Introduction To Seq2Seq

10/19/2018

(adapted from ICML 2017 tutorial)

Outline

- An Overview of seq2seq (~25 minutes)
  - Applications
- Attention (~20 minutes)
  - Applications
- Break (15 mins)
- Loss functions (~15 minutes)
- Advanced Topics
  - Autoregressive Models (~5 mins)
  - Online Seq2Seq (~10 mins)
- New Architectures (~10 mins)
- Questions, discussion, etc. (15 minutes)

Lip Reading

Channel	Series name	# hours	# sent.
BBC 1 HD	News†	1,584	50,493
BBC 1 HD	Breakfast	1,997	29,862
BBC 1 HD	Newsnight	590	17,004
BBC 2 HD	World News	194	3,504
BBC 2 HD	Question Time	323	11,695
BBC 4 HD	World Today	272	5,558
All		4,960	118,116

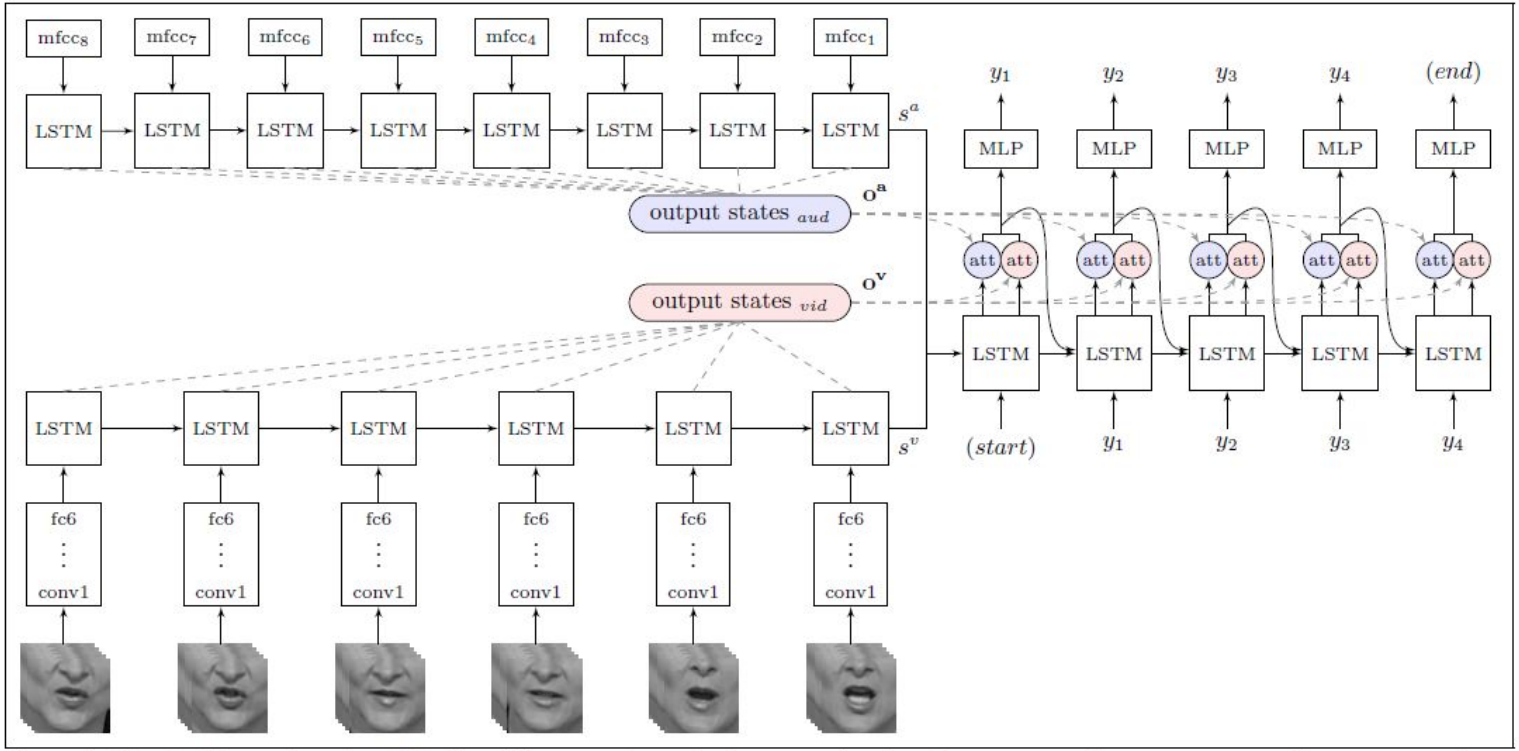
http://www.robots.ox.ac.uk/~vgg/data/lip_reading/

1. Chung, J., et al. "Lip reading sentences in the wild." *CVPR* (2017).
2. Assael, Y., et al. "Lipnet: Sentence-level lipreading." *arxiv* (2016).

Lip Reading

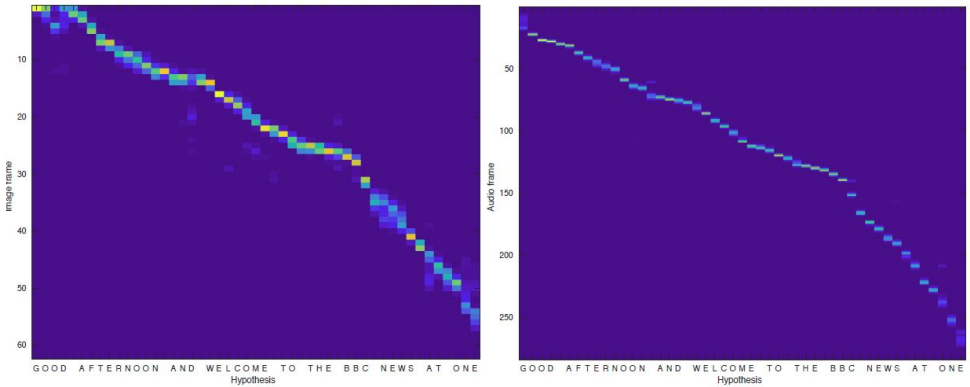
Separate embedding and attention for audio and visual streams

visual: cnn + lstm



Chung, J., et al. "Lip reading sentences in the wild." *CVPR* (2017).

Lip Reading



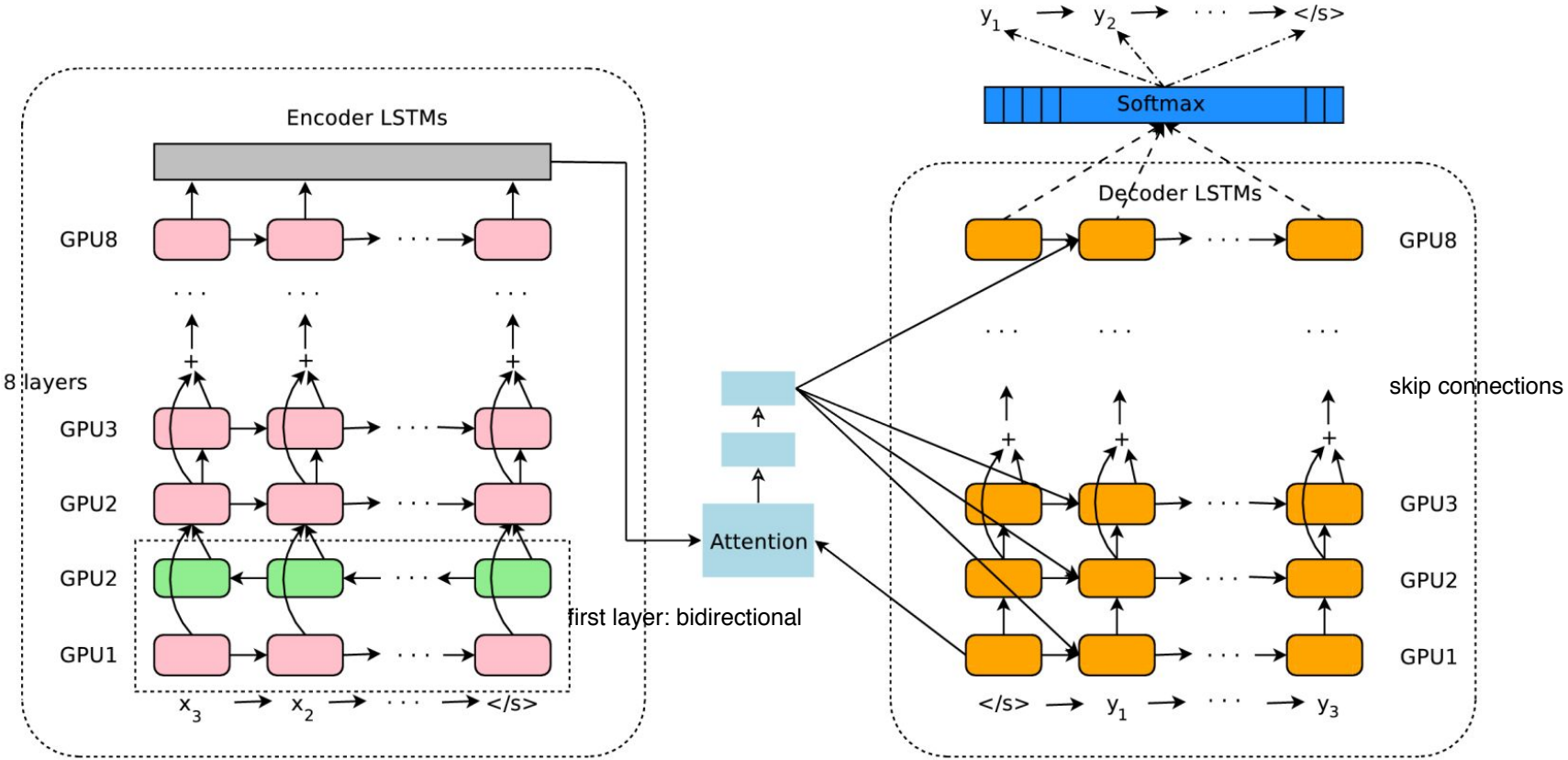
monotonic

better accuracy than human

Method	SNR	CER	WER	BLEU†
Lips only				
Professional‡	-	58.7%	73.8%	23.8
WAS	-	59.9%	76.5%	35.6
WAS+CL	-	47.1%	61.1%	46.9
WAS+CL+SS	-	42.4%	58.1%	50.0
WAS+CL+SS+BS	-	39.5%	50.2%	54.9

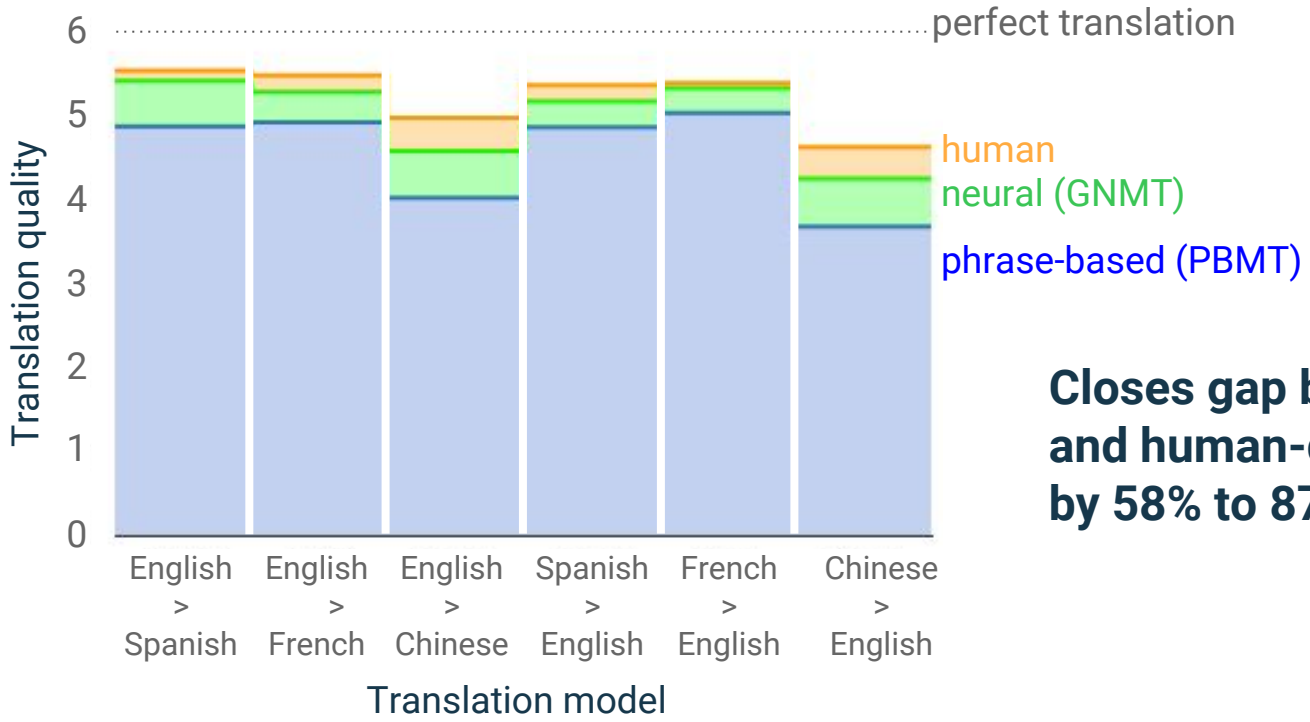
Chung, J., et al. "Lip reading sentences in the wild." *CVPR* (2017).

Google Neural Machine Translation System



Wu, Y., et al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation." *arxiv (2016).*

Google Neural Machine Translation



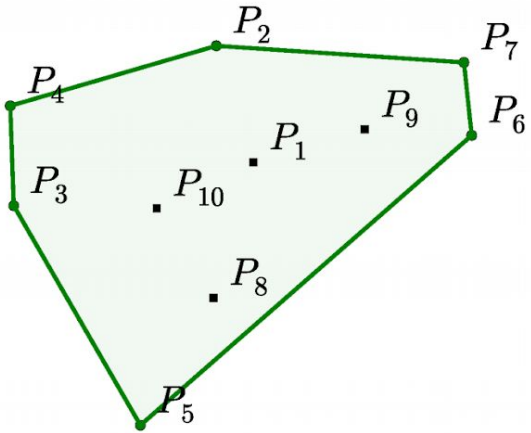
Closes gap between old system and human-quality translation by 58% to 87%

research.googleblog.com/2016/09/a-neural-network-for-machine.html

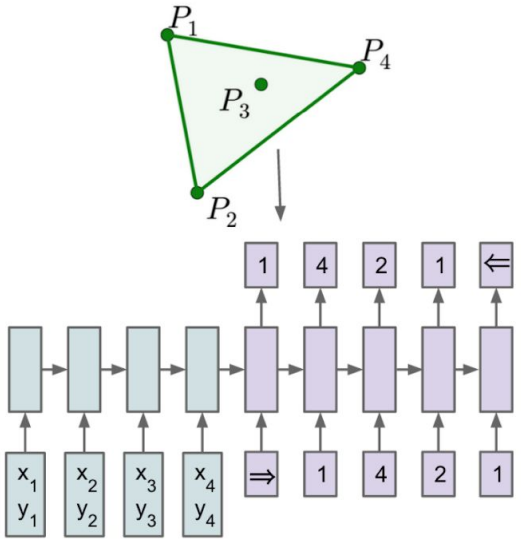
Pointer Networks

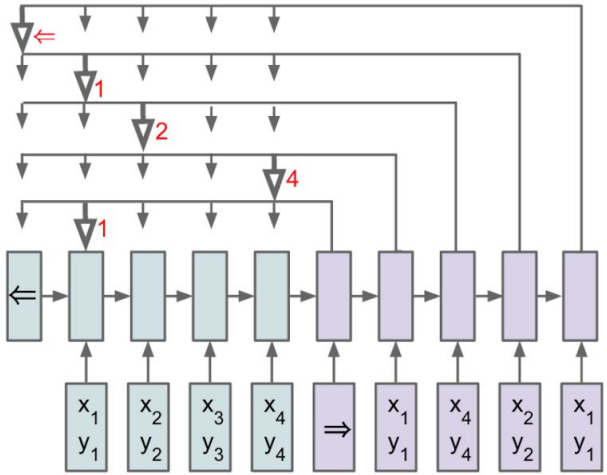
input: set of points
output: point in set

- Adaptation of sequence to sequence with attention for self referential outputs



Convex Hull

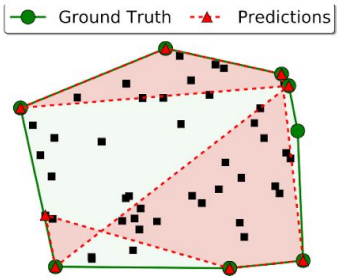




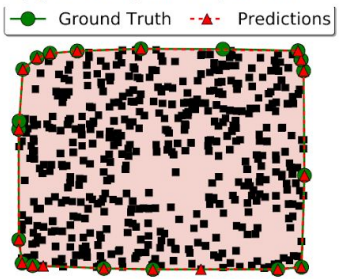
Vinyals, O., Fortunato, M., Jaitly. N. “Pointer networks.” *NIPS (2015).*

Pointer Networks

Convex hull

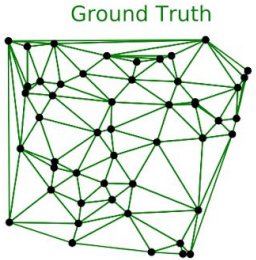


(a) LSTM, m=50, n=50

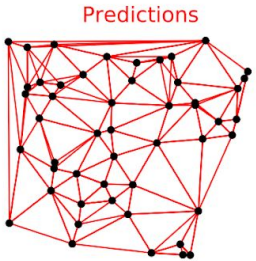


(d) Ptr-Net, m=5-50, n=500

Delauney Triangulation

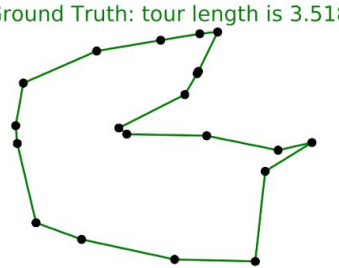


(b) Truth, n=50

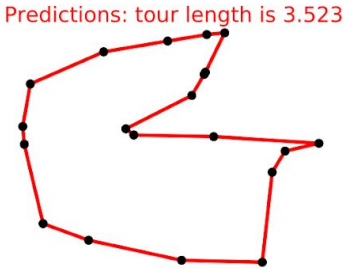


(e) Ptr-Net , m=50, n=50

Travelling Salesman Problem



(c) Truth, n=20



(f) Ptr-Net , m=5-20, n=20

- Merity, Stephen, et al. "Pointer sentinel mixture models." *arXiv preprint arXiv:1609.07843* (2016).
- Kadlec, Rudolf, et al. "Text understanding with the attention sum reader network." *arXiv preprint arXiv:1603.01547* (2016).
- Gulchere, C., et. al. “Pointing the unknown words” *ACL (2016)*

Loss Function

Loss Functions

- Cross Entropy
- Scheduled Sampling [1]
- Expected Loss [2]
- Augmented Loss [3]
- Sequence to Sequence as a beam search optimization [4]
- ~~Learning decoders with different loss function [5]~~

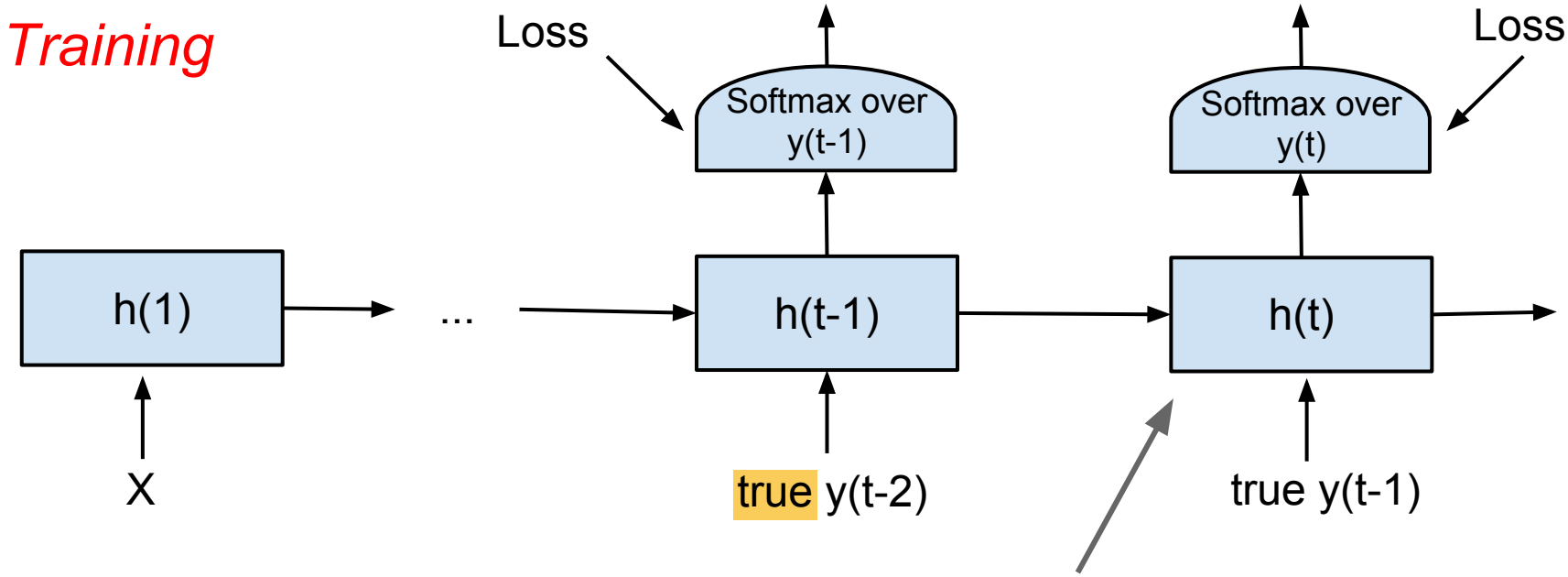
1. Bengio, S., et al. “Scheduled sampling for sequence prediction with recurrent neural networks.” *NIPS (2015).*
2. Ranzato, M., et al. “Sequence level training with recurrent neural networks.” *ICLR (2016).*
3. Norouzi, M., et al. “Reward augmented maximum likelihood for neural structured prediction.” *NIPS (2016).*
4. Wiseman, S., Rush, A. “Sequence-to-sequence learning as beam-search optimization.” *EMLP (2016).*
5. Gu, J, Cho, K and Li, V.O.K. "Trainable greedy decoding for neural machine translation." *arXiv preprint arXiv:1702.02429* (2017).

Cross Entropy (Negative Log Likelihood) Loss

- Log Likelihood, by chain rule is sum of next step log likelihoods $\log p(\mathbf{y}|\mathbf{x}) = \sum_{i=1}^{N} \log p(y_i|y_{<i}, \mathbf{x})$

- Supervised classification for each time step
  - depends on input, past outputs, which are known during training

Training and Inference Mismatch

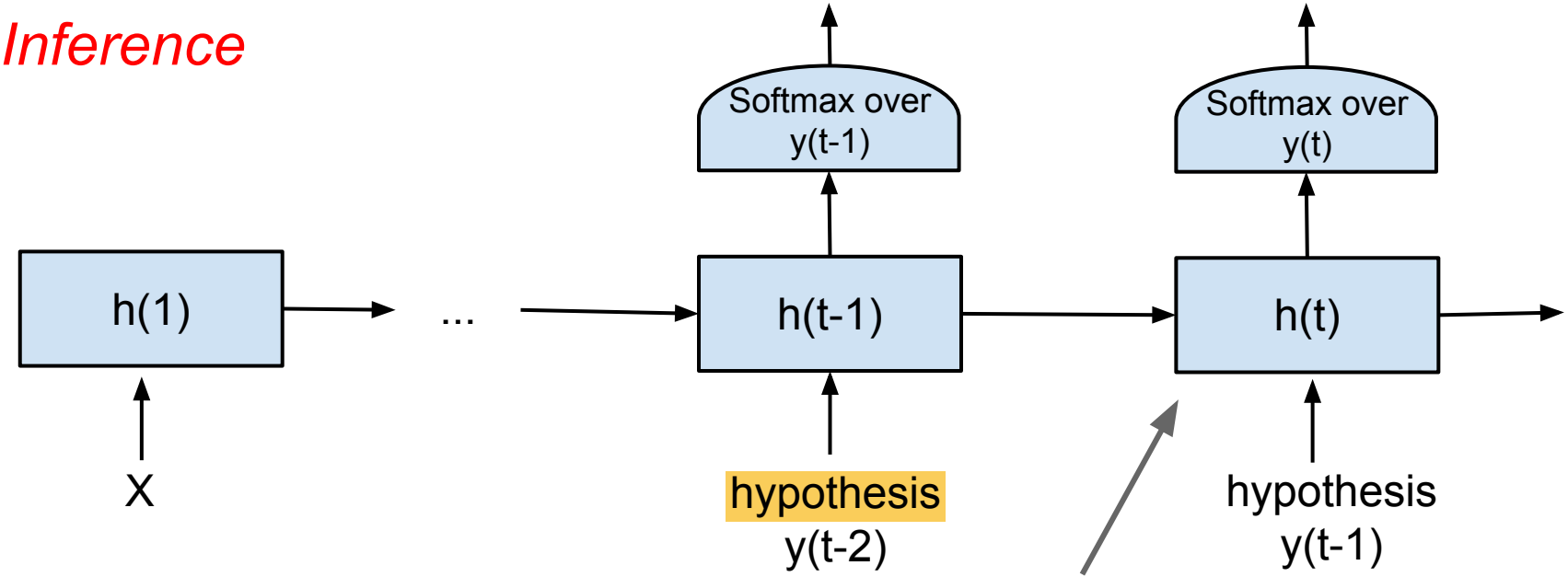


$$P(y_t|h_t) \text{ with } h_t = f(h_{t-1}, y_{t-1}; \theta)$$

Bengio, S., et al. “Scheduled sampling for sequence prediction with recurrent neural networks.” *NIPS (2015).*

Training and Inference Mismatch

Inference

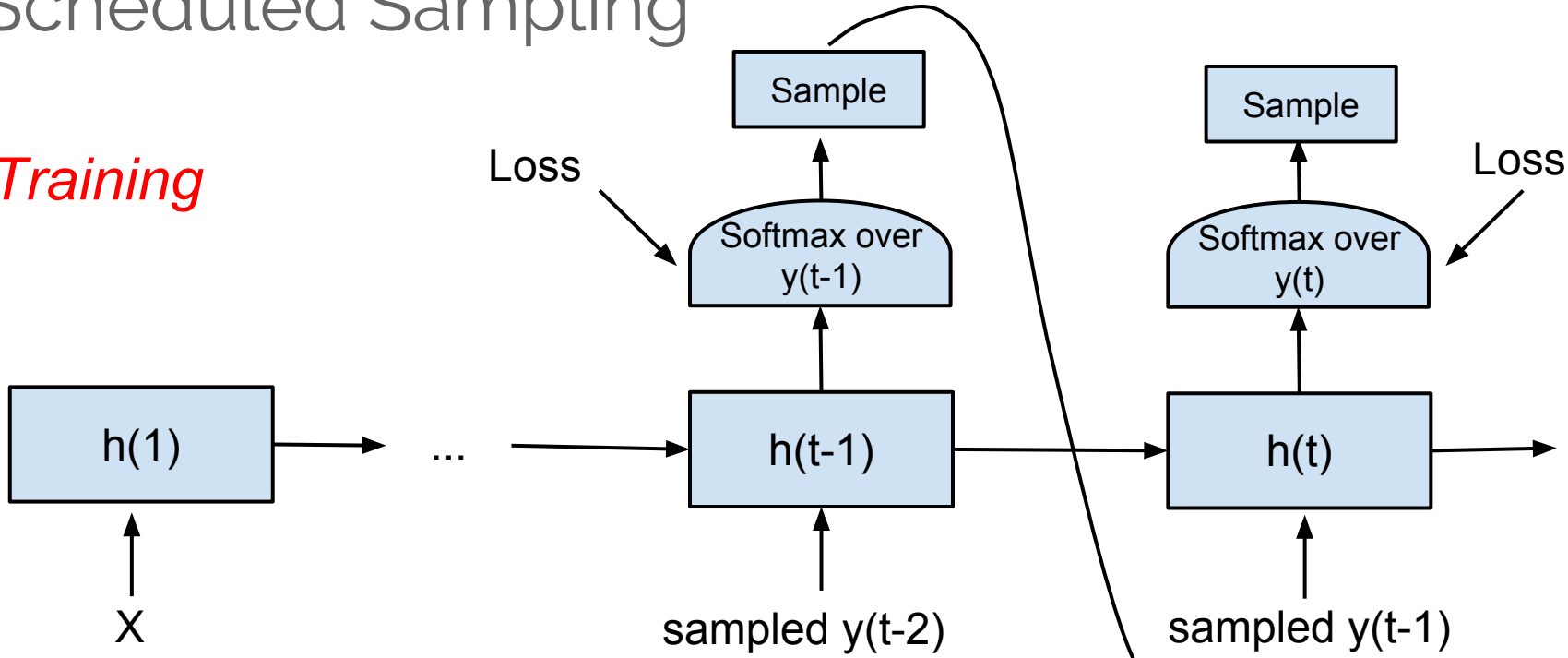


$$P(y_t|h_t) \text{ with } h_t = f(h_{t-1}, y_{t-1}; \theta)$$

Bengio, S., et al. “Scheduled sampling for sequence prediction with recurrent neural networks.” *NIPS (2015).*

Scheduled Sampling

Training



$$P(y_t|h_t) \text{ with } h_t = f(h_{t-1}, \hat{y}_{t-1}; \theta)$$

Bengio, S., et al. “Scheduled sampling for sequence prediction with recurrent neural networks.” *NIPS (2015).*

Scheduled Sampling

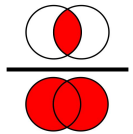
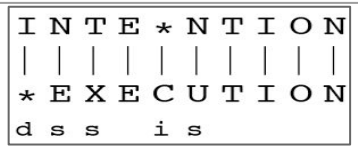
Machine Translation Model	Bleu-4	Meteor	Cider
Baseline	28.8	24.2	89.5
Baseline with dropout	28.1	23.9	87.0
Scheduled sampling	**30.6**	**24.3**	**92.1**

Parsing Model	F1
Baseline LSTM with dropout	87.00
Scheduled sampling with dropout	**88.68**

Speech Recognition Model	WER
LAS + LM Rescoring	12.6
LAS + **Sampling** + LM Rescoring	10.3

Bengio, S., et al. “Scheduled sampling for sequence prediction with recurrent neural networks.” *NIPS (2015).*

Rewards (-loss) used in Structured Prediction

TASK	REWARD	
Classification	0/1 rewards	$r(\mathbf{y}, \mathbf{y}^*) = \mathbb{1}[\mathbf{y} = \mathbf{y}^*]$
Segmentation	Intersection over Union	$r(\mathbf{y}, \mathbf{y}^*) = \cap(\mathbf{y}, \mathbf{y}^*) / \cup(\mathbf{y}, \mathbf{y}^*)$ 
Speech Recognition	Edit Distance	$r(\mathbf{y}, \mathbf{y}^*)$ = (*#d* + *#i* + *#s*) 
Machine Translation	BLEU	

Expected reward (-loss)

Given a dataset of input output pairs, $\mathcal{D} \equiv \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)*})\}_{i=1}^{N}$

learn a conditional distribution $p_\theta(\mathbf{y} \mid \mathbf{x})$ that minimizes

expected loss: $$\mathcal{L}_{\mathrm{RL}}(\boldsymbol{\theta}) = \sum_{(\mathbf{x},\mathbf{y}^*)\in\mathcal{D}} -\sum_{\mathbf{y}\in\mathcal{Y}} p_\theta(\mathbf{y} \mid x)\; r(\mathbf{y}, \mathbf{y}^*)$$

*Sample from the **model** distribution*

Difficult / Impossible to train from scratch!!

Mixed Incremental Cross-Entropy Reinforce (MIXER)

- Gradually interpolate from Cross-Entropy to Expected Loss

Data: a set of sequences with their corresponding context.
Result: RNN optimized for generation.
Initialize RNN at random and set N^{XENT}, $N^{\text{XE+R}}$ and Δ;
for $s = T, 1, -\Delta$ **do**
  if $s == T$ **then**
    train RNN for N^{XENT} epochs using XENT only;
  else
    train RNN for $N^{\text{XE+R}}$ epochs. Use XENT loss in the first s steps, and REINFORCE (sampling from the model) in the remaining $T - s$ steps;
  end
end

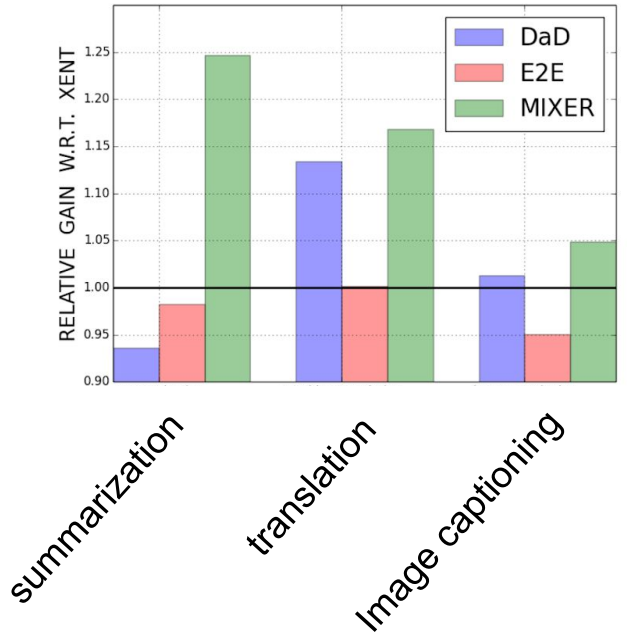
More expected loss optimization as training proceeds

Ranzato, M., et al. “Sequence level training with recurrent neural networks.” *ICLR (2016).*

Mixed Incremental Cross-Entropy Reinforce (MIXER)

TASK	XENT	DAD	E2E	MIXER
summarization	13.01	12.18	12.78	**16.22**
translation	17.74	20.12	17.77	**20.73**
image captioning	27.8	28.16	26.42	**29.16**

better than cross entropy/schedule sampling



Ranzato, M., et al. “Sequence level training with recurrent neural networks.” *ICLR (2016).*

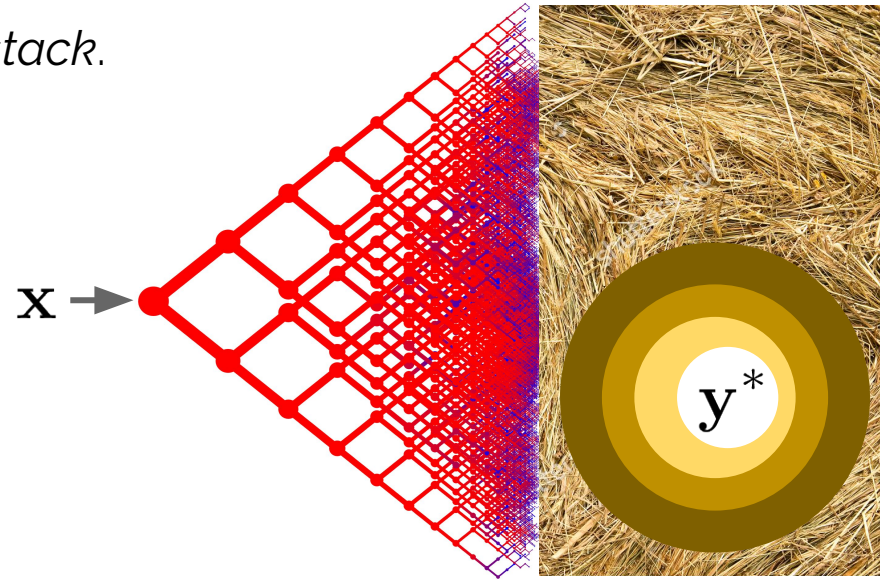
Reward Augmented Maximum Likelihood (RML)

Finding the *right output sequence*, for tasks like speech recognition or machine translation is like finding *a needle in a haystack*.

It is very risky to shoot *only* for the *true target*.

What if we expand the targets to make learning easier?

E.g. by inserting, deleting random words...



Norouzi, M., et al. “Reward augmented maximum likelihood for neural structured prediction.” *NIPS (2016).*

Reward augmented maximum likelihood (RML)

exponential sample reward

$$\mathcal{L}_{\mathrm{RML}}(\boldsymbol{\theta}; \tau) = \sum_{(\mathbf{x}, \mathbf{y}^*) \in \mathcal{D}} \left\{ - \sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{y} \mid \mathbf{y}^*; \tau) \log p_\theta(\mathbf{y} \mid \mathbf{x}) \right\}$$

Optimal $p_\theta(\mathbf{y} \mid \mathbf{x})$:

$$q(\mathbf{y} \mid \mathbf{y}^*; \tau) = \frac{1}{Z(\mathbf{y}^*, \tau)} \exp\{r(\mathbf{y}, \mathbf{y}^*)/\tau\}$$

*Sample from the **reward** distribution, irrespective of the model*

$$\mathcal{L}_{\mathrm{RML}}(\boldsymbol{\theta}; \tau) = \sum_{(\mathbf{x}, \mathbf{y}^*) \in \mathcal{D}} D_{\mathrm{KL}}\left(q(\mathbf{y} \mid \mathbf{y}^*; \tau) \parallel p_\theta(\mathbf{y} \mid \mathbf{x})\right) + \text{constant}$$

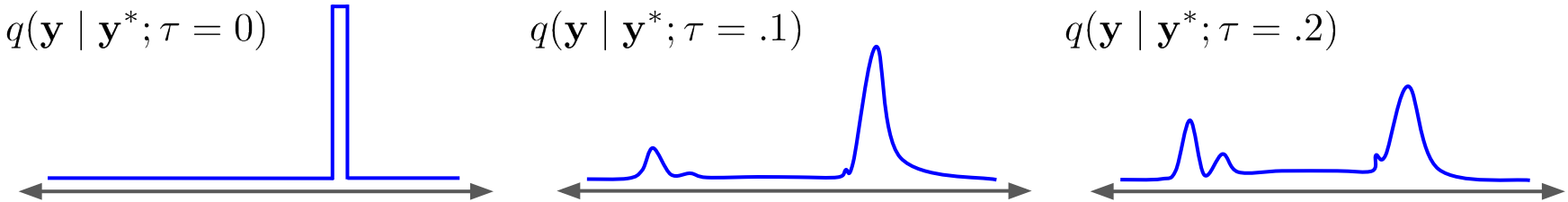
minimize difference between reward distribution and output distribution

Norouzi, M., et al. “Reward augmented maximum likelihood for neural structured prediction.” *NIPS (2016).*

RML - Impact of temperature τ

Temperature impacts spread of distribution we sample from

Cross Entropy Targets → More spread

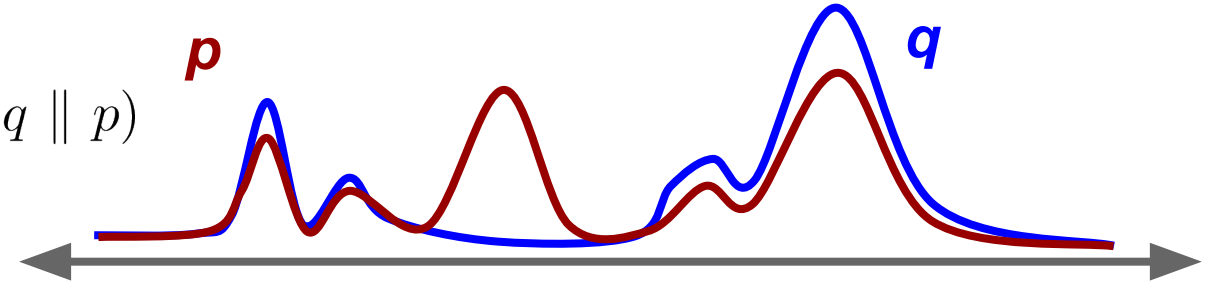


$$\mathcal{L}_{\mathrm{RML}}(\boldsymbol{\theta}; \tau) = \sum_{(\mathbf{x}, \mathbf{y}^*) \in \mathcal{D}} D_{\mathrm{KL}}\left(q(\mathbf{y} \mid \mathbf{y}^*; \tau) \parallel p_\theta(\mathbf{y} \mid \mathbf{x})\right) + \text{constant}$$

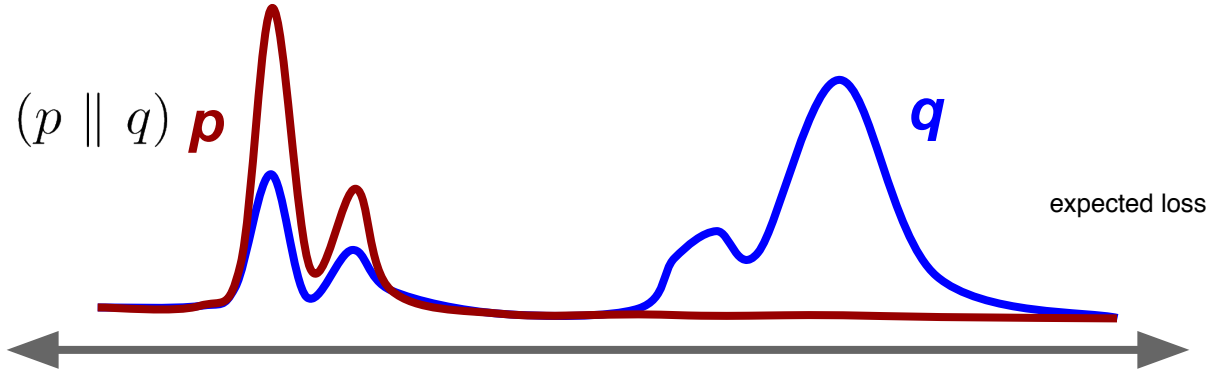
Norouzi, M., et al. “Reward augmented maximum likelihood for neural structured prediction.” *NIPS (2016).*

Contrasting RML with RL

RML $D_{\mathrm{KL}}(q \parallel p)$



RL $D_{\mathrm{KL}}(p \parallel q)$

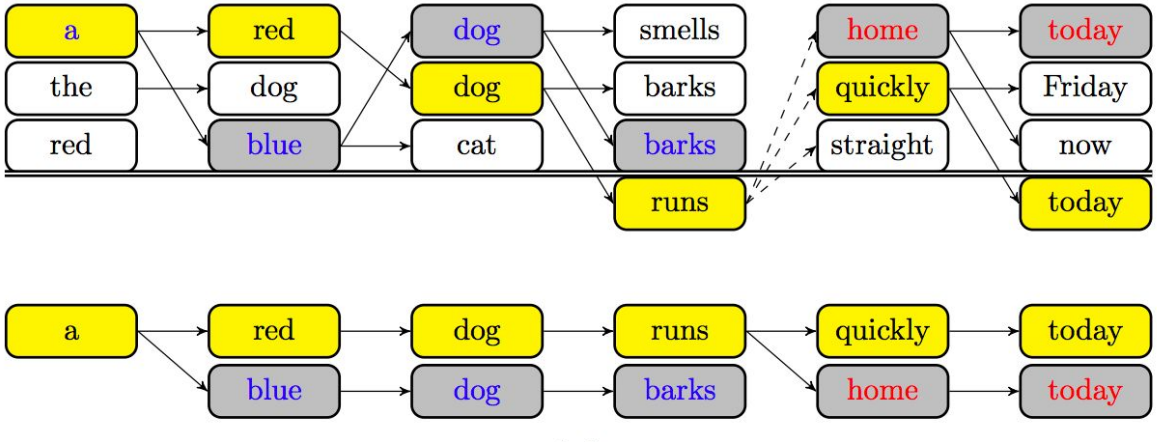


expected loss doesn't find all the modes

Norouzi, M., et al. "Reward augmented maximum likelihood for neural structured prediction." *NIPS (2016).*

Margin Loss

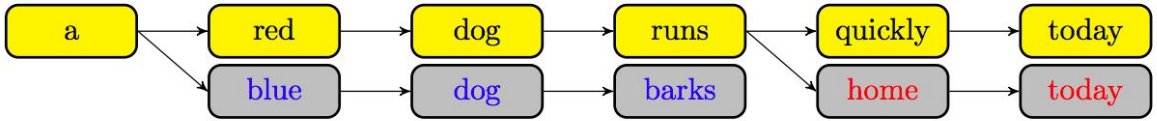
- Perform beam search until correct hypothesis falls out of the beam
- Restart beam whenever there is a violation
- Extract correct hypothesis and competing hypotheses



Wiseman, S., Rush, A. “Sequence-to-sequence learning as beam-search optimization.” *EMLP (2016).*

Margin Loss

- Add a margin score for all time steps where the correct hypothesis is not better than the Kth best hypothesis by a certain margin



$$\mathcal{L}(f) = \sum_{t=1}^{T} \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - f(y_t, \boldsymbol{h}_{t-1}) + f(\hat{y}_t^{(K)}, \hat{\boldsymbol{h}}_{t-1}^{(K)})\right]$$

Loss for error; 0 when margin constraint is satisfied

Score function for prediction of current output

Score function for prediction of K^{th} best output

Wiseman, S., Rush, A. “Sequence-to-sequence learning as beam-search optimization.” *EMLP (2016).*

Margin Loss

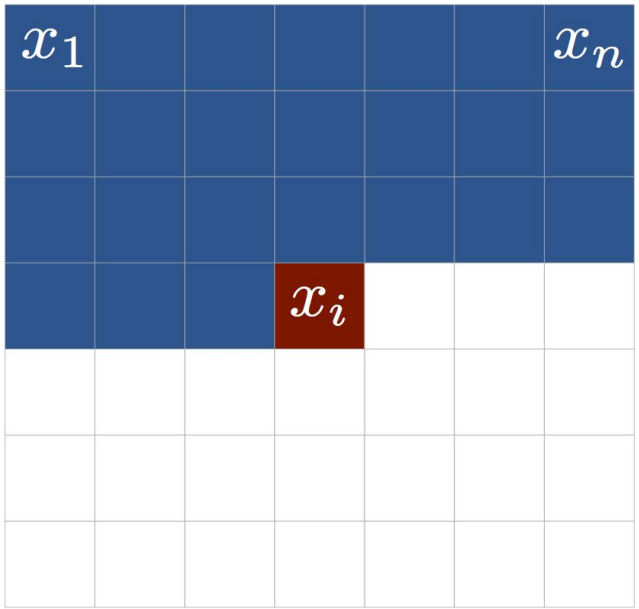
	Machine Translation (BLEU)		
	$K_{te} = 1$	$K_{te} = 5$	$K_{te} = 10$
seq2seq	22.53	24.03	23.87
BSO, SB-Δ	**23.83**	**26.36**	**25.48**
XENT	17.74	20.10	20.28
DAD	20.12	22.25	22.40
MIXER	20.73	21.81	21.83

Wiseman, S., Rush, A. “Sequence-to-sequence learning as beam-search optimization.” *EMLP (2016).*

Advanced Topics

Autoregressive Generative Models

PixelRNN - Model



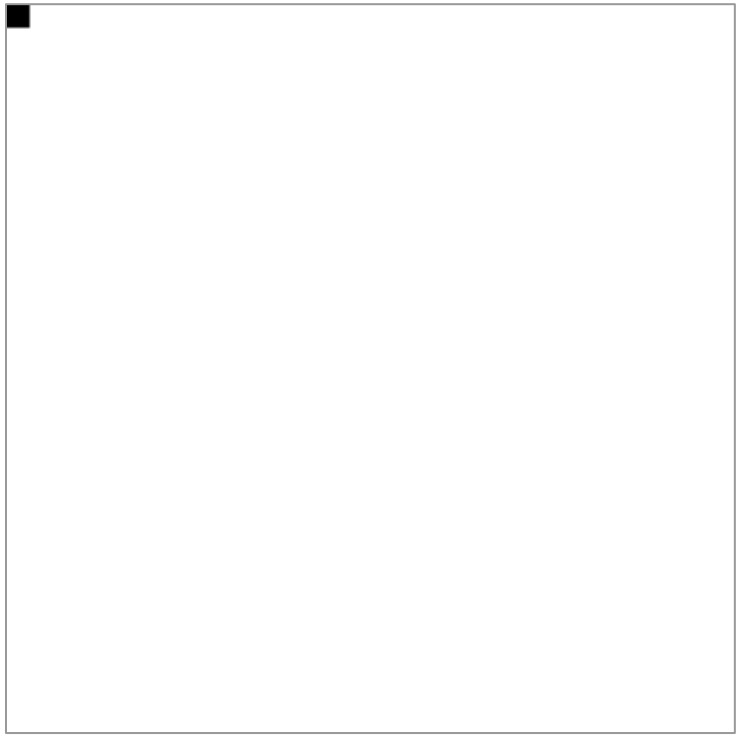
$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i | x_1, \ldots, x_{i-1})$$

- Fully visible
- Similar to language models with RNNs
- Model pixels with Softmax

van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

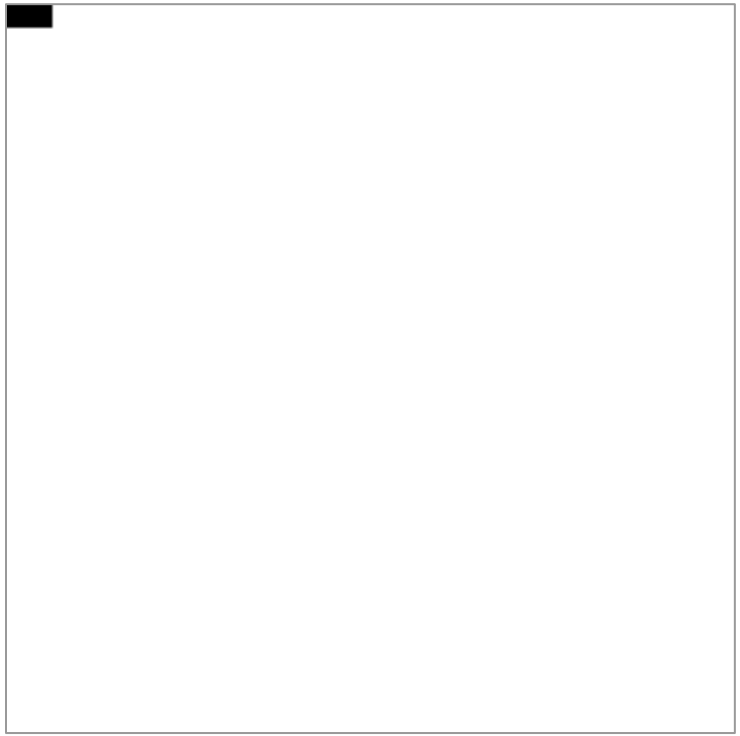
Softmax Sampling

inference: sample one pixel at a time autoregressively on previous pixels



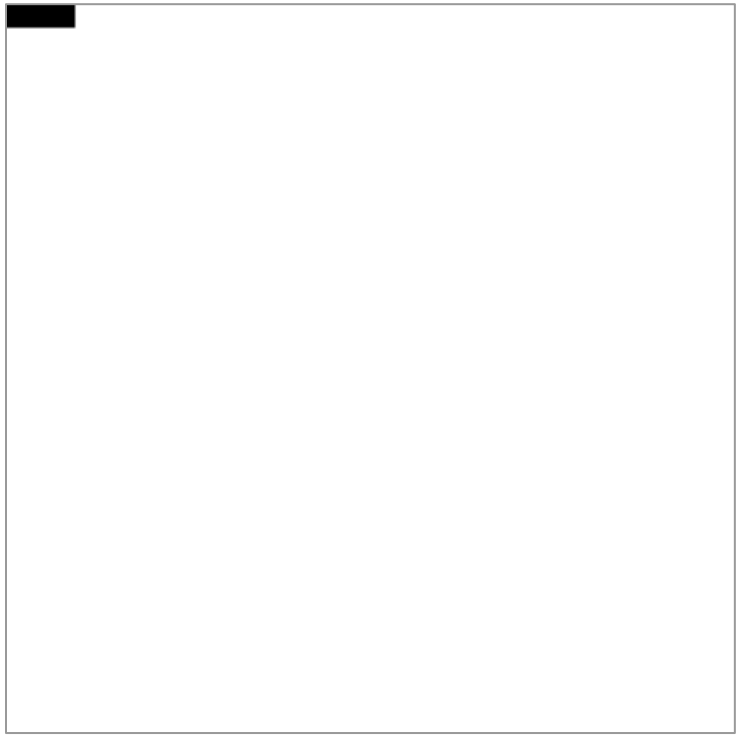
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



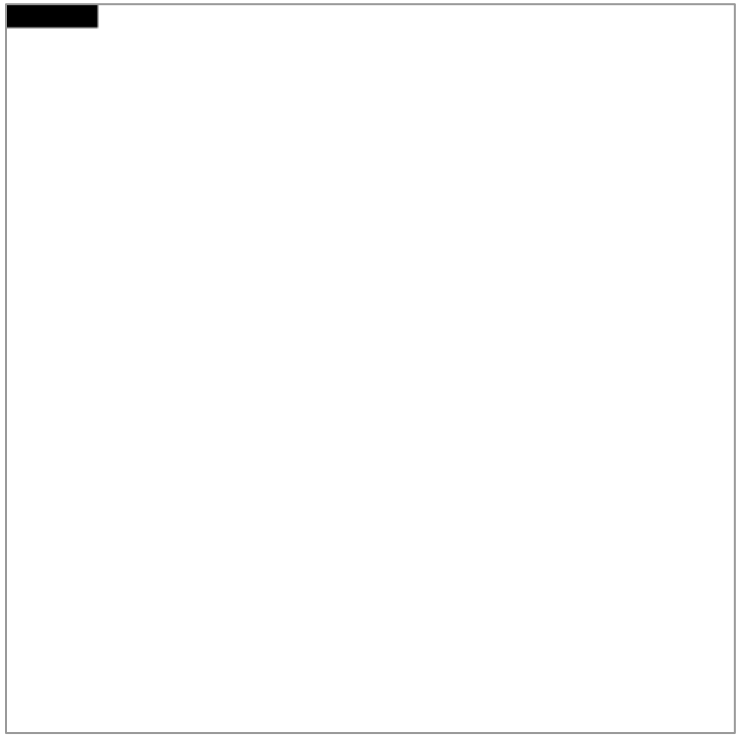
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



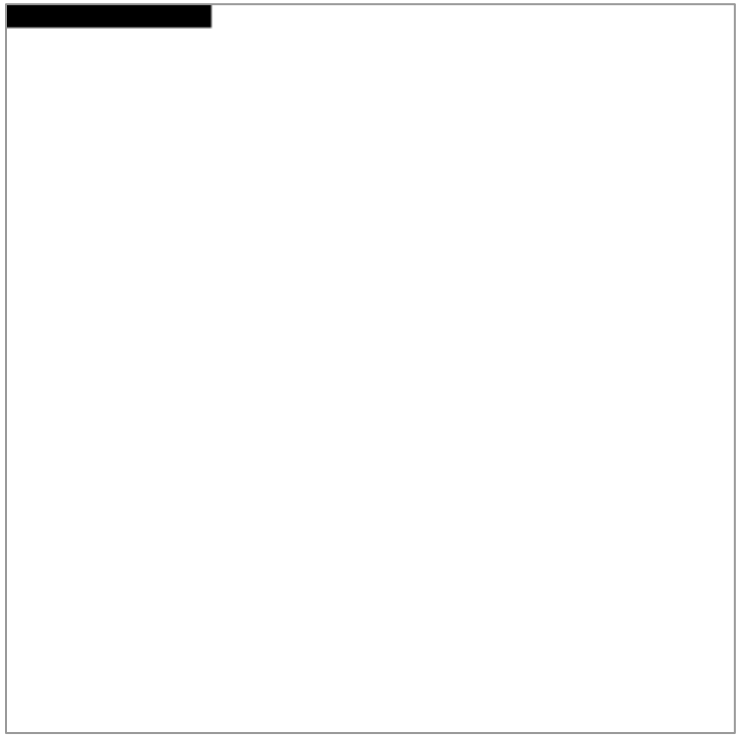
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



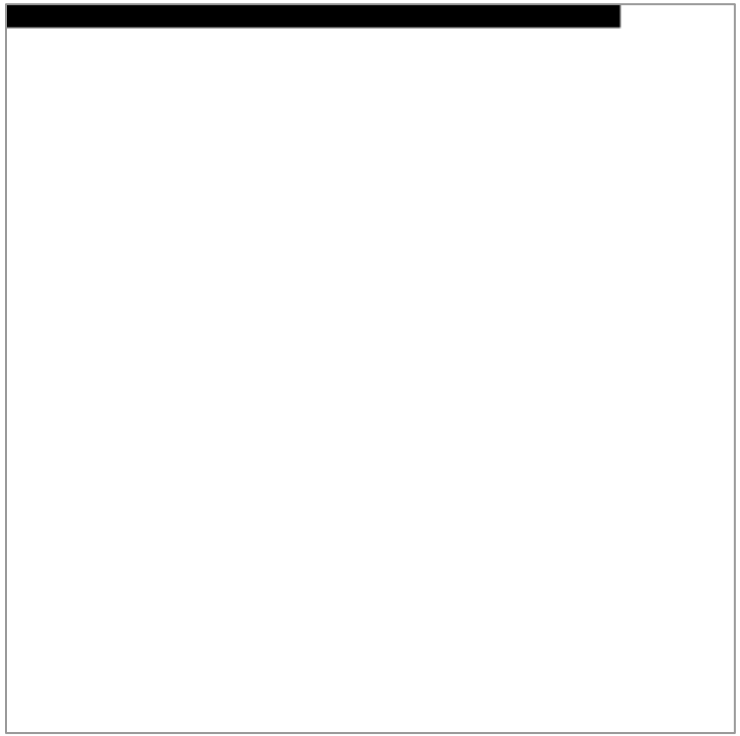
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling

van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling

van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling

van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

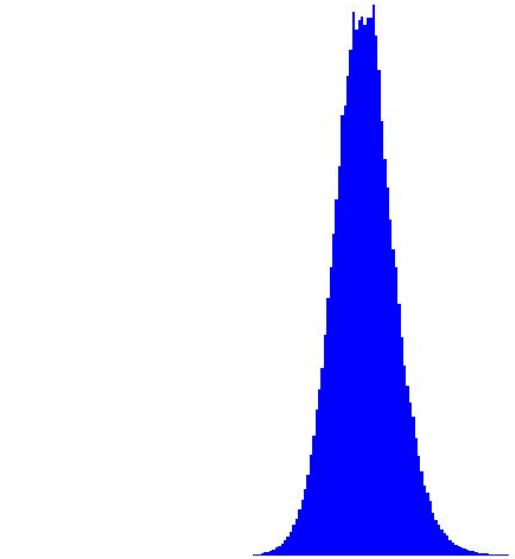
Softmax Sampling

van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling

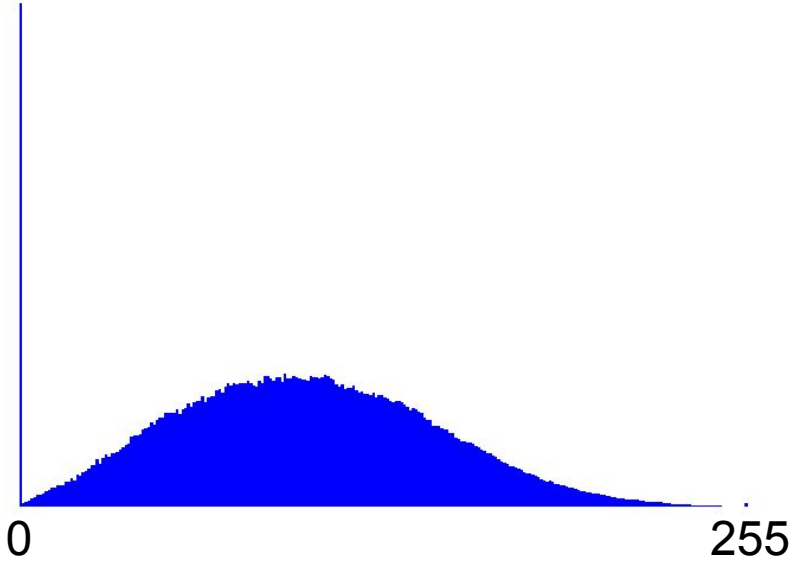
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



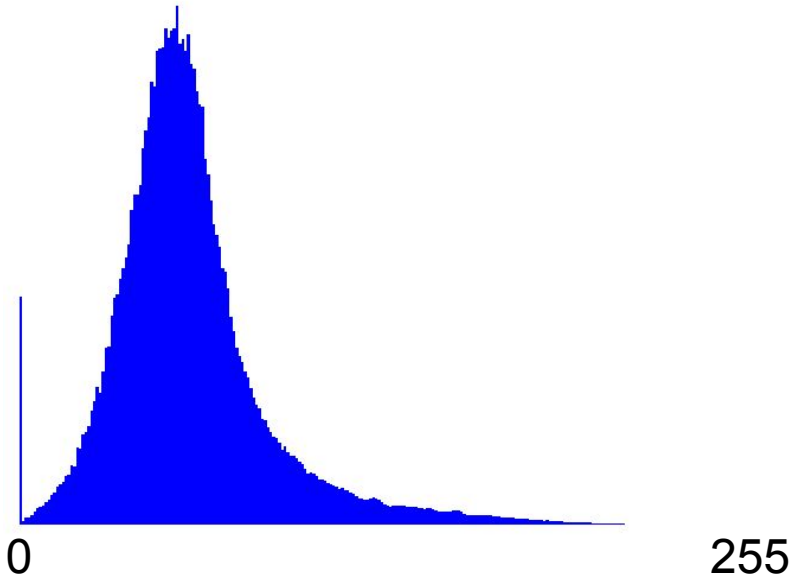
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



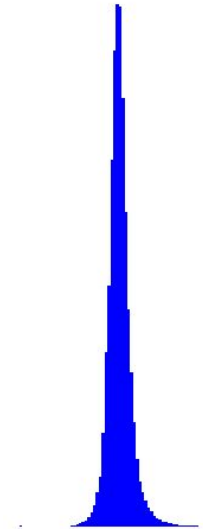
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling

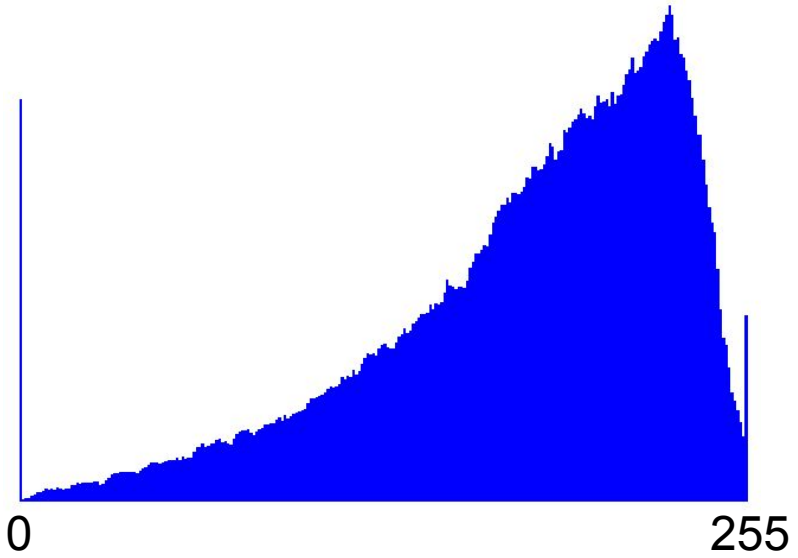


0

255

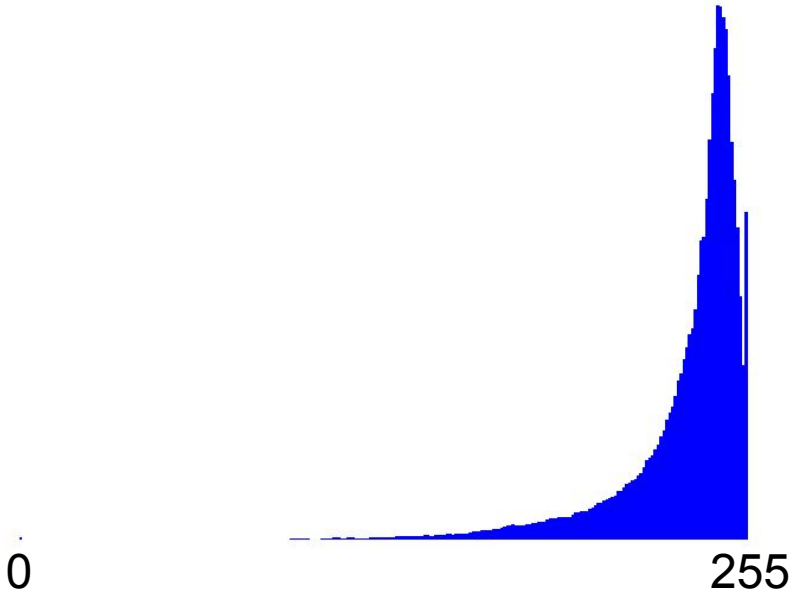
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



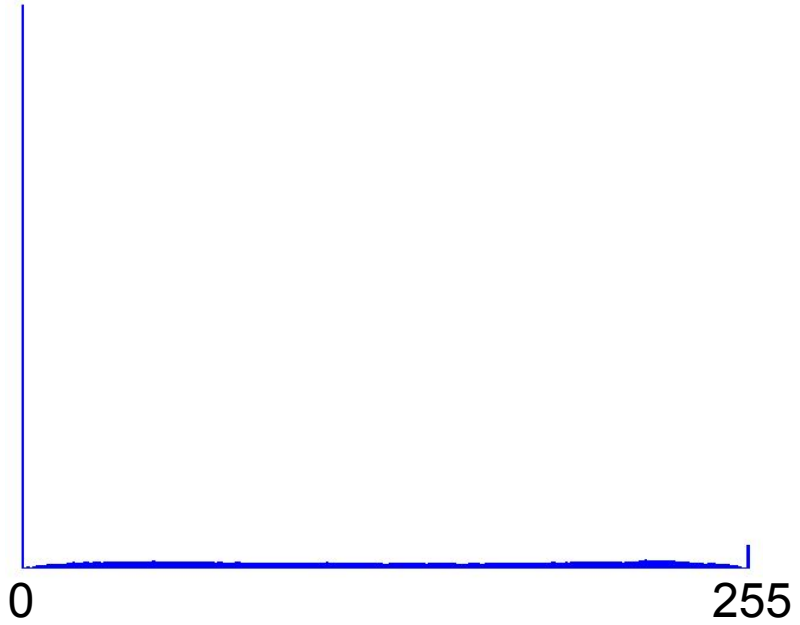
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



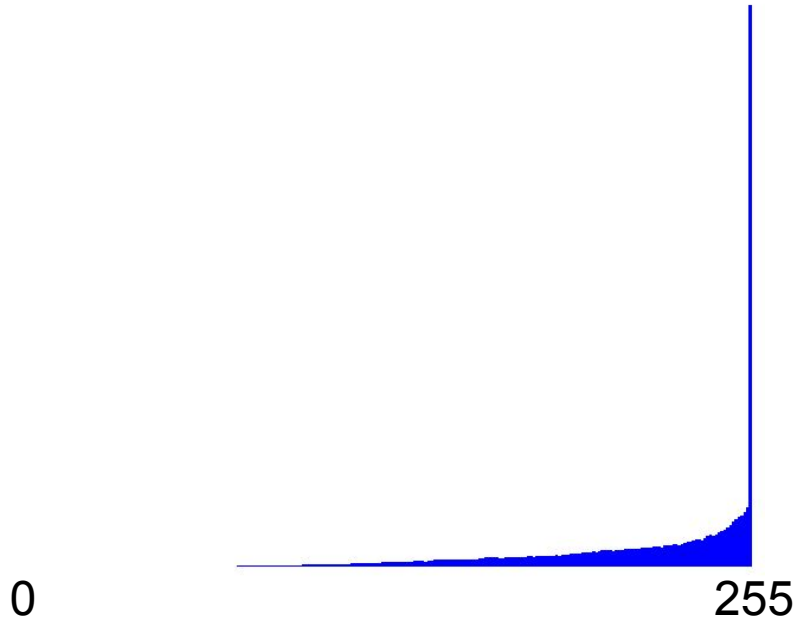
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



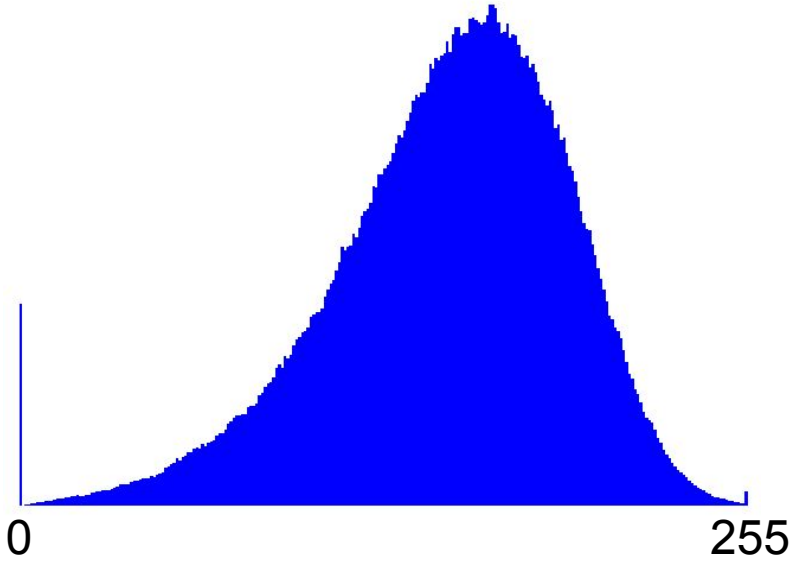
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



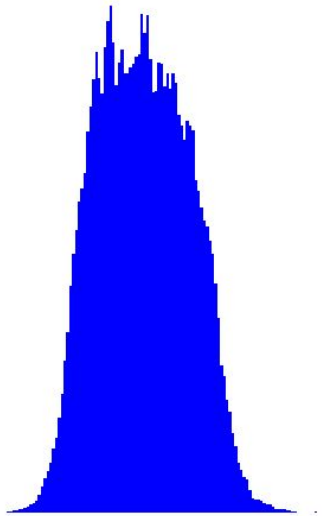
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling

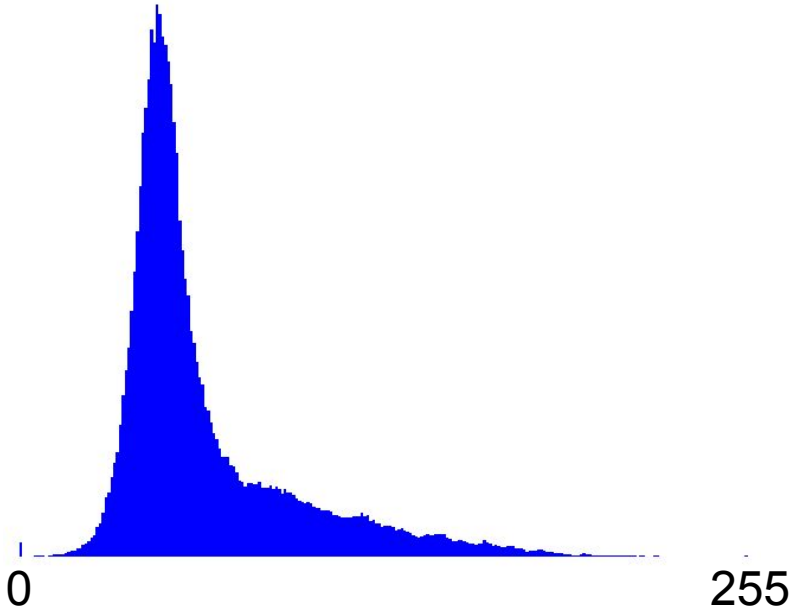


0

255

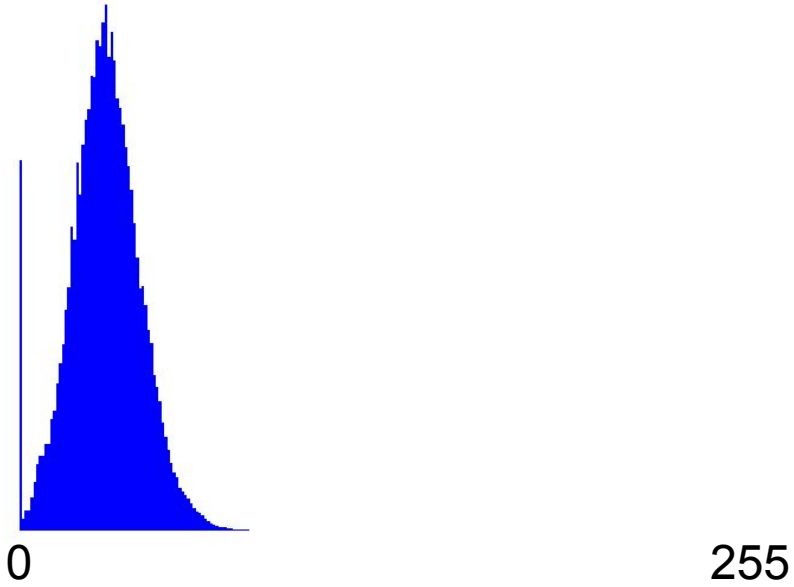
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

Softmax Sampling



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

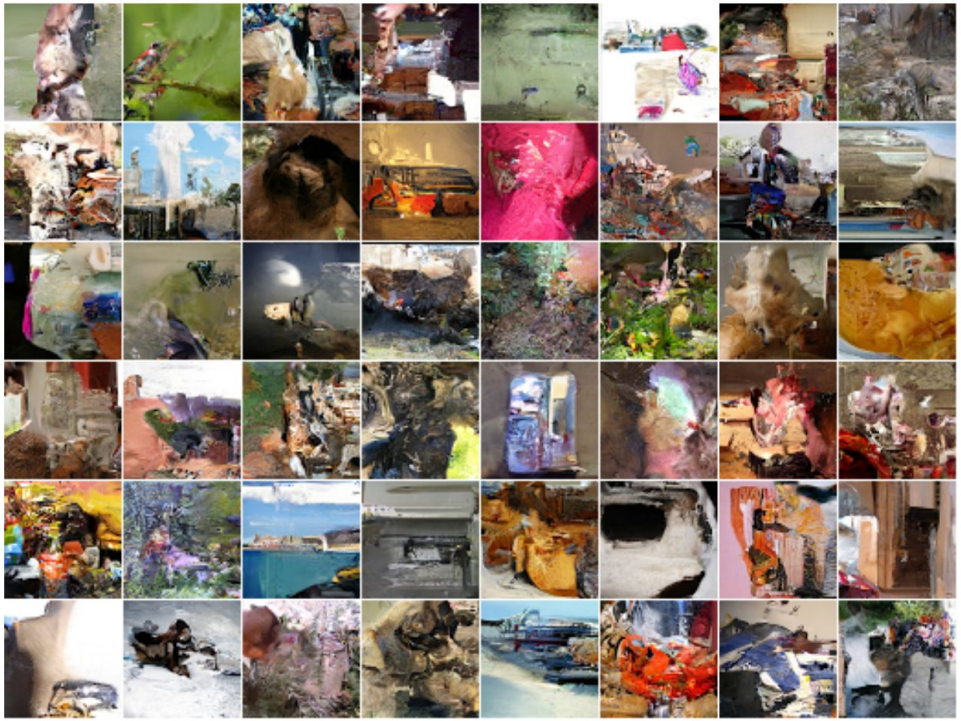
Softmax Sampling



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

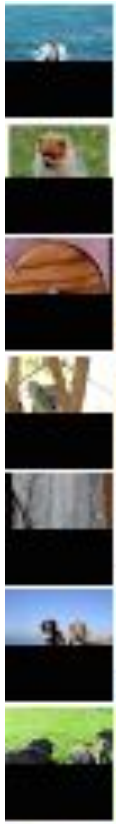
Pixel RNN

Sequence of Words == Sequence of Pixels



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

occluded



van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

image completion

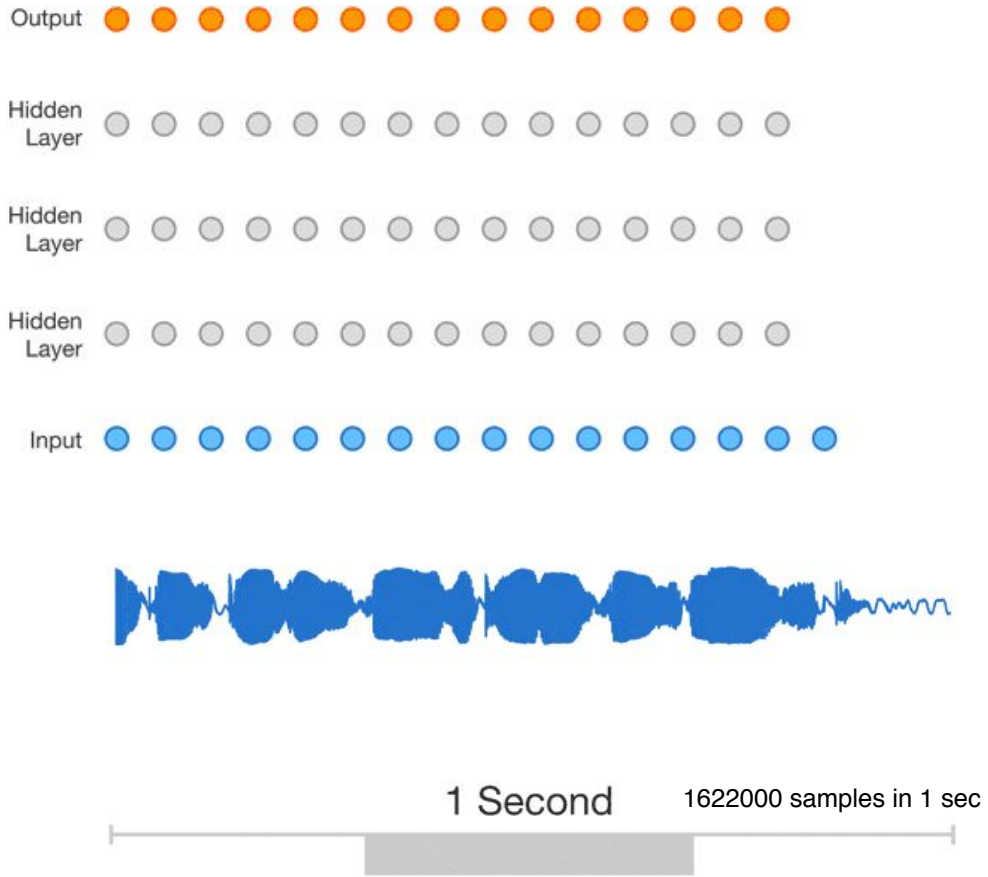
occluded **completions**

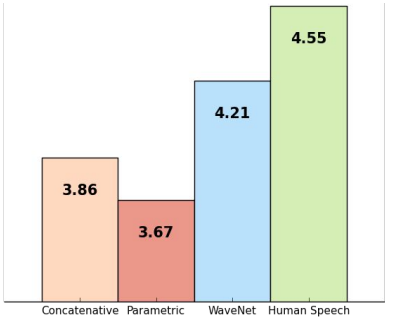
van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

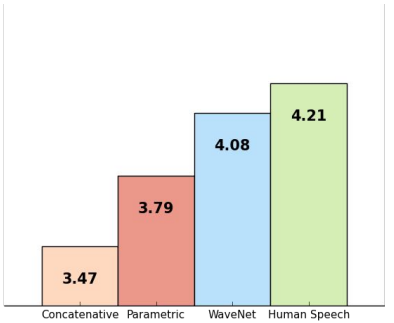
occluded completions original

van den Oord, A., et al. “Pixel Recurrent Neural Networks.” *ICML (2016).*

WaveNets







van den Oord, A., et al. “WaveNet: A Generative Model for Raw Audio.” *arxiv (2016).*

Video Pixel Network (VPN)

data entropy

Model	Test
(Shi et al., 2015)	367.2
(Srivastava et al., 2015a)	341.2
(Brabandere et al., 2016)	285.2
(Patraucean et al., 2015)	179.8
Baseline model	110.1
VPN	**87.6**
Lower Bound	86.3

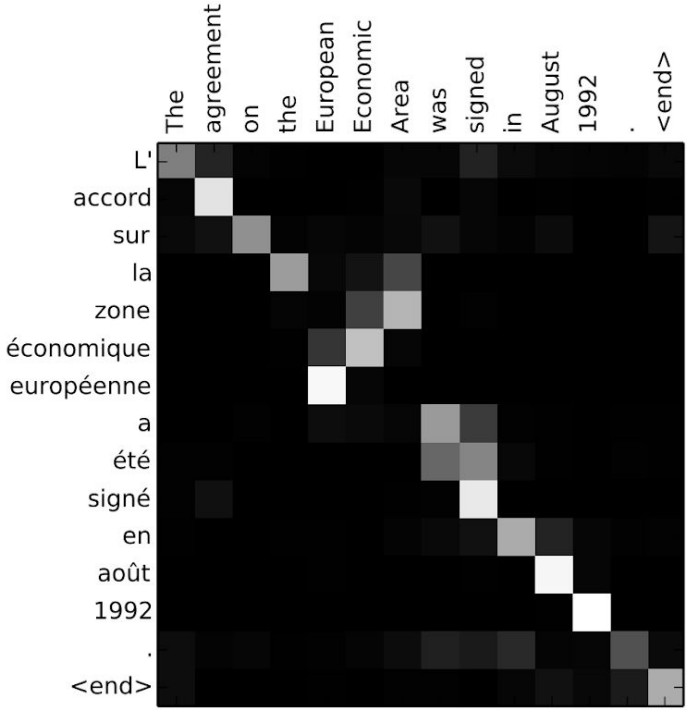
very expensive!

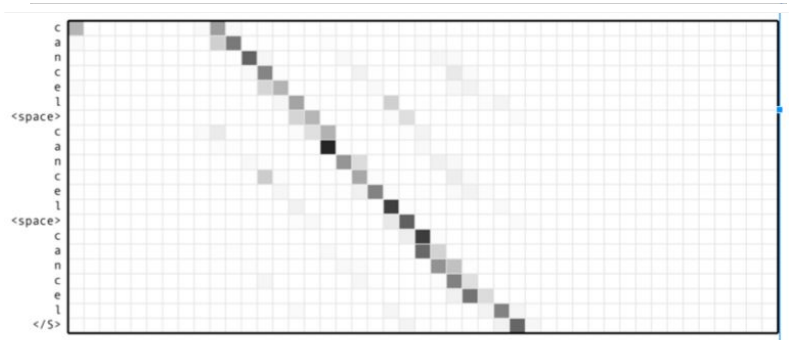
Kalchbrenner, N., et al. “Video Pixel Networks.” ***ICML (2017)***.

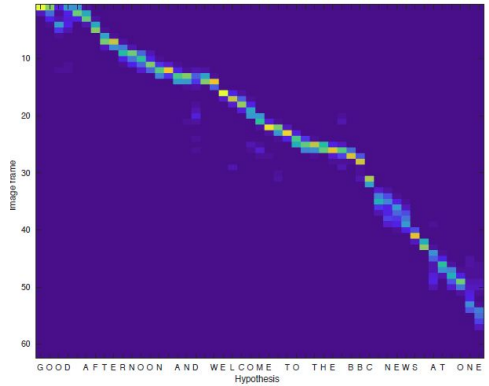
Online Seq2Seq

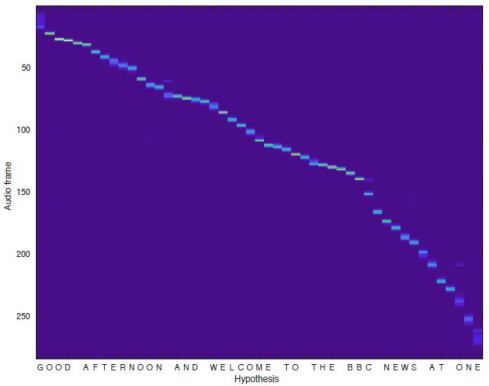
Practical limitations of Online Seq2Seq

- Have to wait for all of the input to arrive
- Yet, attention vector reveals a lot of globally monotonic structure









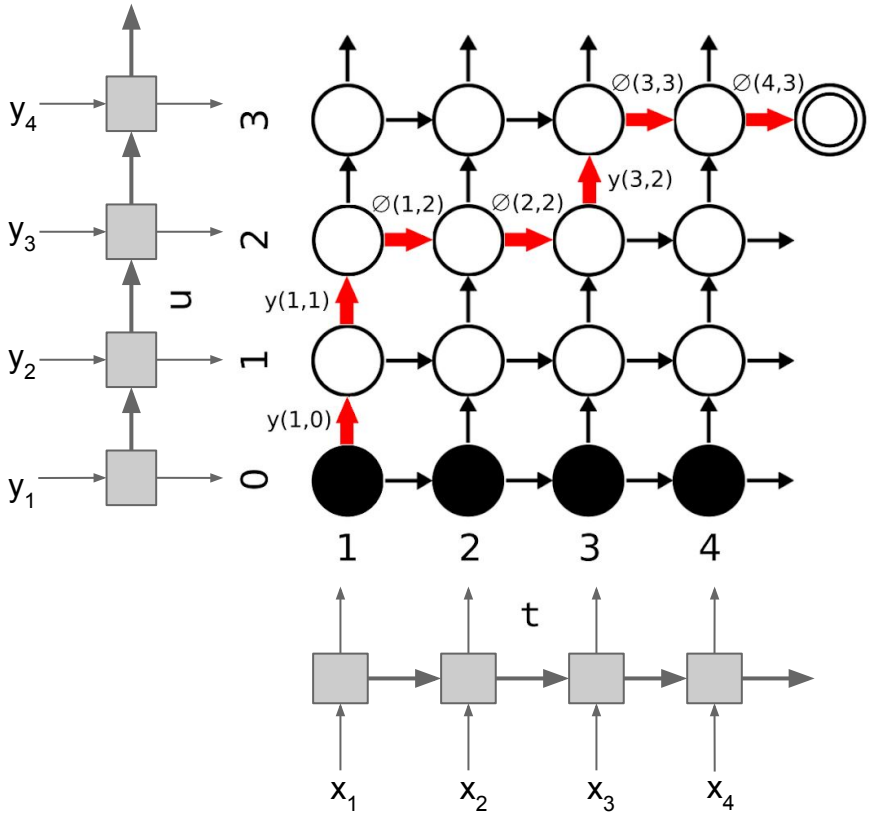
Online Sequence to Sequence Models

- Overcome limitations of sequence to sequence models
  - No need to wait for the entire input sequence to arrive
- Has to solve the following problem:
  - Has enough information arrived for the model to make its next prediction ?

- Yu, L., Buys, J., Blunsom, P. “Online Segment to Segment Neural Transduction.” *EMLP (2016).*
- Jaitly, N., et al. “An Online Sequence-to-Sequence Model Using Partial Conditioning.” *NIPS (2016).*
- Luo, Y., et al. “Learning online alignments with continuous rewards policy gradient.” *ICASSP (2017).*
- Gu, J, Neubig, G, Cho, K, Li, V.O.K.. “Learning to translate in real-time with neural machine translation”. *EACL (2017)*
- Raffel, Colin, et al. "Online and Linear-Time Attention by Enforcing Monotonic Alignments." *ICML 2017 (*Tue Aug 8th 01:30*)*

Sequence Transducer

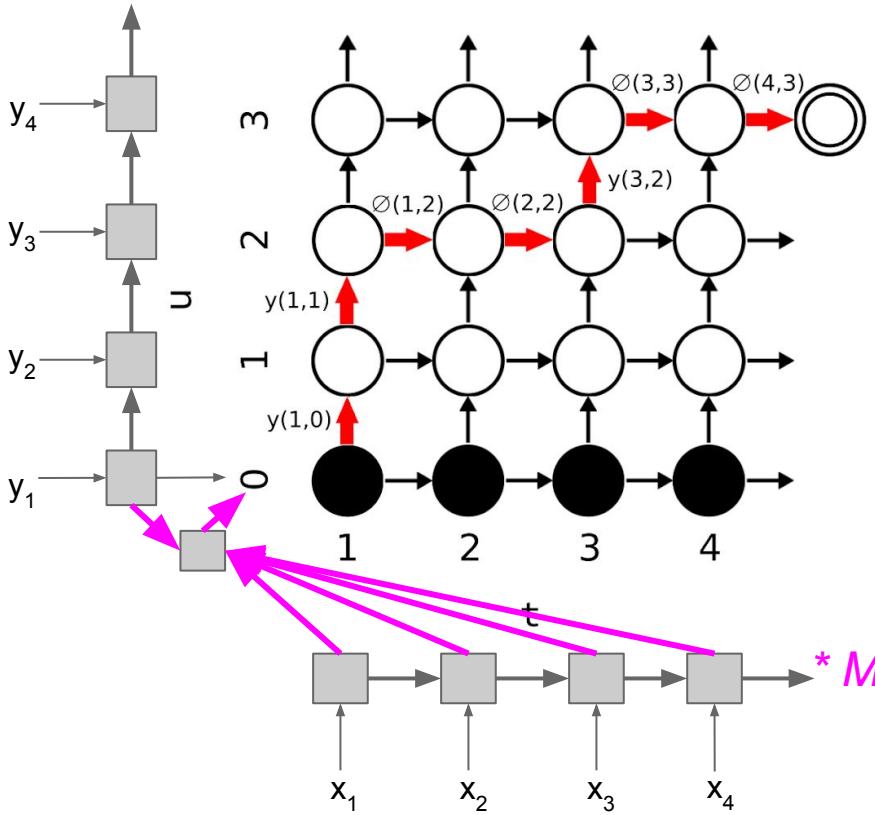
idea: separate states of encoder and decoder



- RNN over input (encoder)
- RNN over output (transducer)
- Right arrow consumes input without producing output
- Up arrow produces output without consuming input
- Edge probabilities computed from softmax that is a sum of logits from the encoder and the decoder

Graves, A. “Sequence transduction with recurrent neural networks.” *ICML (2012).*

Sequence Transducer vs Sequence-to-Sequence

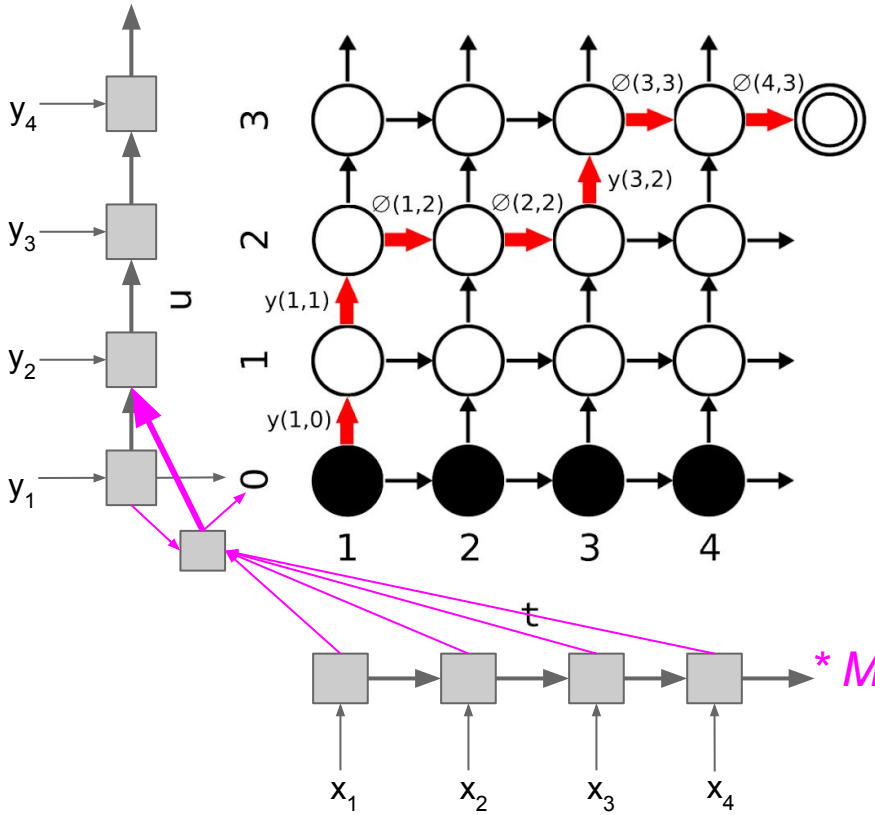


- Transducer step interacts with only one input step

** Missing connections in sequence transducer*

Graves, A. “Sequence transduction with recurrent neural networks.” *ICML (2012).*

Sequence Transducer vs Sequence-to-Sequence

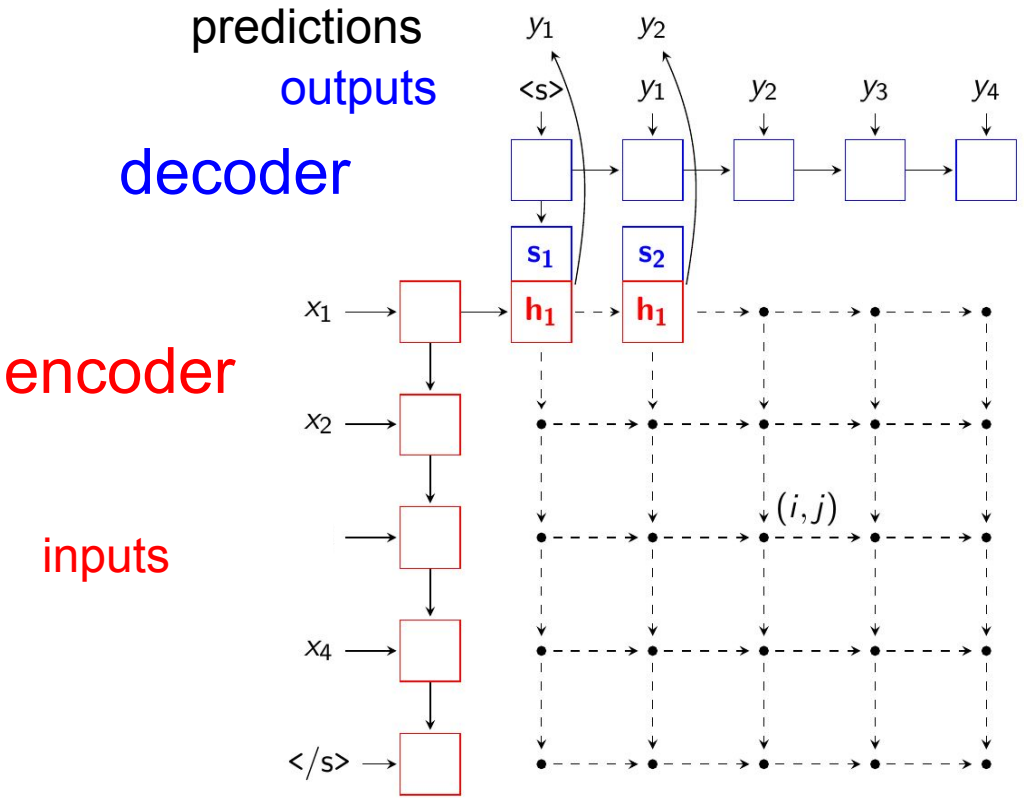


- Transducer output interacts with only one input step
- Transducer states do not depend on path history because context vector connection is missing

** Missing connections in sequence transducer*

Graves, A. “Sequence transduction with recurrent neural networks.” *ICML (2012).*

Online Segment to Segment Neural Transduction



Yu, L., Buys, J., Blunsom, P. “Online Segment to Segment Neural Transduction.” *EMLP (2016).*

Online Segment to Segment Neural Transduction

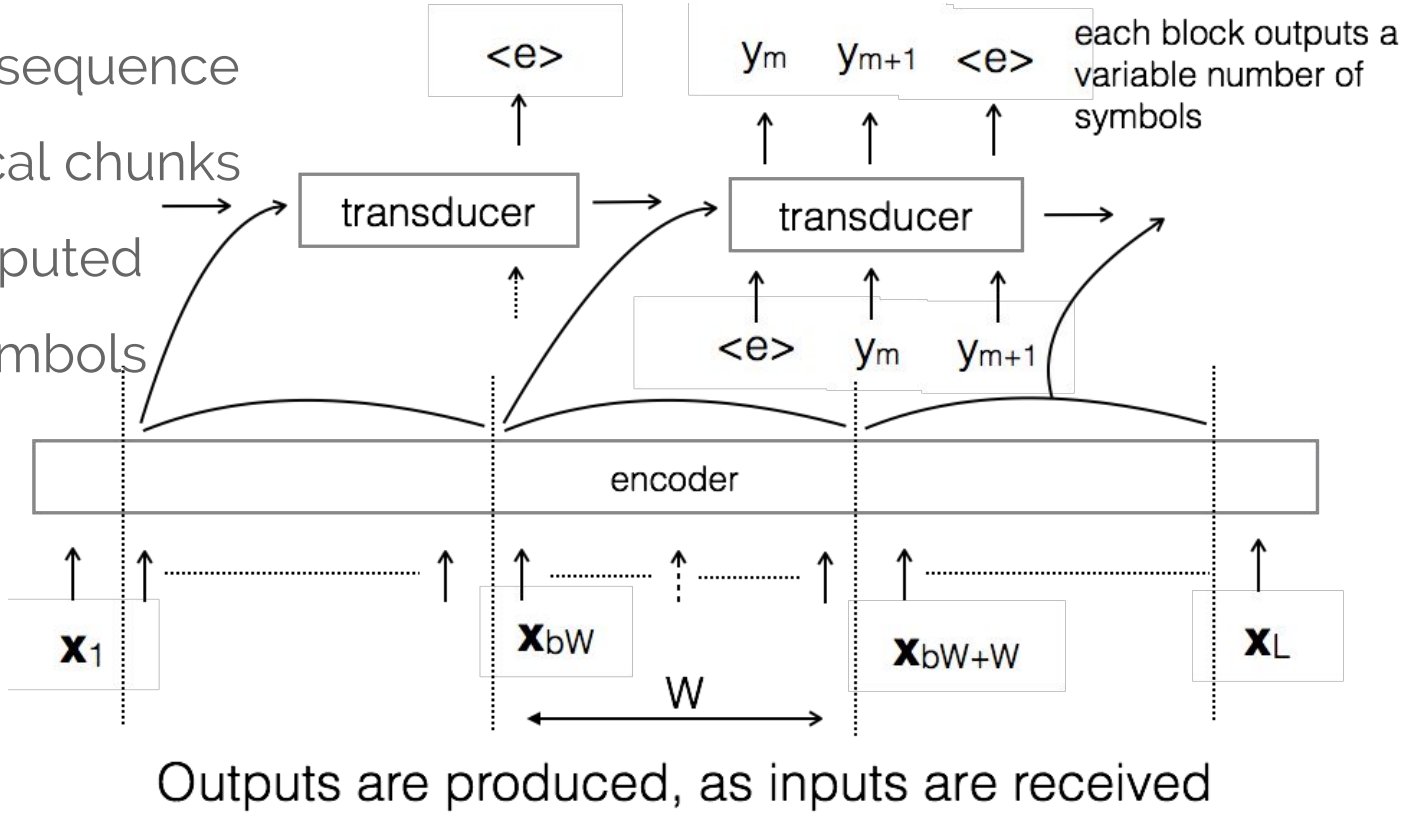
Model	ROUGE-1	ROUGE-2	ROUGE-L
Seq2Seq	25.2	9.1	23.1
Seq2Seq + attention	30.3	14.2	28.5
Neural Transduction	31.2	14.3	28.9
Neural Transduction + bidirectional encoder	31.7	14.7	29.3

F1 values on Abstractive sentence summarisation task

1. Yu, L., Buys, J., Blunsom, P. “Online Segment to Segment Neural Transduction.” *EMLP (2016).*
2. Yu, L., et al. “The Neural Noisy Channel.” *ICLR (2017).*

A Neural Transducer

- sequence-to-sequence models on local chunks
- Features computed depend on symbols output so far
- Introduces an alignment problem

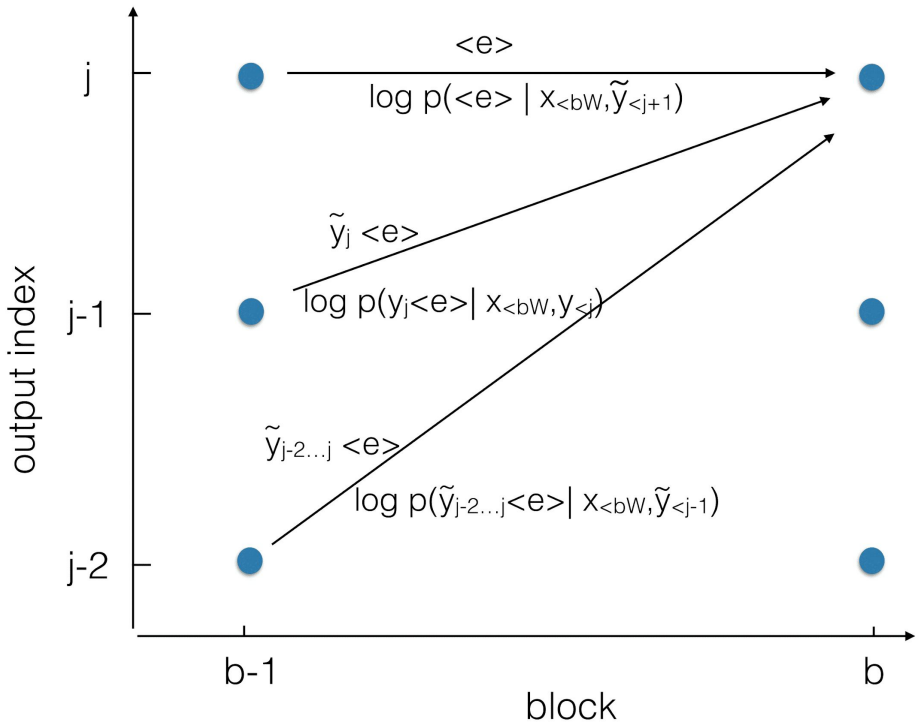


where should we produce the output?/how to align segment training signals?

Jaitly, N., et al. “An Online Sequence-to-Sequence Model Using Partial Conditioning.” *NIPS (2016).*

A Neural Transducer - Viterbi-like training

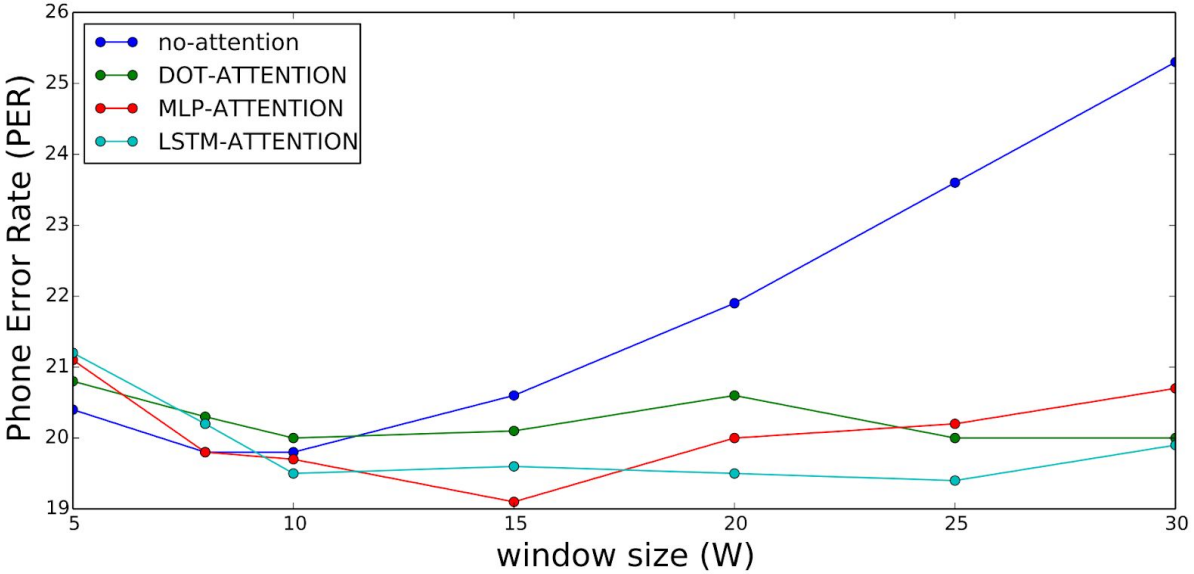
- Naive beam search for Viterbi training fails easily because of corner cases
- Approximate Dynamic programming for finding best alignment works quite well



Jaitly, N., et al. “An Online Sequence-to-Sequence Model Using Partial Conditioning.” *NIPS (2016).*

A Neural Transducer - Results

# of layers in encoder / transducer	1	2	3	4
2		19.2	18.9	18.8
3		18.5	**18.2**	19.4

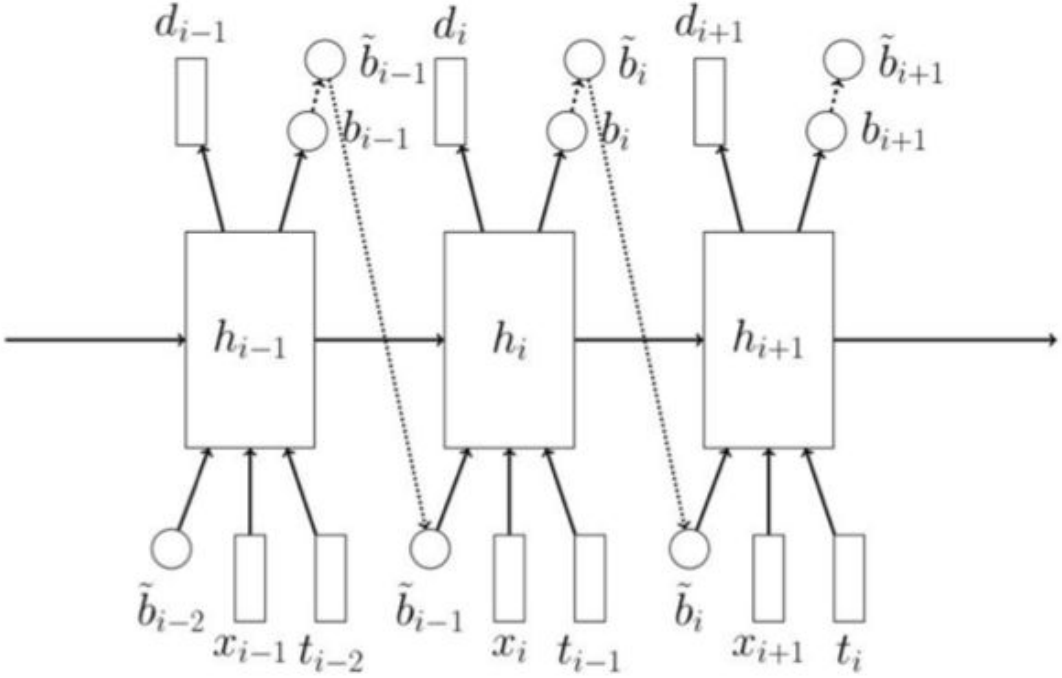


Jaitly, N., et al. “An Online Sequence-to-Sequence Model Using Partial Conditioning.” *NIPS (2016).*

A Stochastic Emissions Model

idea: reinforcement learning to decide whether to produce output

- Autoregressive at every step

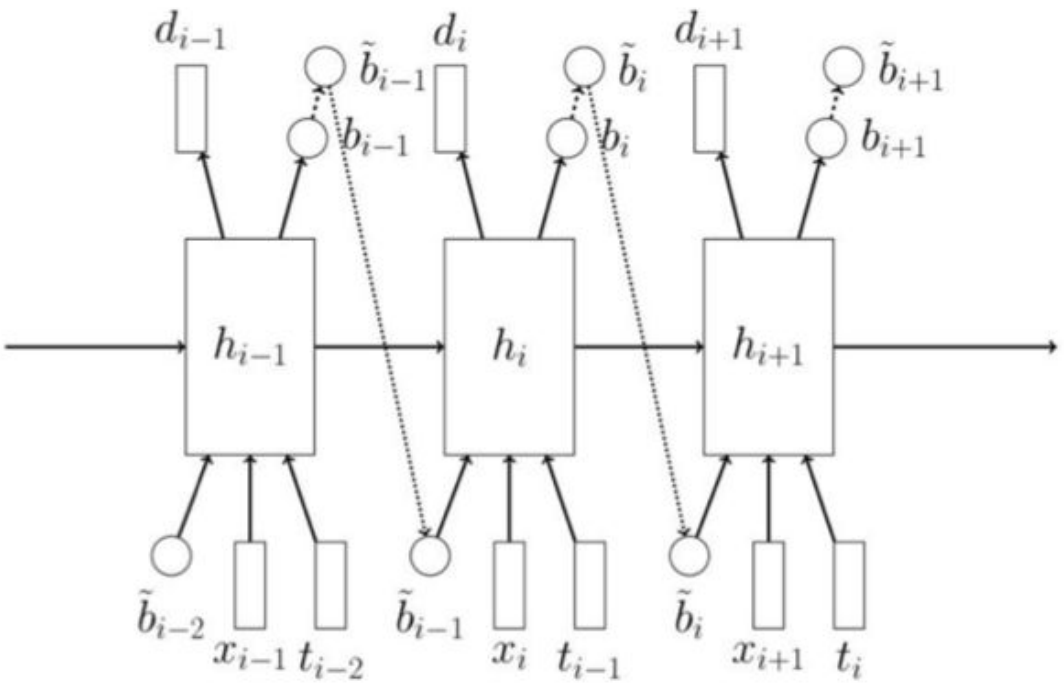


d	distributions over phoneme
b	probabilities to emit
$\tilde{b}$	samplings of *Bernoulli(b)*
h	states of RN
x	input
t	last true label

Luo, Y., et al. “Learning online alignments with continuous rewards policy gradient.” *ICASSP (2017).*

A Stochastic Emissions Model

- Continuous Reward $R_i = \tilde{b}_i \log p(d_i = t_i | \mathbf{x}_{\leq i}, \tilde{\mathbf{b}}_{<i}, \mathbf{t}_{<i})$

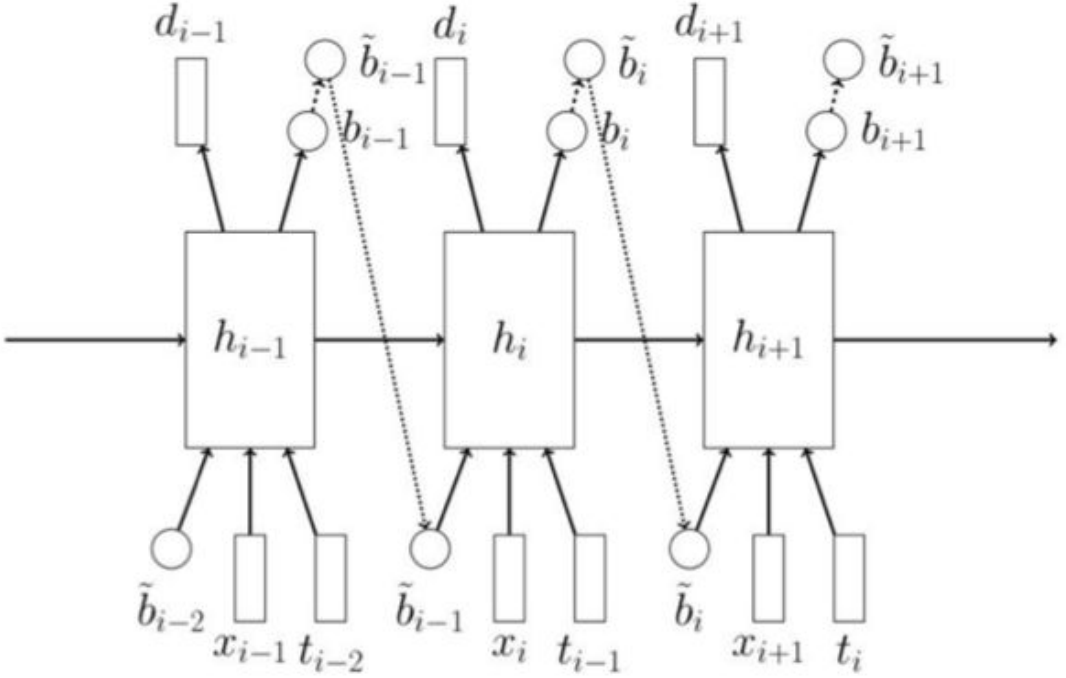


d	distributions over phoneme
b	probabilities to emit
$\tilde{b}$	samplings of *Bernoulli(b)*
h	states of RN
x	input
t	last true label

Luo, Y., et al. “Learning online alignments with continuous rewards policy gradient.” *ICASSP (2017).*

A Stochastic Emissions Model

- Optimize for expected reward: $\mathcal{R} = \mathbb{E}_{\tilde{\mathbf{b}}}\left[R(\tilde{\mathbf{b}})\right]$

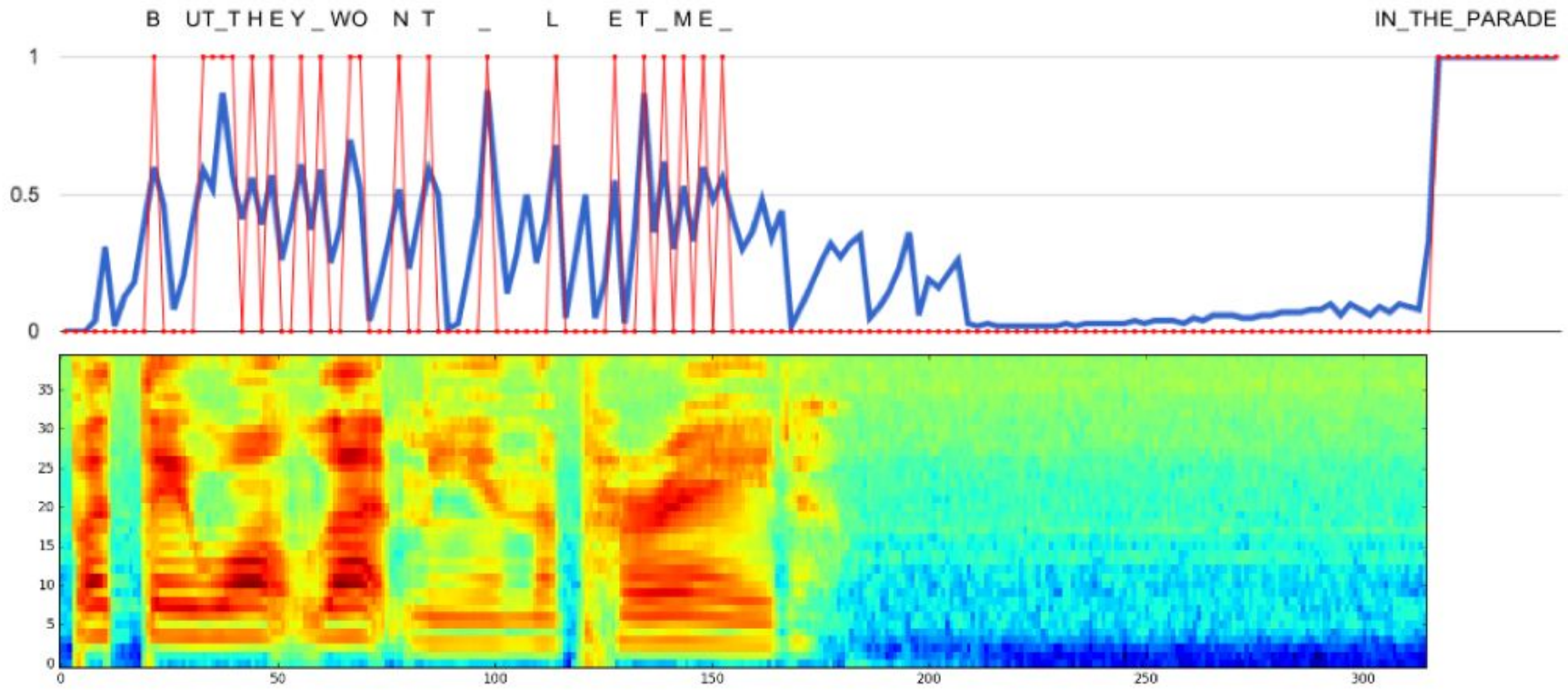


d	distributions over phoneme
b	probabilities to emit
$\tilde{b}$	samplings of *Bernoulli(b)*
h	states of RN
x	input
t	last true label

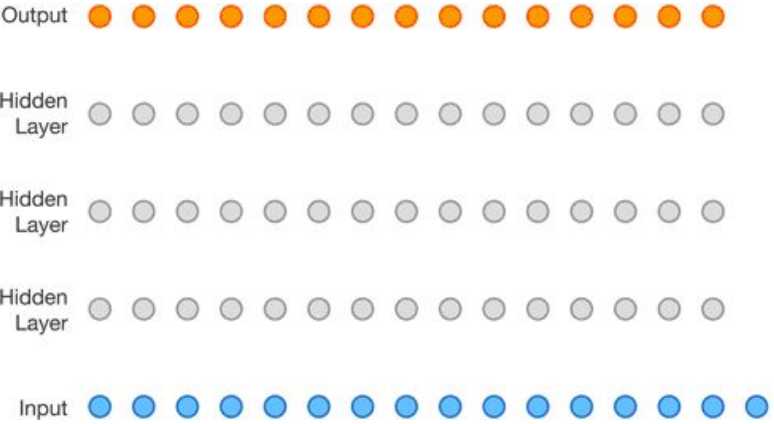
Luo, Y., et al. “Learning online alignments with continuous rewards policy gradient.” *ICASSP (2017).*

A Stochastic Emissions Model

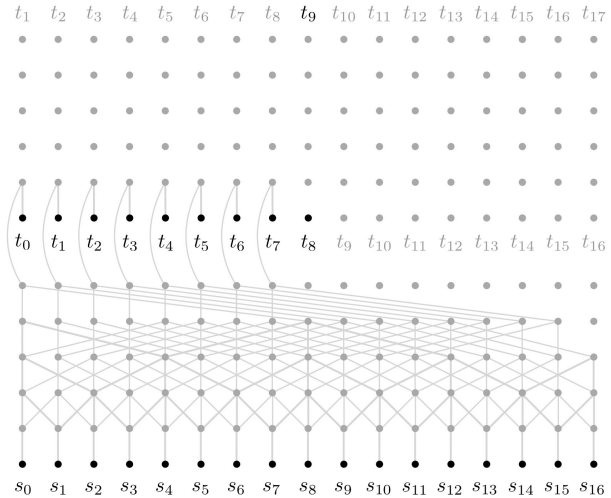
speech recognition
blue: emission probability
red: whether/not admitting a symbol



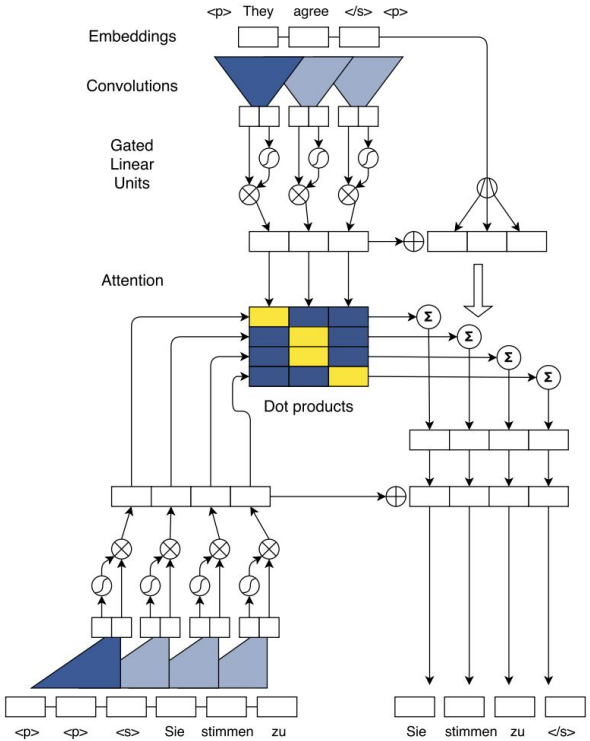
New Architectures



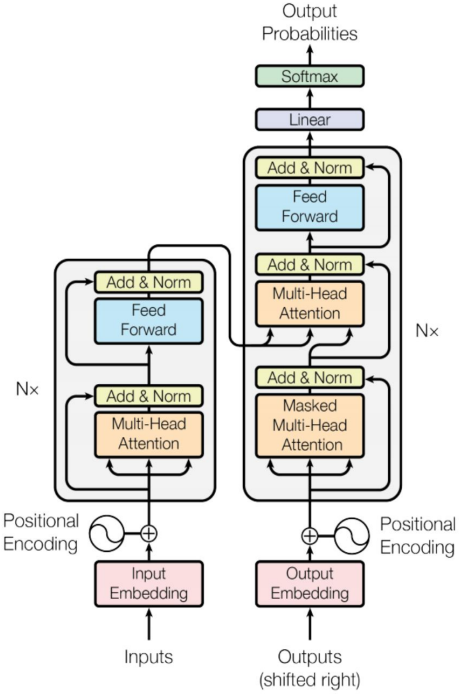
Wavenet, van den Oord, et al, 2016



Bytenet, Kalchbrenner, et al, 2016



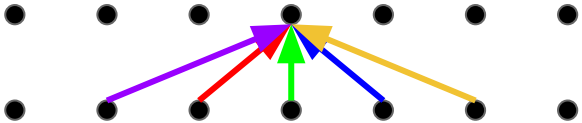
Conv seq2seq, Gehring, et al, 2017



Att is all you need, Vaswani, et al, 2017

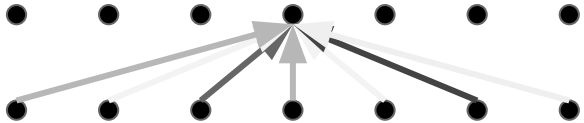
Self-Attention

Convolution



fixed sliding window with fixed weights

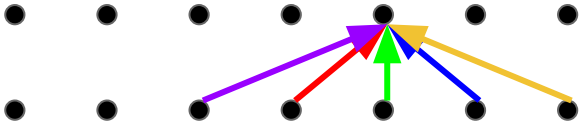
Self-Attention



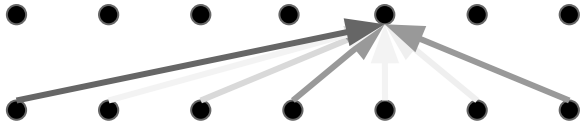
changing weights computed everywhere

Self-Attention

Convolution



Self-Attention

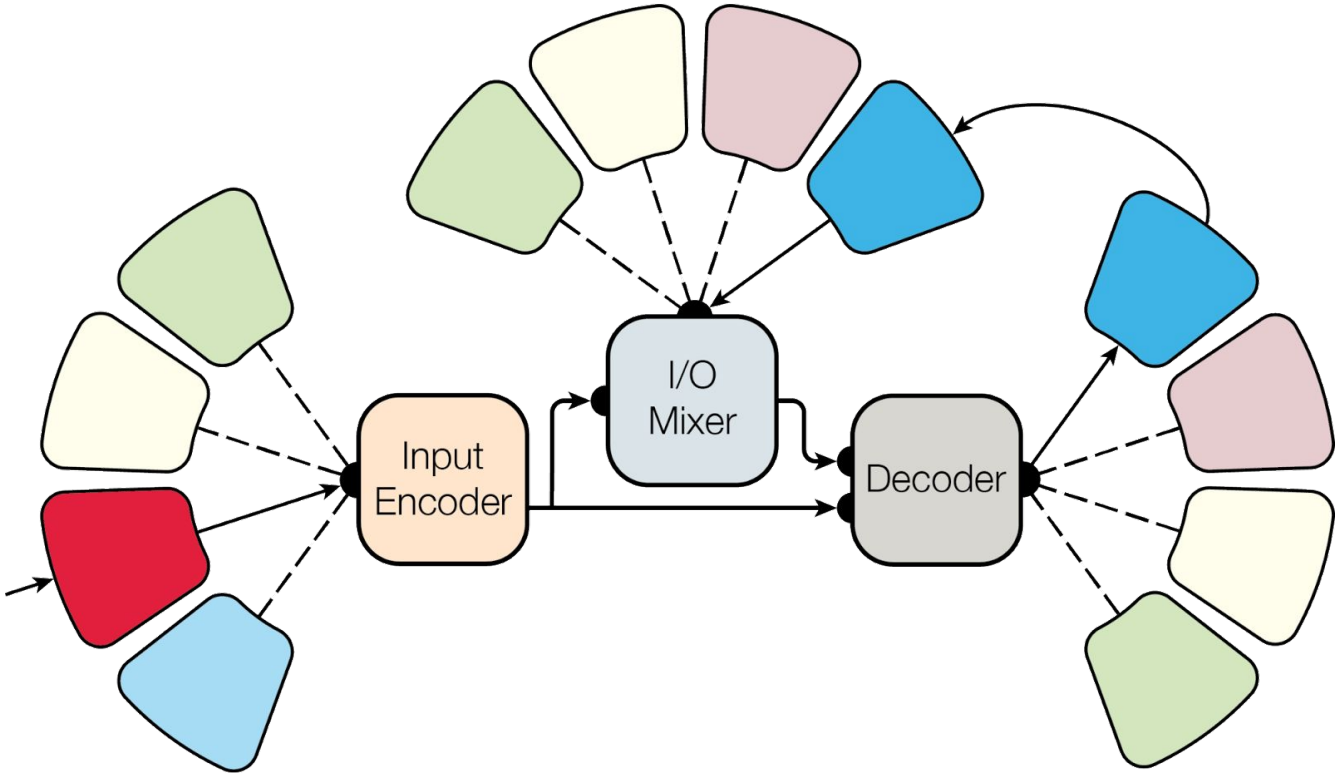


Machine Translation Results: WMT-14

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [17]	23.75			
Deep-Att + PosUnk [37]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [36]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [31]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [37]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [36]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

MultiModel

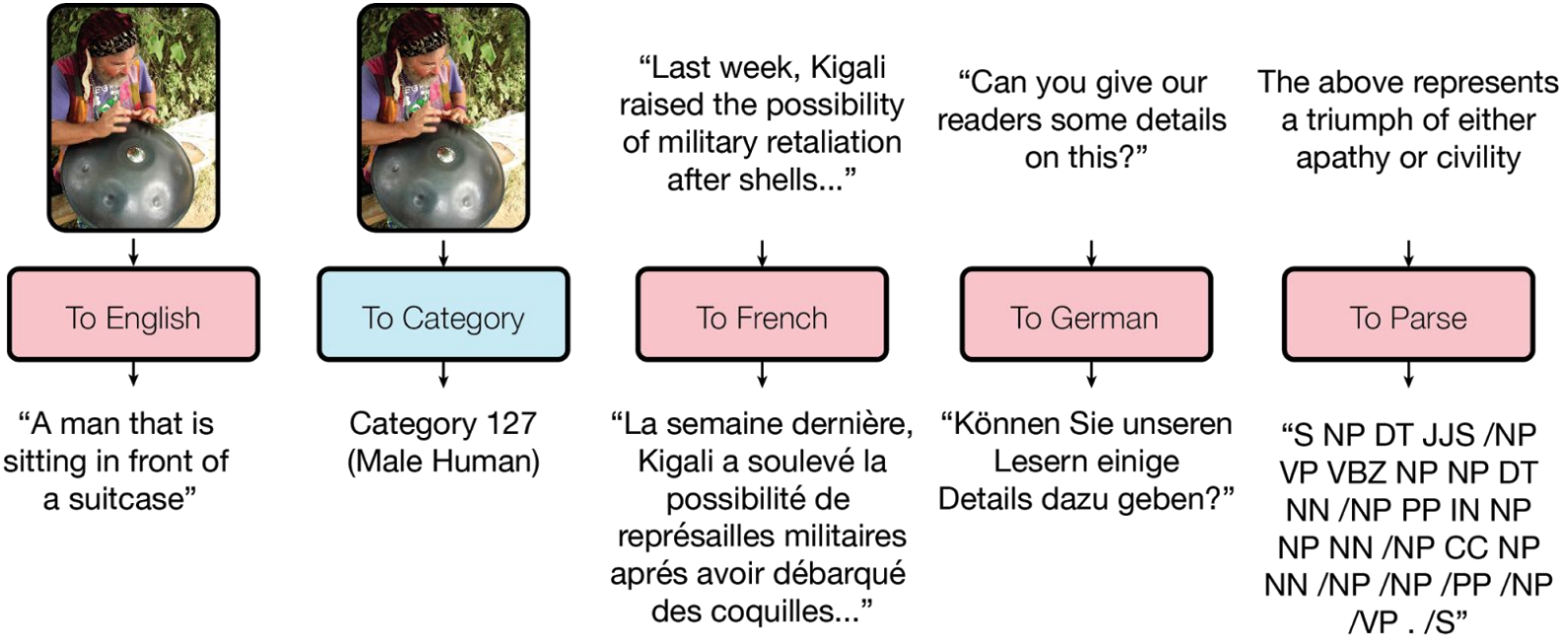
many sequences to many sequences



Kaiser, L., et al. “One Model To Learn Them All.” *arxiv (2017).*

MultiModel

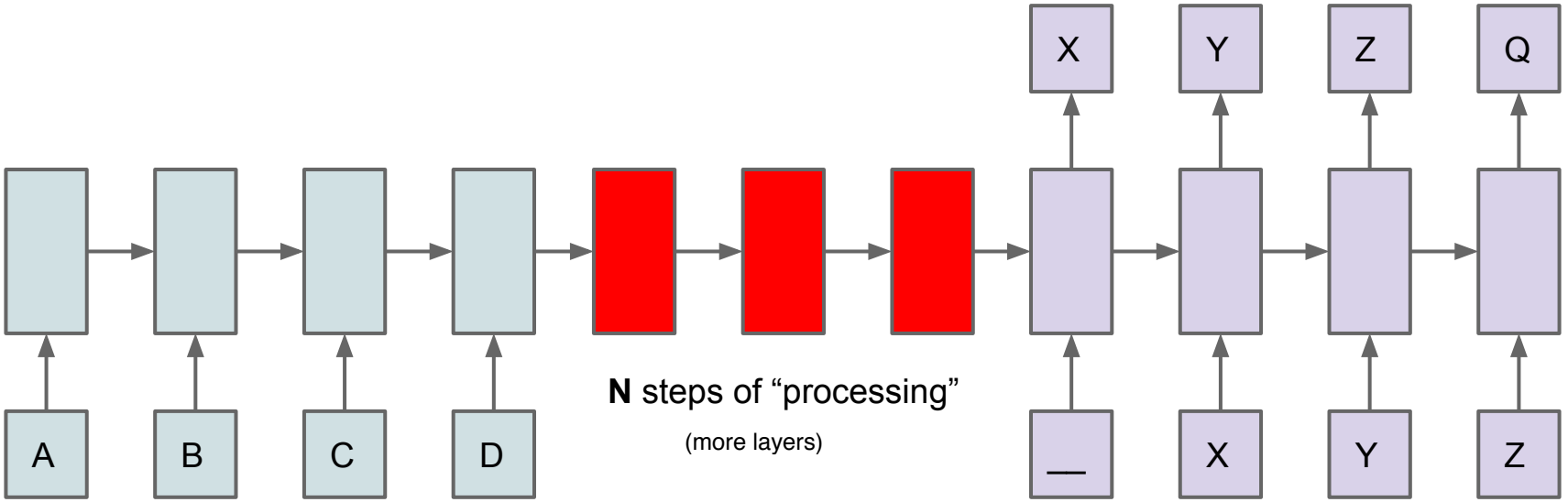
transfer learning from a parsing task



Kaiser, L., et al. “One Model To Learn Them All.” *arxiv (2017).*

Processing Depth

model decides when to compute and when to compute more



1. Vinyals, O., et al. “Order Matters: Sequence to Sequence for sets.” *ICLR (2015).*
2. Sukhbaatar, S., et al. “End-to-End Memory Networks.” *NIPS (2015).*

How to pick N? Learn it of course!

Slide Credit:
Alex Graves

Add a *halting unit h* to the output

$$h_t^n = \sigma\left(W_h s_t^n + b_h\right)$$

Use this to define the *halt probability* p_t^n at ponder step *n*

$$p_t^n = \begin{cases} R(t) \text{ if } n = N(t) \\ h_t^n \text{ otherwise} \end{cases}$$

Where *N(t)* is # updates at *t*

$$N(t) = \min\{n' : \sum_{n=1}^{n'} h_t^n >= 1 - \epsilon\}$$

And *R(t)* is the *remainder* at *t*

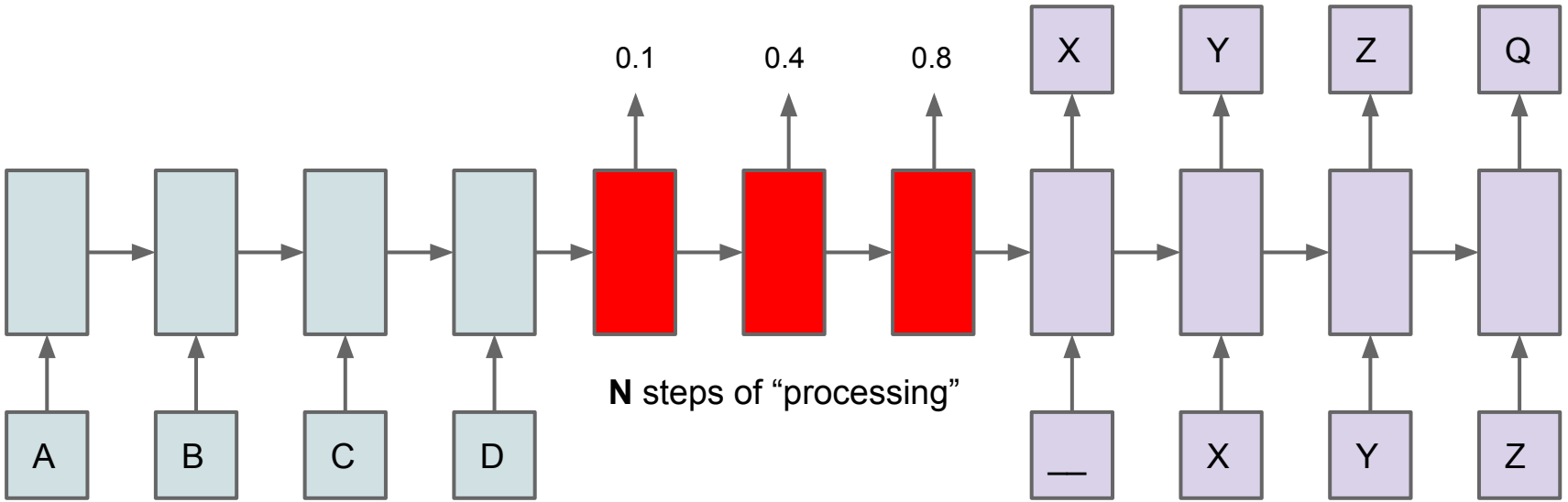
$$R(t) = 1 - \sum_{n=1}^{N(t)-1} h_t^n$$

The final states and outputs at *t* are weighted sums (!)

$$s_t = \sum_{n=1}^{N(t)} p_t^n s_t^n \qquad y_t = \sum_{n=1}^{N(t)} p_t^n y_t^n$$

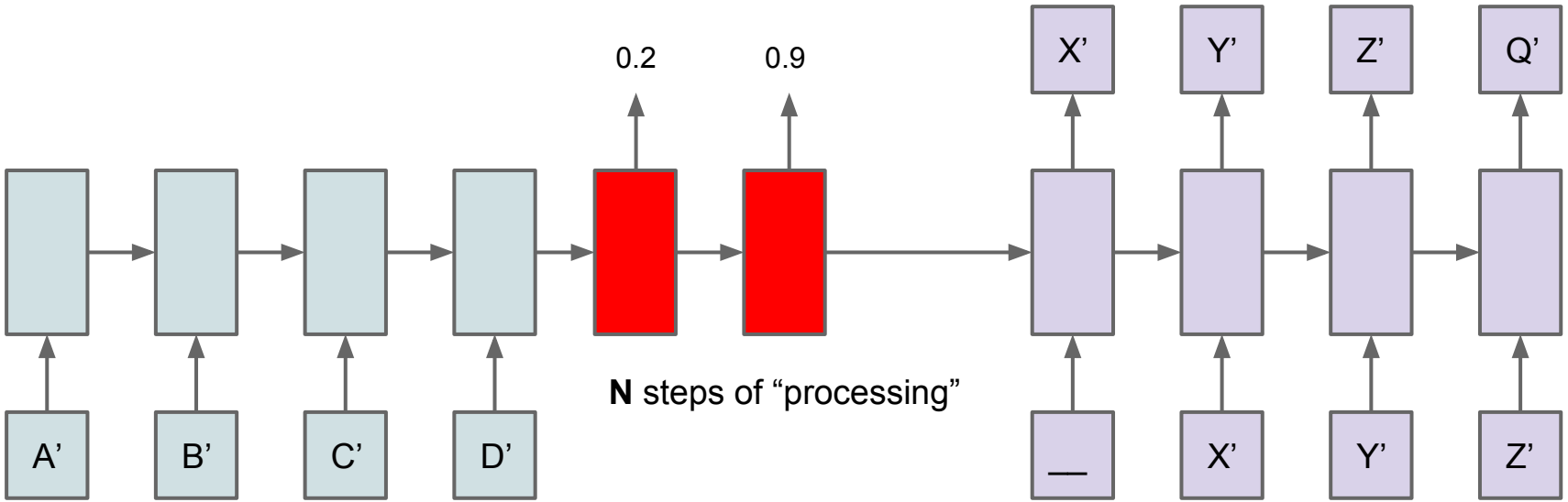
Graves, A. “Adaptive Computation Time for Recurrent Neural Networks.” *arxiv (2016).*

Adaptive Computation Time (ACT)



Graves, A. “Adaptive Computation Time for Recurrent Neural Networks.” *arxiv (2016).*

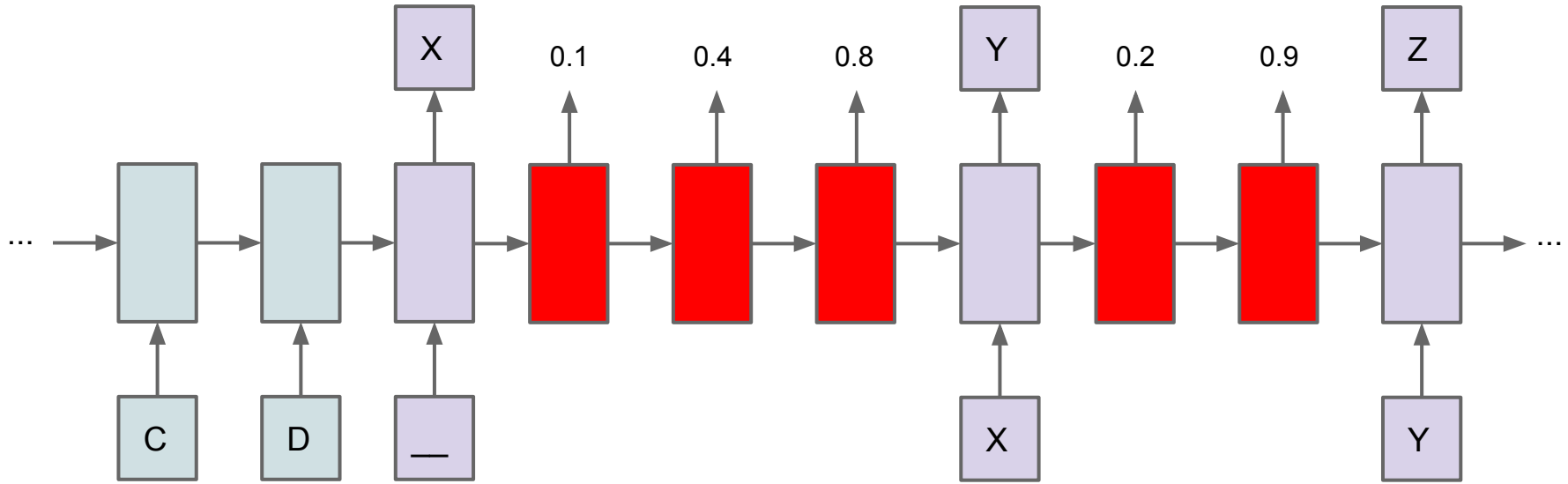
Adaptive Computation Time (ACT) [Graves, 2016]



Graves, A. “Adaptive Computation Time for Recurrent Neural Networks.” *arxiv (2016).*

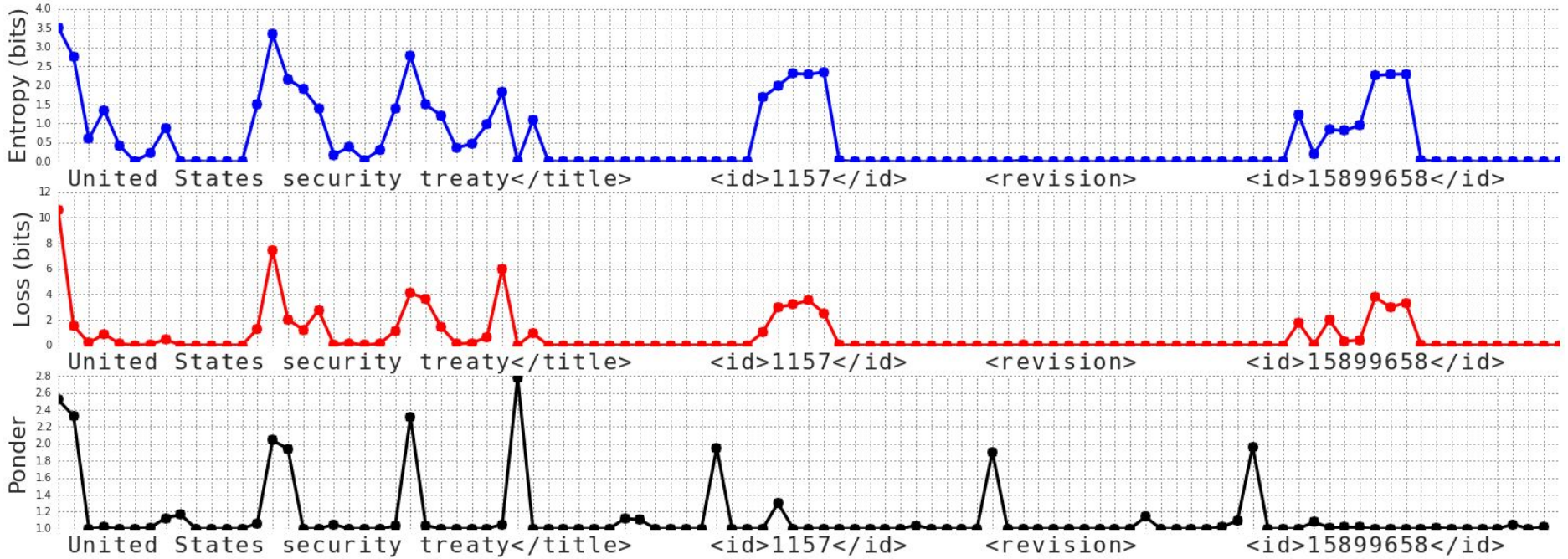
(in)ACT [Graves, 2016]

same architecture within decoder



Graves, A. “Adaptive Computation Time for Recurrent Neural Networks.” *arxiv (2016).*

Character level LM



Graves, A. “Adaptive Computation Time for Recurrent Neural Networks.” *arxiv (2016).*