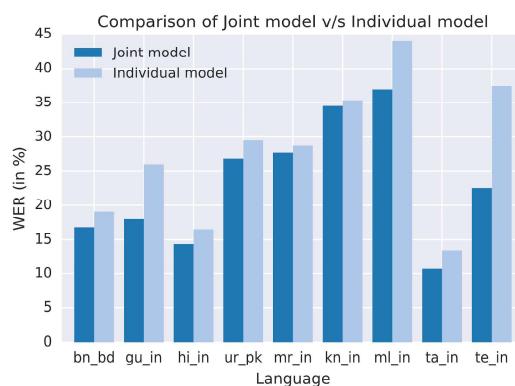


★ LM variations

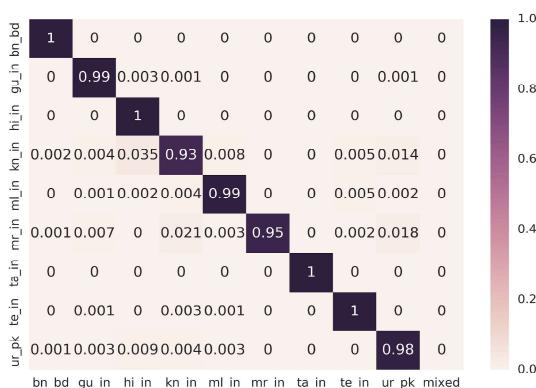
Google

[1] S. Toshniwal, T.N. Sainath, R.J. Weiss, B. Li et.al [Multilingual speech recognition with a single end-to-end model](#), ICASSP 2018

IndicX - Co-training



★ co-trained model is consistently better



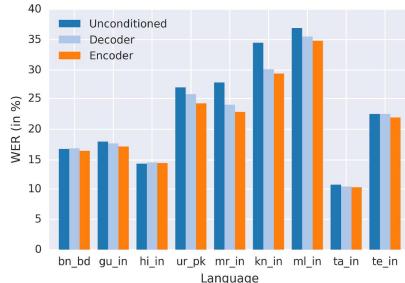
★ co-trained model chooses the right script

★ tested multitask learning (LID and ASR), not helpful
★ the model cannot do code switching, faithful to one language

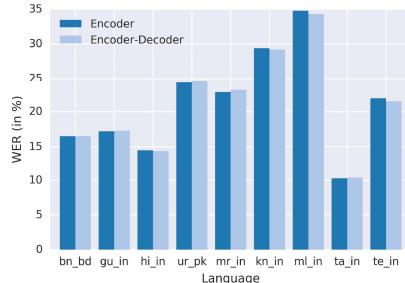
Google

[1] S. Toshniwal, T.N. Sainath, R.J. Weiss, B. Li et.al [Multilingual speech recognition with a single end-to-end model](#), ICASSP 2018

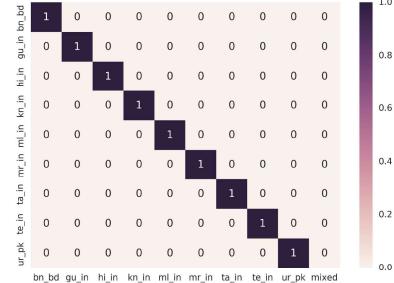
IndicX - Co-training with LID



★ helps more on encoder



★ feeds to encoder only is sufficient



★ chooses the correct script

★ faithful to language ID,
wrong ID leads to wrong script

Google

[1] S. Toshniwal, T.N. Sainath, R.J. Weiss, B. Li et.al [Multilingual speech recognition with a single end-to-end model](#), ICASSP 2018

Summary and Open questions

- Summary
 - End-to-end models can be competitive to production
 - We now have models which can endpoint, are streaming, can do contextual biasing
- Open Questions
 - How to inject pronunciations?
 - How to handle long-tail problems (numerics)?
- Expanding to new domains
 - Speech To Parse
 - Audio-visual

Google

176

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.
- [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. 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.

Google

References

- [Graves, 2012] A. Graves, "Sequence Transduction with Recurrent Neural Networks," Proc. of ICML Representation Learning Worksop, 2012.
- [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.
- [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. ICASSP, 2018.

Google

References

- [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.
- [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.
- [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.

Google

References

- [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," CoRR, vol. abs/1808.02480, 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] Hasim Sak, Andrew Senior, Kanishka Rao, Francoise Beaufays, "Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition," Proc. of Interspeech, 2015.
- [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.

Google

References

- [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.
- [Sriram et al., 2017] A. Sriram, H. Jun, S. Satheesh, A. Coates, "Cold Fusion: Training Seq2Seq Models Together with Language Models," CoRR, vol. abs/1708.06426, 2017.
- [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," in Proc. of ICASSP, 2013.
- [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.
- [Wiseman and Rush, 2016] S. Wiseman and A. M. Rush, "Sequence-to-Sequence Learning as Beam Search Optimization," Proc. of EMNLP, 2016.

Google