CS7150 Deep Learning

Jiaji Huang

https://jiaji-huang.github.io

02/10/2024

Recap of Last Lecture

- Casual Language Model
  - N-gram
  - RNN, LSTM

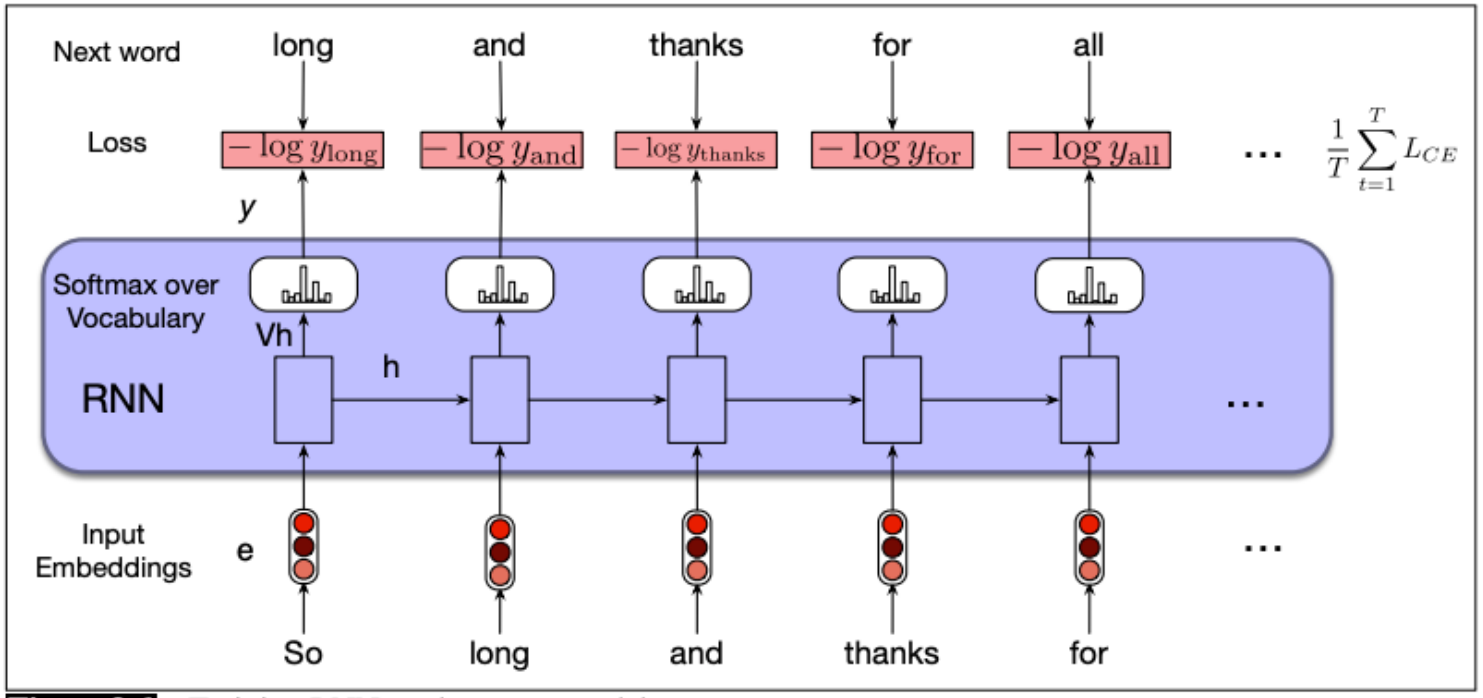


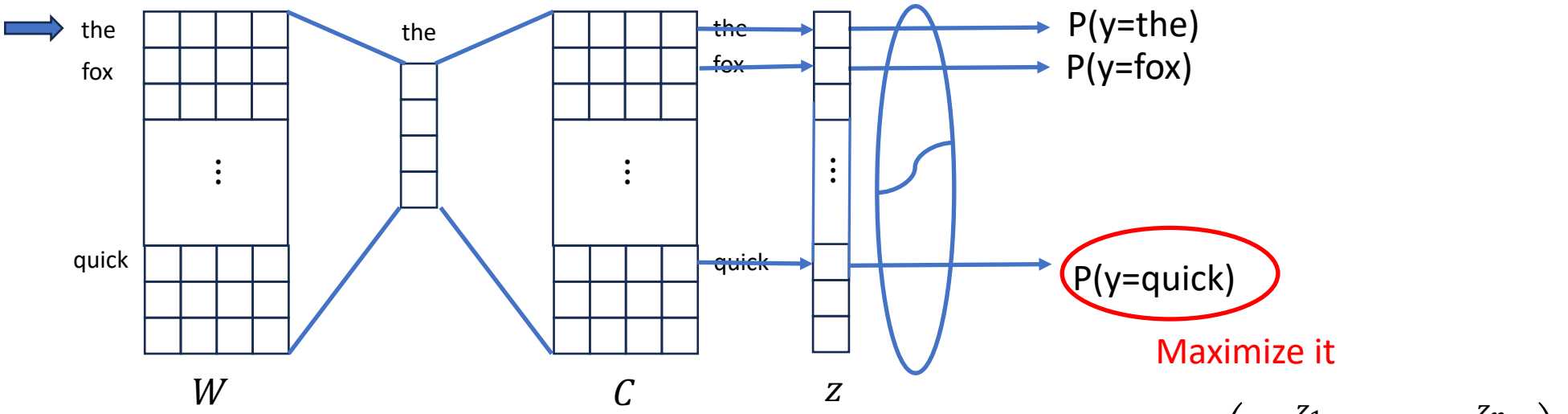
Figure 9.6 Training RNNs as language models.

Illustration from https://web.stanford.edu/~jurafsky/slp3/9.pdf

Recap of Last Lecture

- Word Representations: Word2vec

  e.g, predict context word “quick” from input word “the”



Recap of Last Lecture

- Transformer encoder and decoder

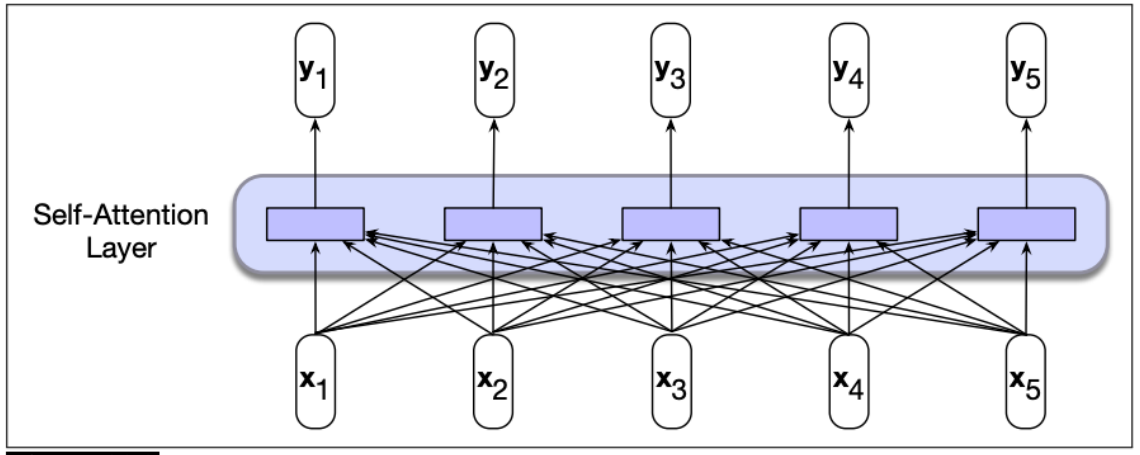


Figure 11.2 Information flow in a bidirectional self-attention model. In processing each element of the sequence, the model attends to all inputs, both before and after the current one.

Encoder

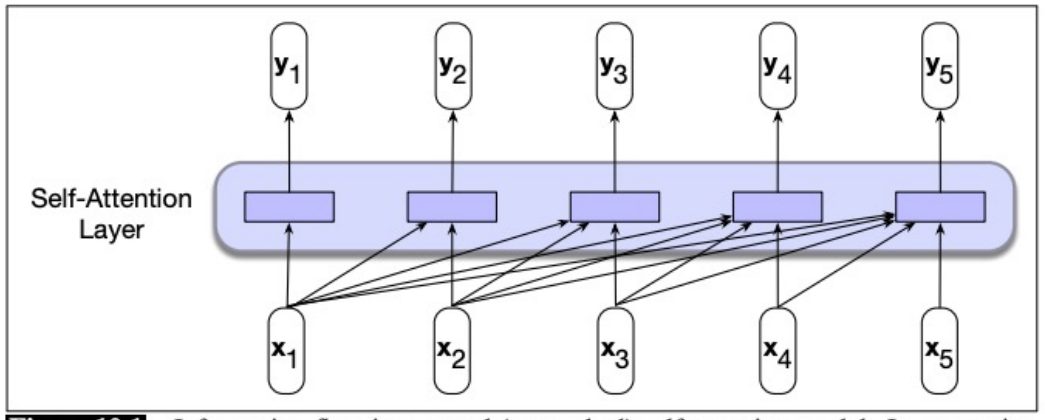


Figure 10.1 Information flow in a causal (or masked) self-attention model. In processing each element of the sequence, the model attends to all the inputs up to, and including, the current one. Unlike RNNs, the computations at each time step are independent of all the other steps and therefore can be performed in parallel.

Decoder

Illustration from https://web.stanford.edu/~jurafsky/slp3/11.pdf

Encoder vs Decoder

- Encoder: Understanding, often bidirectional
- Decoder: Generation, often causal
- Encoder-Decoder: Does both

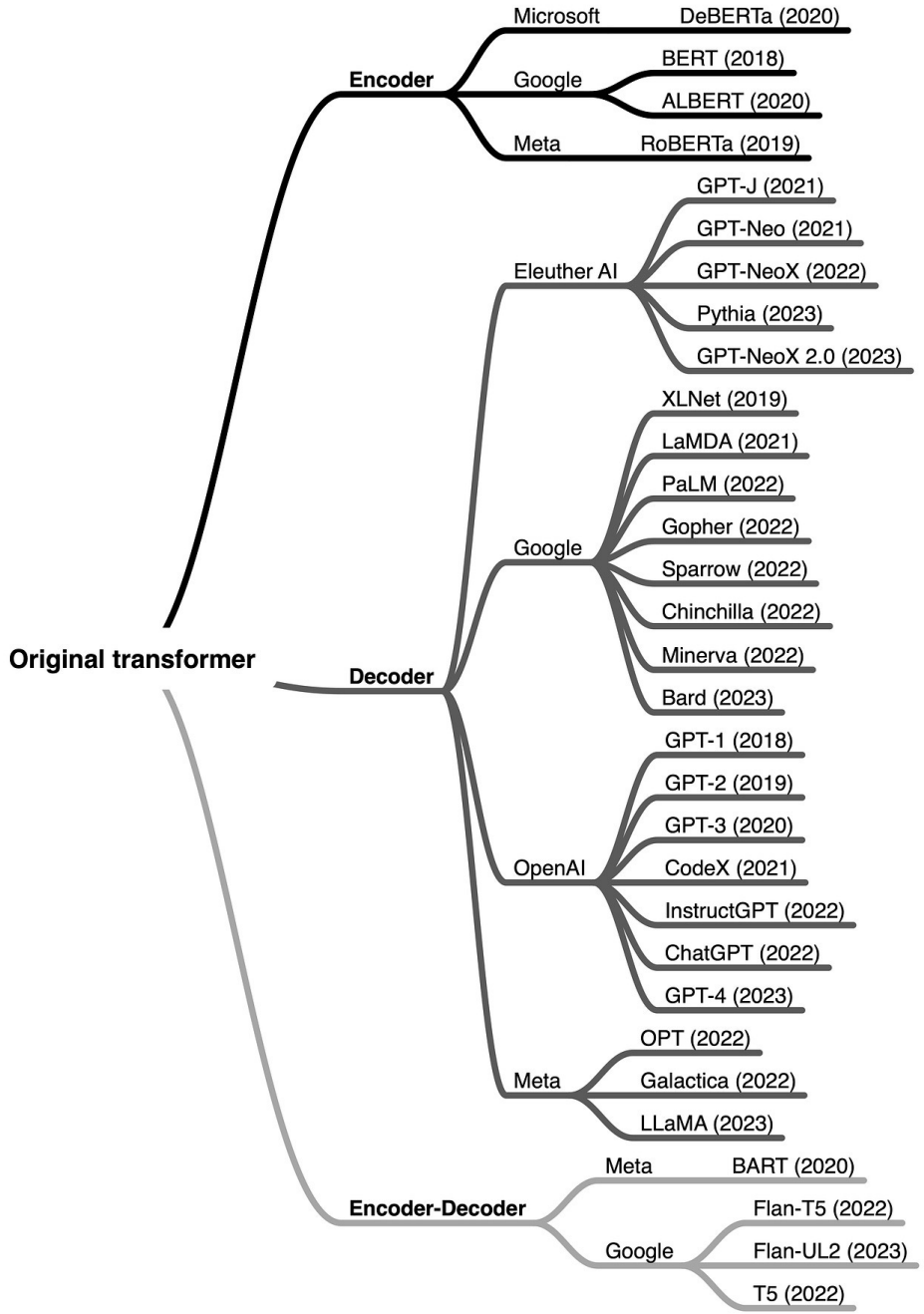


Illustration from this [blogpost]

Agenda

- Encoder Model
  - ELMO
  - Masked LM and BERT
- Decoder Model
  - GPT
- Enc-Dec Model
  - Translation
  - Speech Recognition

ELMO  : LSTM as Encoder

Deep contextualized word representations

Matthew E. Peters[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washington

NAACL 2018 Best Paper

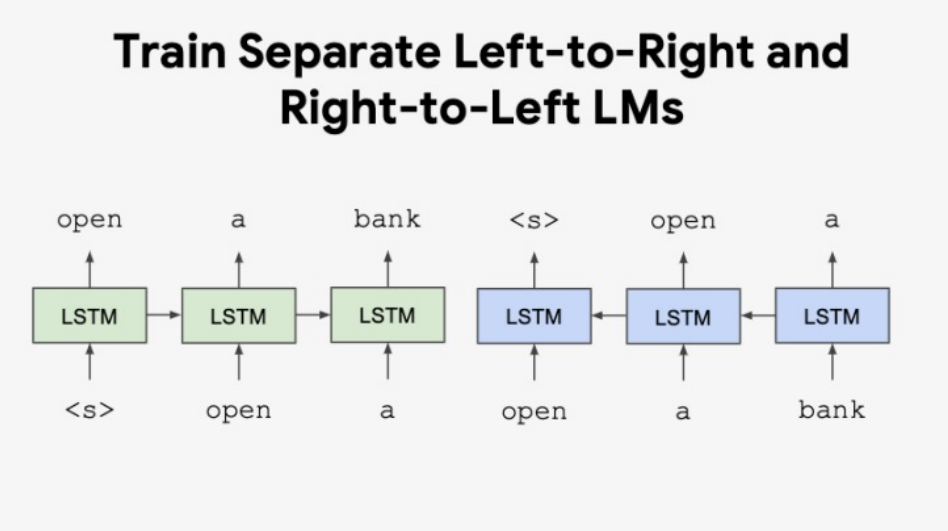
ELMO: Training Phase

- Both left and right contexts are important for understanding word's meaning

We went to the river bank

v.s.

I need to go to bank to make a deposit

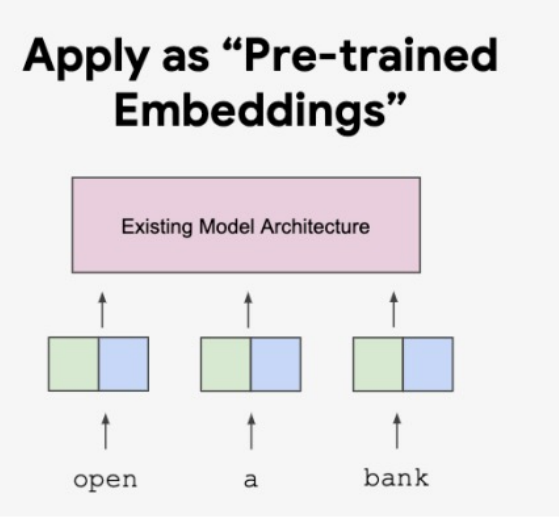


$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big) .$$

Cartoon and example from cos597G lecture slides

ELMO: Applying Phase

- Treat the two learned LSTM models as two functions
- Apply the two functions to get two embeddings, and concatenate
- Learn your task specific classifier on top
- Similar to transfer learning we saw in computer vision!
- More generally, linearly combine embeddings at each layer



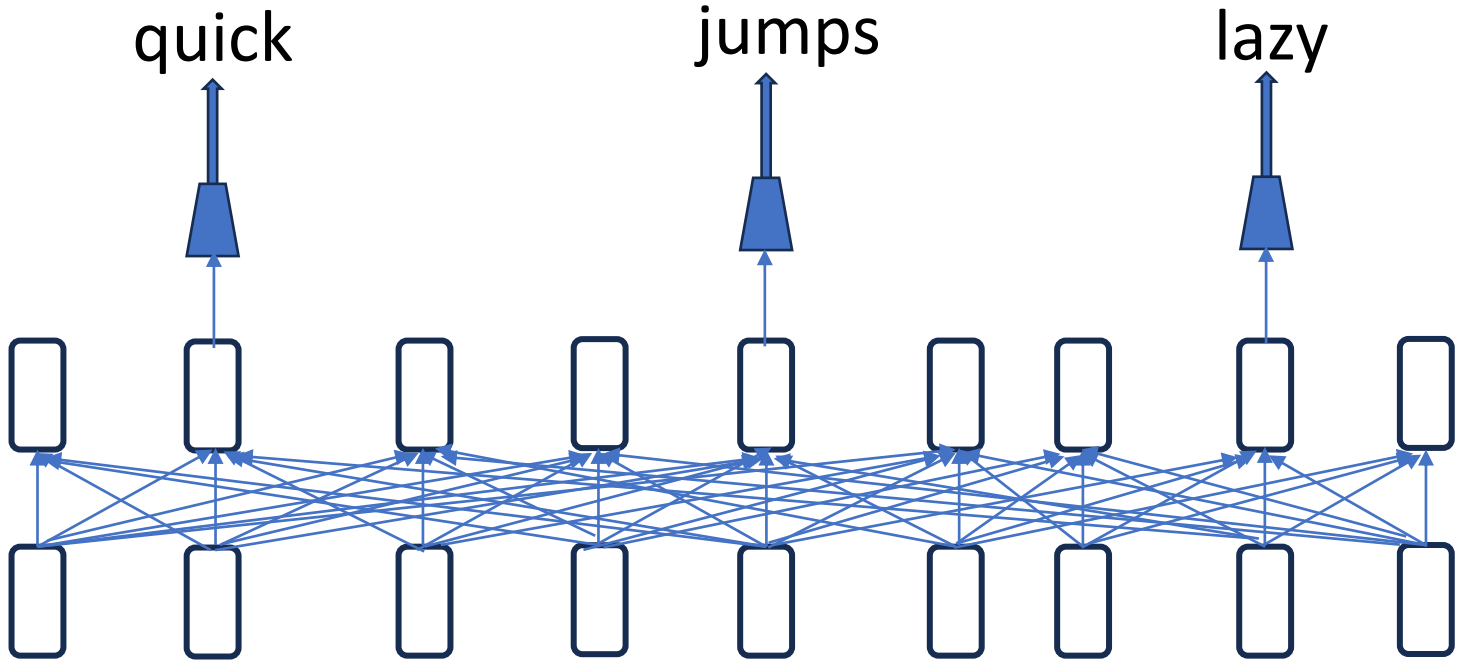
Cartoon from cos597G lecture slides

Discussion

- Advantage over word2vec?
- How to compare ELMO vs. word2vec?
- Replace the architecture with transformer?

Masked LM

- randomly mask $k\%$ words, and predict



BERT

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Released in 2018/10, NAACL 2019 best paper

Masking Strategies

- Typically use $k = 15$, uniformly sample them
- Other options?
- 80-10-10 corruption
  - 80% of the time, put a [mask]
  - 10% of the time, put a random word
  - 10% of the time, put the original word
- Why?

Next Sentence Prediction

- Motivation: learn the relationship between two sentences
- Useful for tasks like: Question Answering, natural language inference

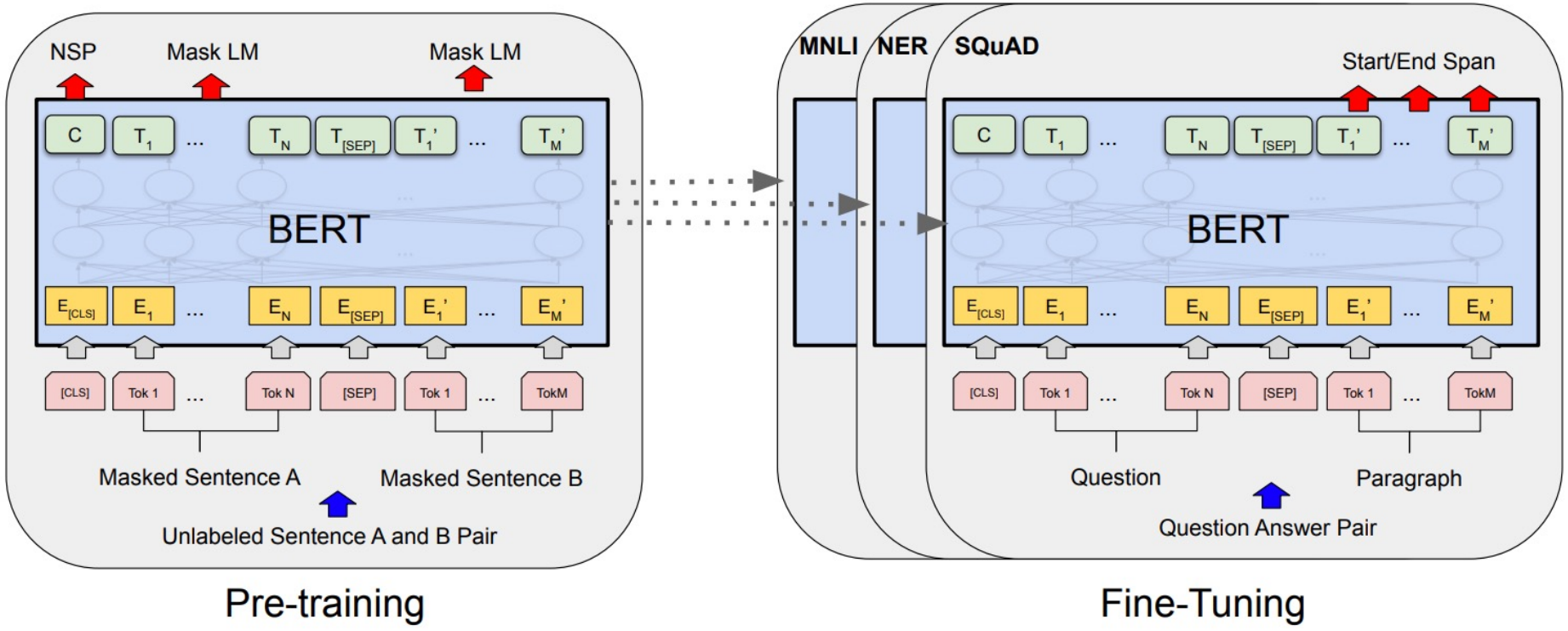
Input: [cls] my dog is cute [sep] he likes playing
Output: IsNext

Input: [cls] my dog is cute [sep] the man went to the store
Output: NotNext

- Sample next sentence 50% of time, 50% as a random one
- Use embedding for [cls] to build a binary classifier

Put together

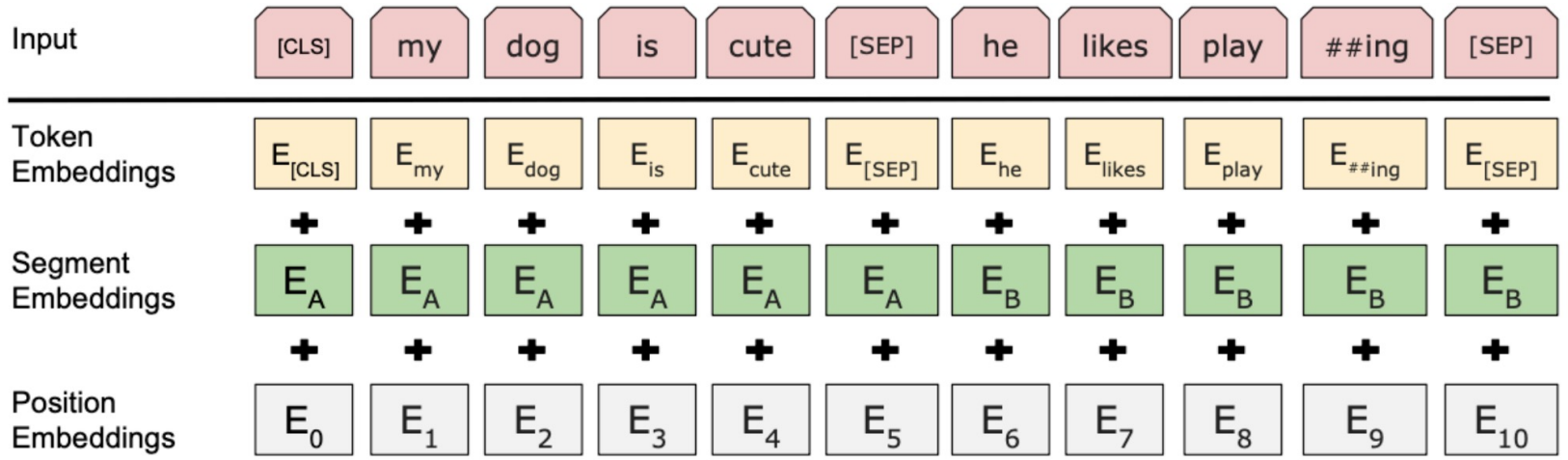
- Pretrain: Masked LM + NSP
- Finetune: task specific (will revisit)



Similar to transfer learning we saw in computer vision!

Input embedding

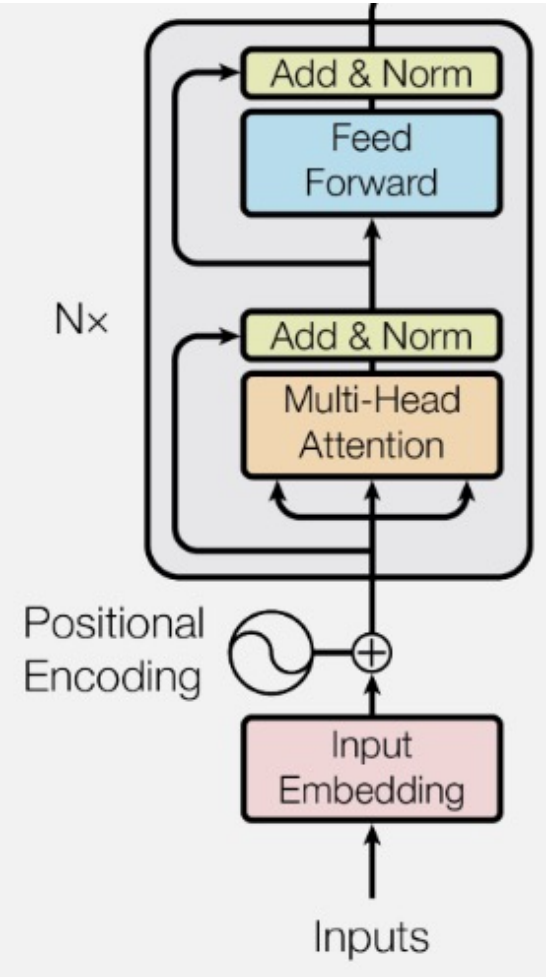
- Tokenized Input: word to subwords (recall fastText in last lecture)



Architecture

- Transformer encoder
- BERT base: N=12, d=768, H=12
- BERT large: N=24, d=1024, H=16

- Training: Wikipedia (2.5B words)
  +book corpus (0.8B words)
- Max input length: 512 subwords
  (≈256 each sentence)



A bit more on subword tokenizers

- Byte Pair Encoding (BPE): Merge the most frequent pair of tokens

corpus

```
5   l o w _
2   l o w e s t _
6   n e w e r _
3   w i d e r _
2   n e w _
```

vocabulary

```
_, d, e, i, l, n, o, r, s, t, w
```

corpus

```
5   l o w _
2   l o w e s t _
6   n e w er _
3   w i d er _
2   n e w _
```

vocabulary

```
_, d, e, i, l, n, o, r, s, t, w, er
```

Example from https://web.stanford.edu/~jurafsky/slp3/13.pdf

BPE (cont.)

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er, er_
2 l o w e s t _	
6 n e w er_	
3 w i d er_	
2 n e w _	

corpus	vocabulary
5 l o w _	_, d, e, i, l, n, o, r, s, t, w, er, er_, ne
2 l o w e s t _	
6 ne w er_	
3 w i d er_	
2 ne w _	

Drawback of BPE

- Small non-meaningful subwords

Natural Language Engineering (2020), **26**, pp. 375–382
doi:10.1017/S1351324920000145

CAMBRIDGE UNIVERSITY PRESS

EMERGING TRENDS

Emerging trends: Subwords, seriously?

Kenneth Ward Church

Baidu, USA
E-mail: kenneth.ward.church@gmail.com

Abstract

Subwords have become very popular, but the BERT[a] and ERNIE[b] tokenizers often produce surprising results. Byte pair encoding (BPE) trains a dictionary with a simple information theoretic criterion that sidesteps the need for special treatment of unknown words. BPE is more about training (populating a dictionary of word pieces) than inference (parsing an unknown word into word pieces). The parse at inference time can be ambiguous. Which parse should we use? For example, "electroneutral" can be parsed as electron-eu-tral or electro-neutral, and "bidirectional" can be parsed as bid-ire-ction-al and bi-directional. BERT and ERNIE tend to favor the parse with more word pieces. We propose minimizing the number of word pieces. To justify our proposal, a number of criteria will be considered: sound, meaning, etc. The prefix, bi-, has the desired vowel (unlike bid) and the desired meaning (bi is Latin for two, unlike bid, which is Germanic for offer).

Original:	`corrupted`	Original:	`Completely preposterous suggestions`
BPE:	`cor rupted`	BPE:	`Comple t ely prep ost erous suggest ions`

Examples from https://web.stanford.edu/~jurafsky/slp3/13.pdf

Wordpiece

- Initialize with a set of all characters
- Repeat till there are V wordpieces
  - Train an n-gram language model, using the current set
  - Consider concatenating two word pieces, so that the resulting n-gram has biggest likelihood increase

SentencePiece

- A library implementing BPE and another method called *Unigram*
- How Unigram works
  - Fix token set, learn probabilistic split of words (into these tokens), via EM
  - Prune away subwords with low probabilities

Original:	corrupted	Original:	Completely preposterous suggestions
BPE:	cor rupted	BPE:	Comple t ely prep ost erous suggest ions
Unigram:	corrupt ed	Unigram:	Complete ly pre post er ous suggestion s

Example from https://web.stanford.edu/~jurafsky/slp3/13.pdf

Agenda

- Encoder Model
  - ELMO
  - Masked LM and BERT
- Decoder Model
  - GPT
- Enc-Dec Model
  - Translation
  - Speech Recognition

GPT-1

- Pretraining

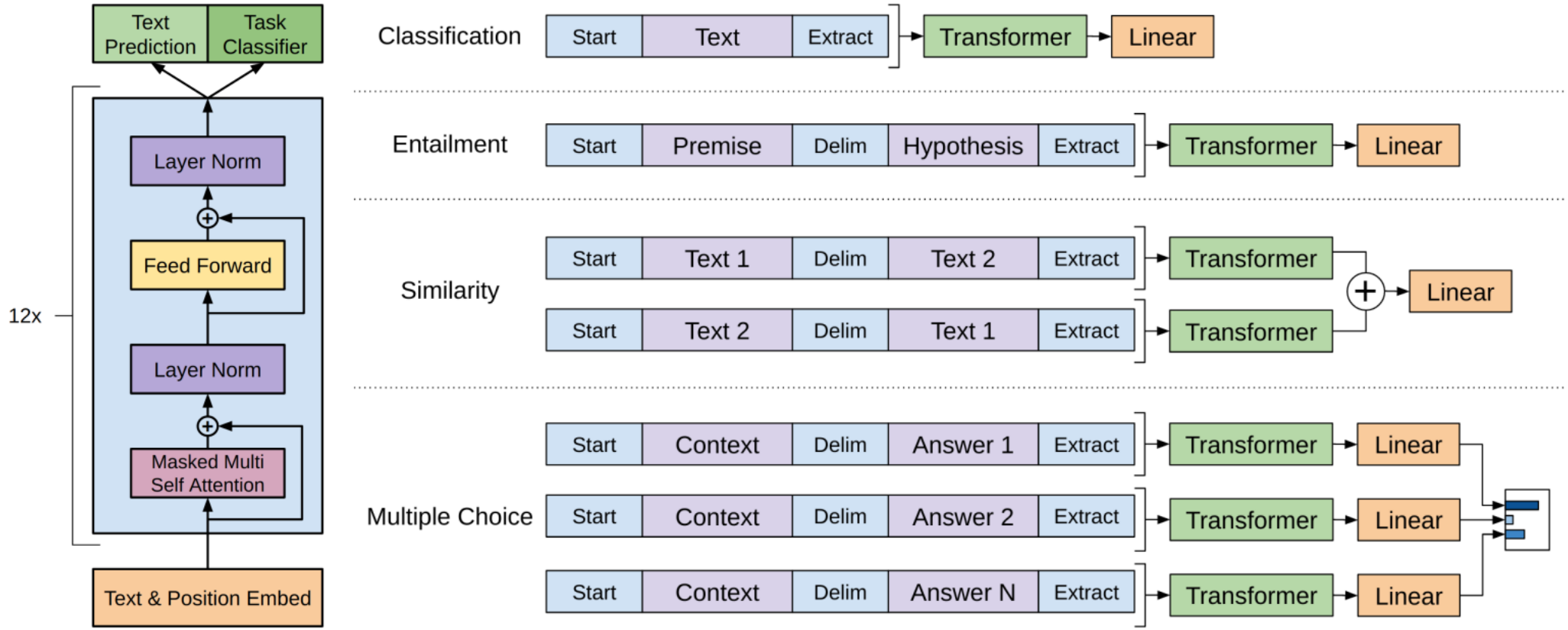
$$\min_{\Theta} \sum_{i} -\log p(x_i | x_{i-k}, \ldots, x_{i-1}; \Theta)$$

  where k is context window size

- Supervised fintuning

$$\min_{\Theta} \sum_{(x,y)} -\log p(y | x_1, \ldots, x_m)$$

GPT-1: architecture



[Radford et. al 2018]

GPT-2

- Training

$$\min_{\Theta} \sum_{i} -\log p(output|input, task; \Theta)$$

- Example

(task = translate to french, input = english text, output = french text)

- No finetuning used

Radford et. al (2019)

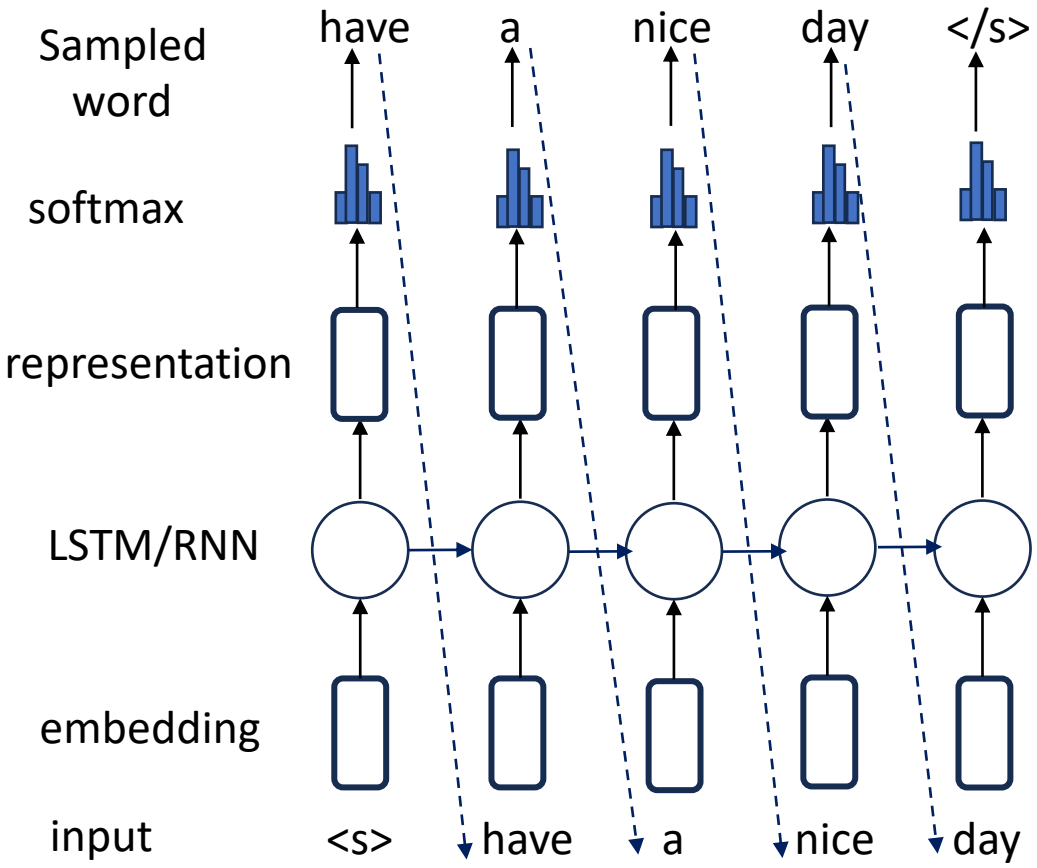
Some Results from GPT-2

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

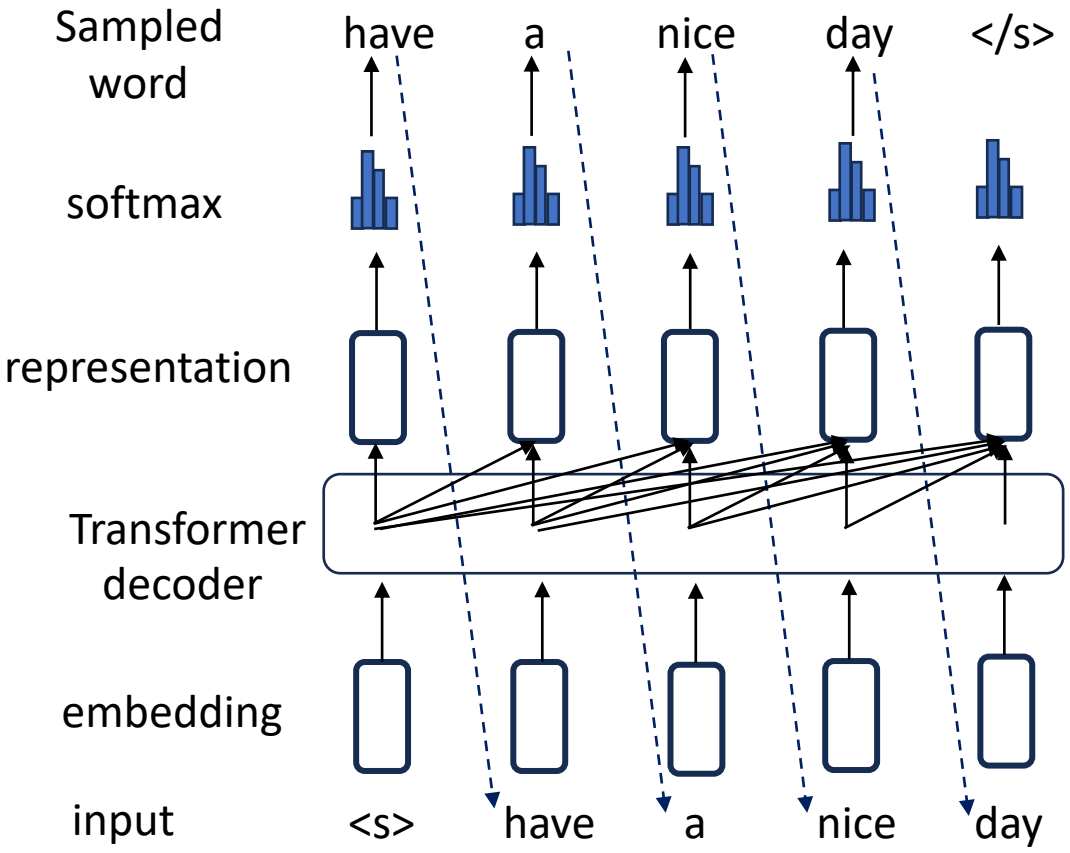
Table 3. Zero-shot results on many datasets. No training or fine-tuning was performed for any of these results. PTB and WikiText-2 results are from (Gong et al., 2018). CBT results are from (Bajgar et al., 2016). LAMBADA accuracy result is from (Hoang et al., 2018) and LAMBADA perplexity result is from (Grave et al., 2016). Other results are from (Dai et al., 2019).

Generation from Decoders

LSTM based Decoder



transformer based Decoder



Agenda

- Encoder Model
  - ELMO
  - Masked LM and BERT
- Decoder Model
  - GPT
- Enc-Dec Model
  - Translation
  - Speech Recognition

Classical Approach

Statistical Machine Translation

- Bayesian Rule

$$T^* = arg\max_T P(T|S) = arg\max_T P(S|T)P(T)$$

- $P(S|T)$: translation model, faithfulness
- $P(T)$: language model, fluency
- IBM models:
  - Alignment a
  - $P(S|T) = \sum_a P(S, a|T)$

Alignment

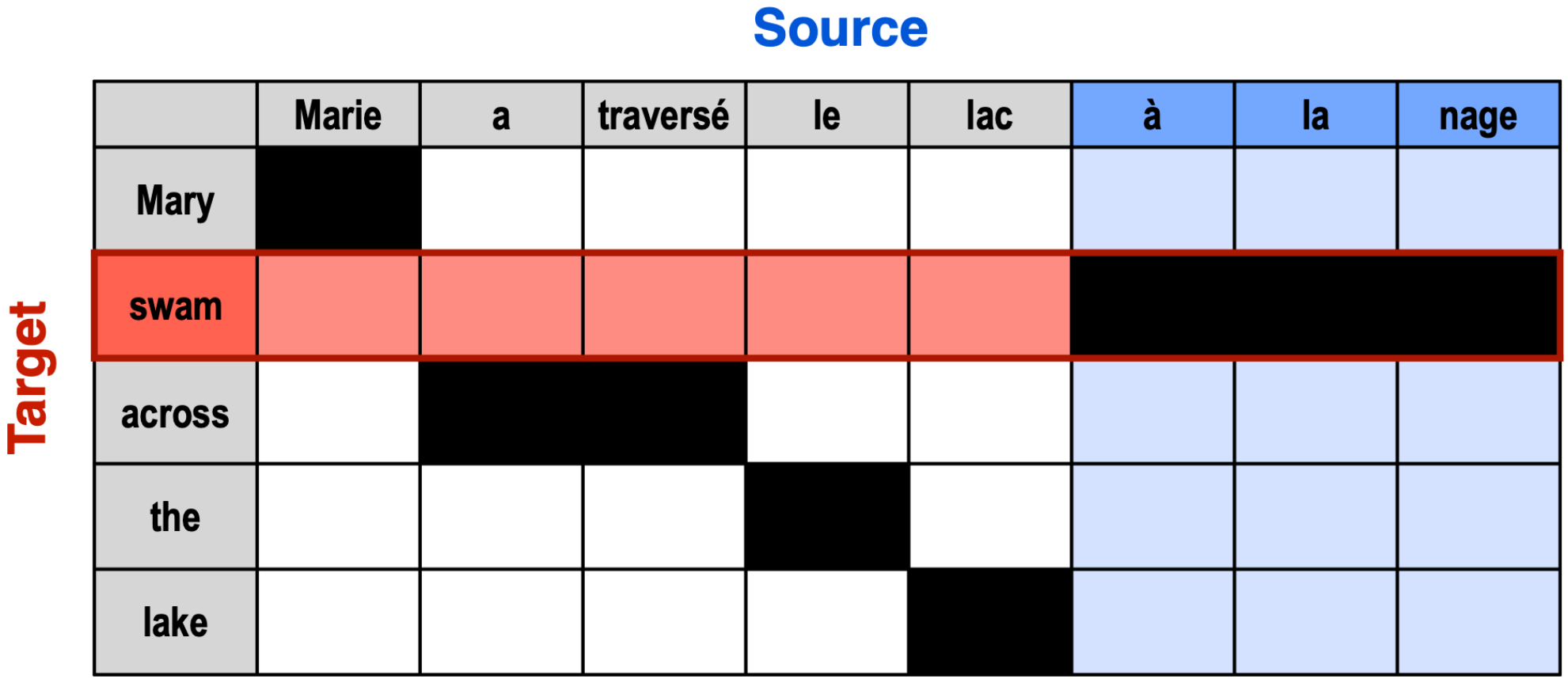


Illustration from https://courses.grainger.illinois.edu/CS447/fa2020/Slides/Lecture14.pdf

Architecture

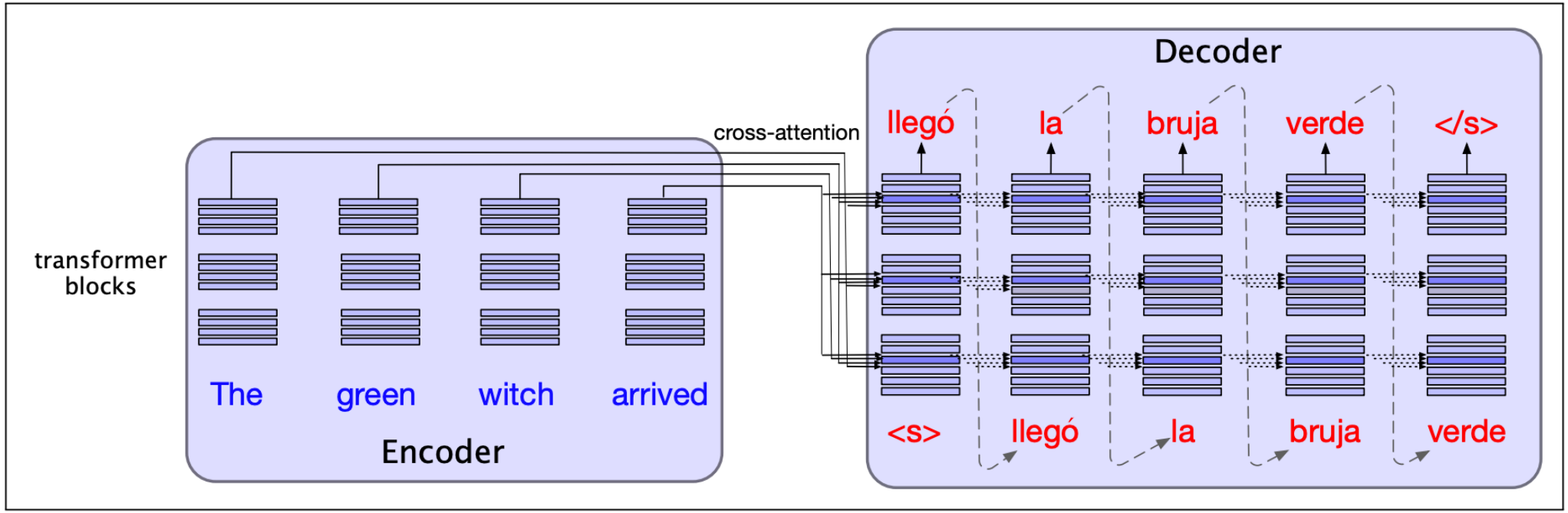
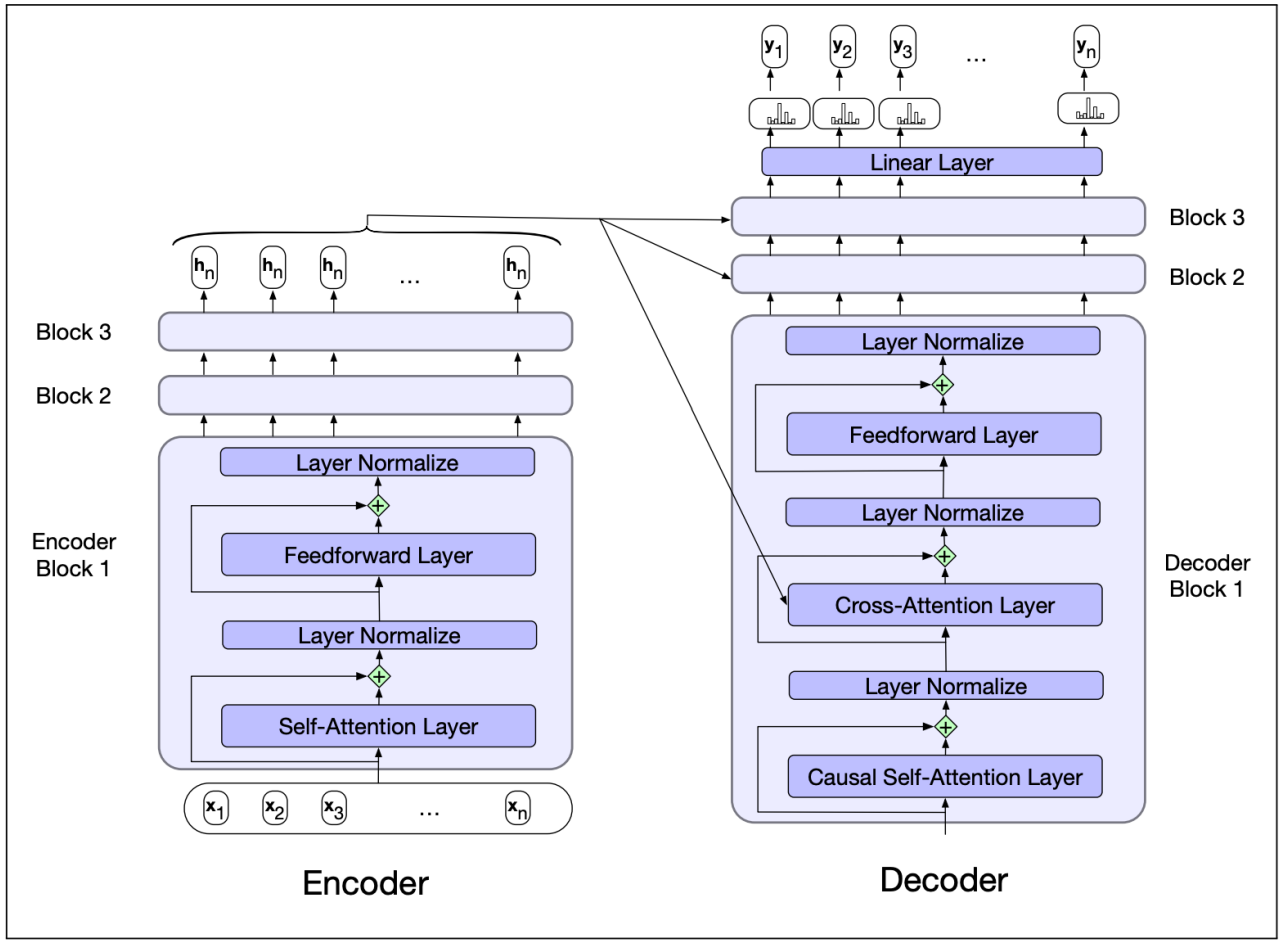


Figure 10.5 The encoder-decoder transformer architecture for machine translation. The encoder uses the transformer blocks we saw in Chapter 9, while the decoder uses a more powerful block with an extra **cross-attention** layer that can attend to all the encoder words. We'll see this in more detail in the next section.

Illustration from https://web.stanford.edu/~jurafsky/slp3/13.pdf

Zoom in for cross-attention

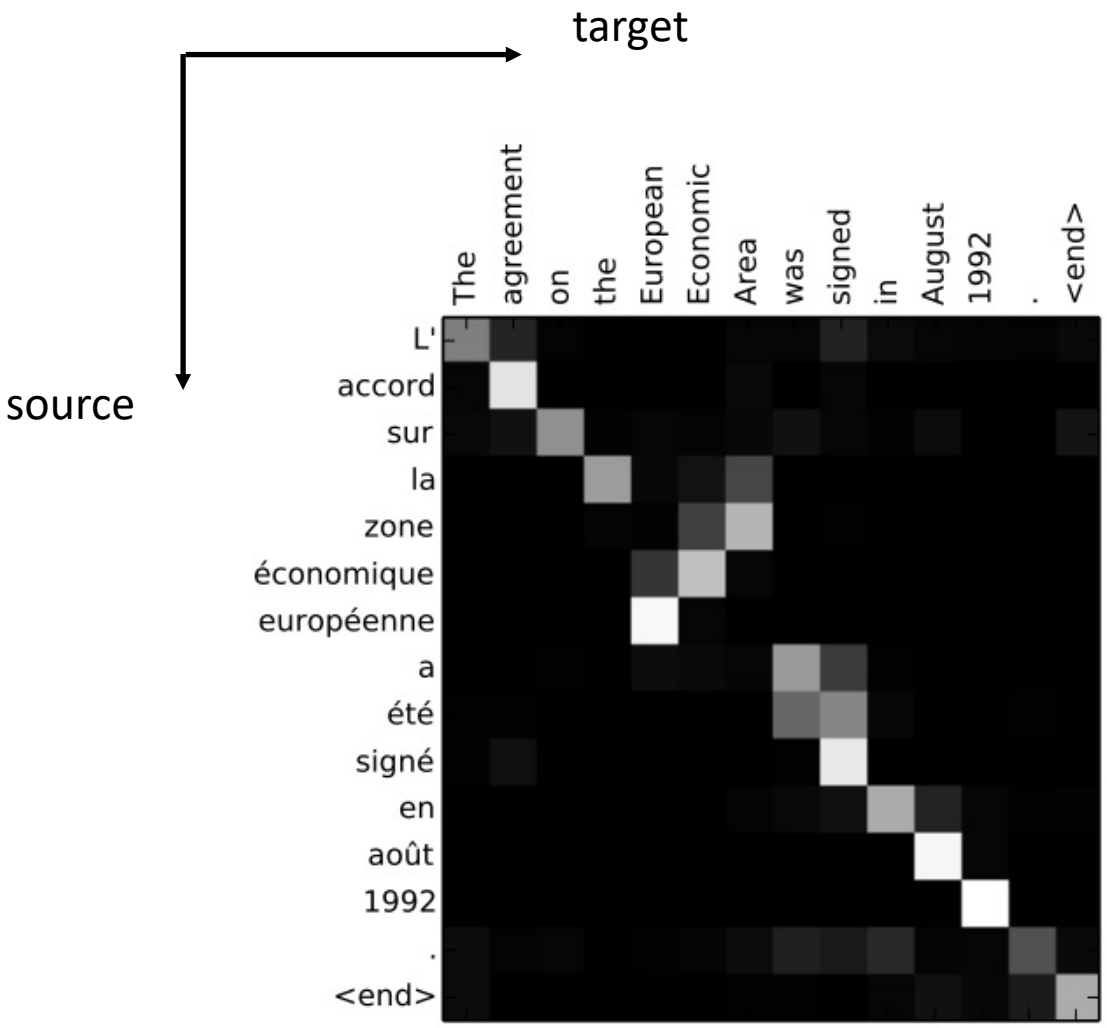


$$\mathbf{Q} = \mathbf{W}^{\mathbf{Q}}\mathbf{H}^{dec[i-1]};\ \mathbf{K} = \mathbf{W}^{\mathbf{K}}\mathbf{H}^{enc};\ \mathbf{V} = \mathbf{W}^{\mathbf{V}}\mathbf{H}^{enc}$$

$$\text{CrossAttention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\intercal}}{\sqrt{d_k}}\right)\mathbf{V}$$

Illustration from https://web.stanford.edu/~jurafsky/slp3/13.pdf

Cross Attention is alignment



arXiv
https://arxiv.org › cs

Neural Machine Translation by Jointly Learning to Align ...

by D Bahdanau · 2014 · Cited by 31472 — With this new approach, we achieve a **translation** performance comparable to the existing state-of-the-art phrase-based **system** on the task of ...

Note the alignment is neither diagonal, nor triangular!

Training

- Minimize cross-entropy loss

$$\sum_t H(y_t, \hat{y}_t)$$

- Consider t-th input to decoder
- feeding ground-truth y_t has exposure bias
- As at inference, we feed in decoder output of last timestep, $\hat{y}_t$
- Scheduled sampling: mimic inference time by feeding $\hat{y}_t$
- Hybrid scheme?

Inference

- Greedy
  - Each time step take the most likely token and input to next step
- Beam Search

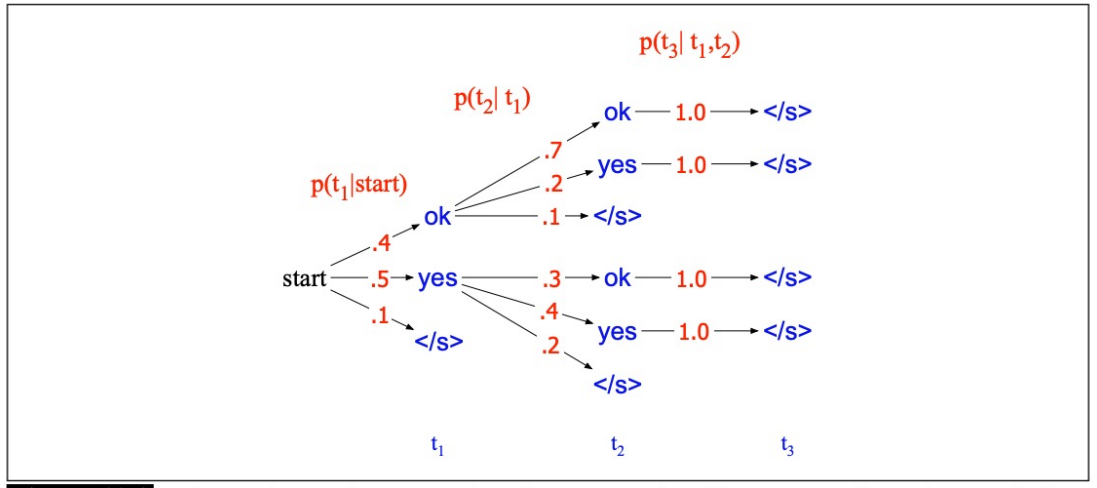


Figure 10.8 A search tree for generating the target string $T = t_1, t_2, \ldots$ from the vocabulary $V = \{\text{yes}, \text{ok}, \text{<s>}\}$, showing the probability of generating each token from that state. Greedy search would choose *yes* at the first time step followed by *yes*, instead of the globally most probable sequence *ok ok*.

Illustration from https://web.stanford.edu/~jurafsky/slp3/10.pdf

Evaluation Metric: BLEU score

- Count the number of common n-grams

Reference: the cat is on the mat

Decoded: on mat sits the cat

N=1: “on”, “mat”, “the”, “cat”, precision $p_1 = 4/5$

N=2: “the cat”, precision $p_2 = 1/4$

- Calculate brevity penalty, punish very short decoded sentence, $\exp\left(1 - \max\left(1, {}^{L_{ref}}/_{L_{decoded}}\right)\right) = e^{1-6/5} = 0.8187$
- BLEU = brevity penalty $\times \exp(\sum_n w_n \ln p_n)$, higher is better

Extensions

- Simultaneous Translation

ACL Anthology
https://aclanthology.org › ...

STACL: Simultaneous Translation with Implicit Anticipation ...

by M Ma · 2019 · Cited by 194 — **Simultaneous translation**, which translates sentences before they are finished, is use- ful in many scenarios but is notoriously dif- ficult due to word-order ...

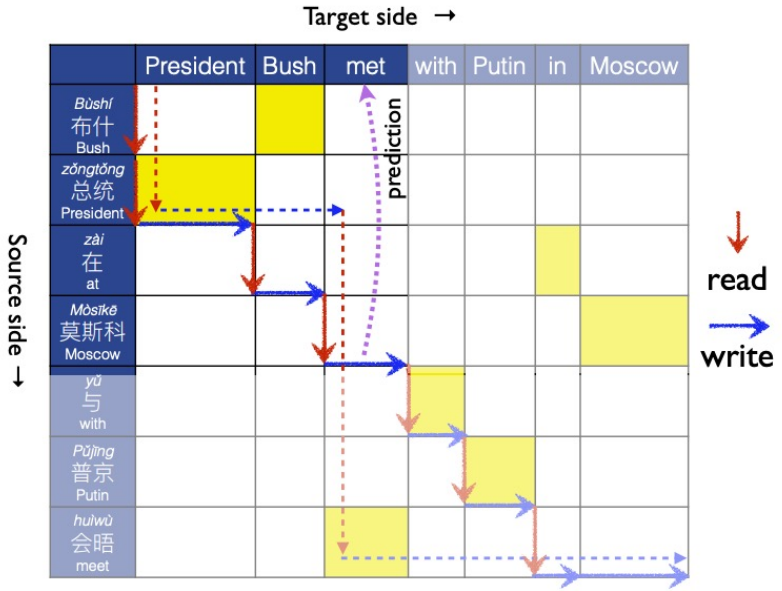


Figure 1: Our wait-k model emits target word y_t given source-side prefix $x_1 \ldots x_{t+k-1}$, often before seeing the corresponding source word (here k=2, outputing y_3="met" before x_7="*huìwù*"). Without anticipation, a 5-word wait is needed (dashed arrows). See also Fig. 2.

Extensions

- Multilingual Translation
- Language encoding token: l_s , l_t

$$\begin{aligned} \mathbf{h} &= \text{encoder}(x, l_s) \\ y_{i+1} &= \text{decoder}(\mathbf{h}, l_t, y_1, \ldots, y_i)) \quad \forall i \in [1, \ldots, m] \end{aligned}$$

Extensions

UNSUPERVISED MACHINE TRANSLATION
USING MONOLINGUAL CORPORA ONLY

Guillaume Lample † ‡ , **Alexis Conneau** † , **Ludovic Denoyer** ‡ , **Marc'Aurelio Ranzato** †
† Facebook AI Research,
‡ Sorbonne Universités, UPMC Univ Paris 06, LIP6 UMR 7606, CNRS

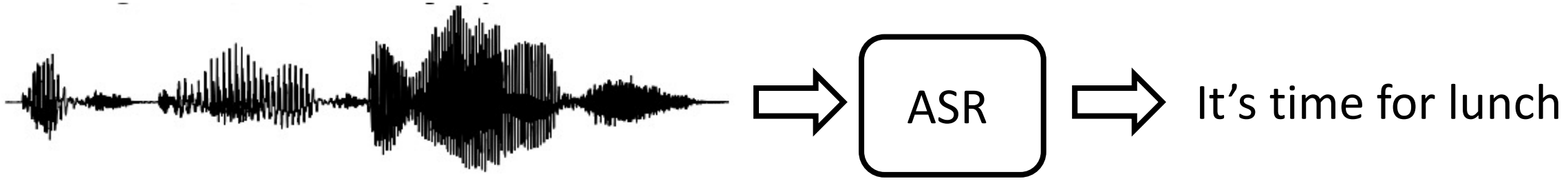
- Unsupervised Machine Translation
  - Source and target text has no correspondence
- Method
  - Initialize: Word-to-word translation (bilingual lexicon induction)
  - Train encoder-decoder by
    - Sample source sentence x, translation y using current model
    - Train as if supervised case, using (x, y)

Agenda

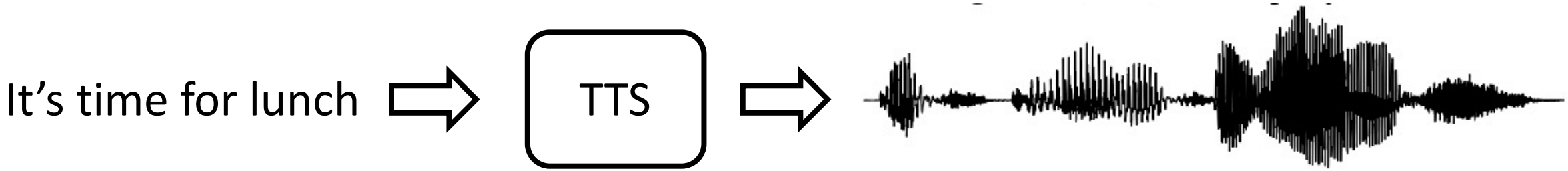
- Encoder Model
  - ELMO
  - Masked LM and BERT
- Decoder Model
  - GPT
- Enc-Dec Model
  - Translation
  - Speech Recognition

Two Tasks

- Automatic Speech Recognition (ASR)



- Speech synthesis, or Text-to-Speech (TTS)



Word Error Rate (WER) on WSJ eval92

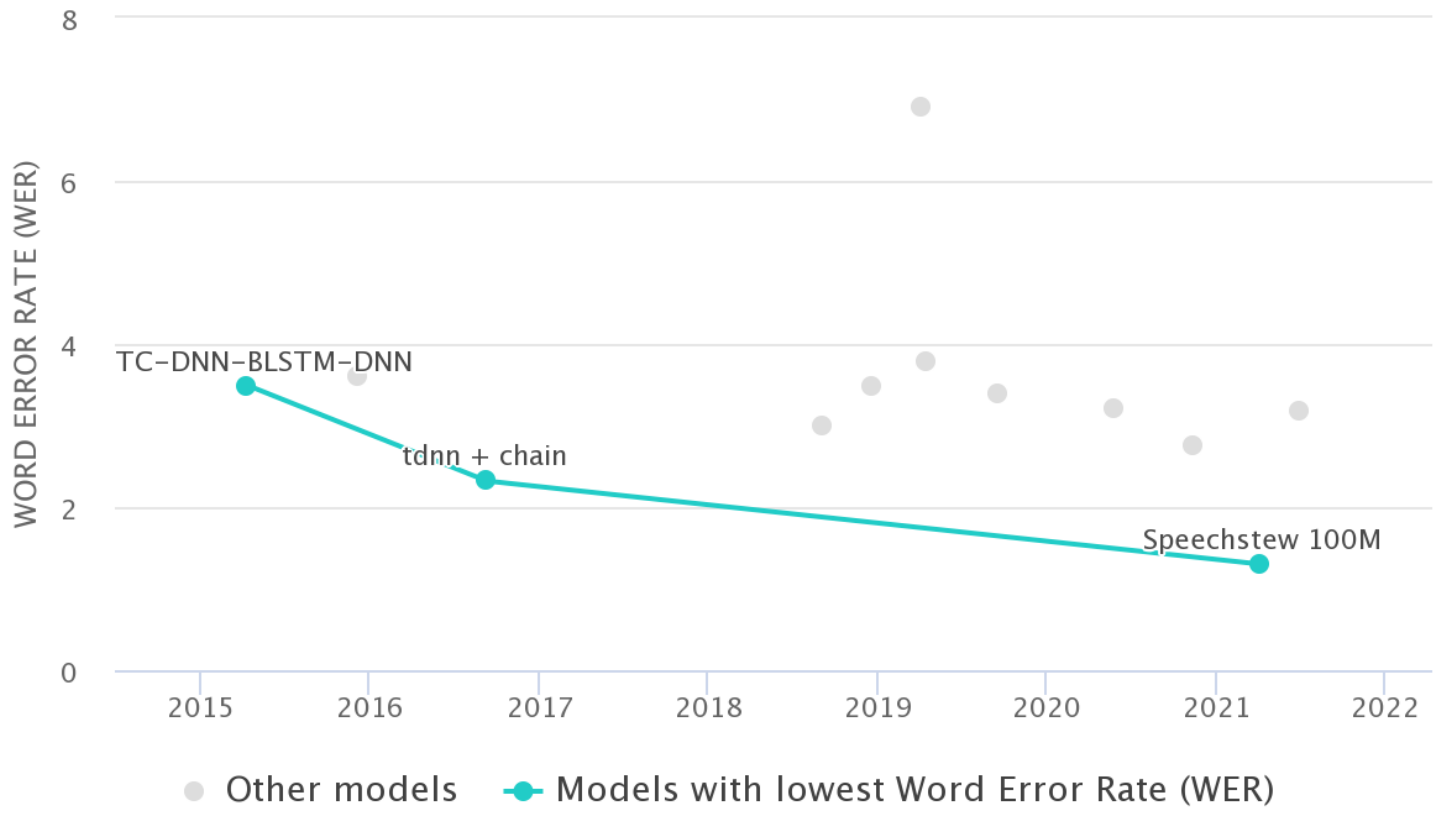


Image from https://paperswithcode.com/sota/speech-recognition-on-wsj-eval92

End-to-end ASR

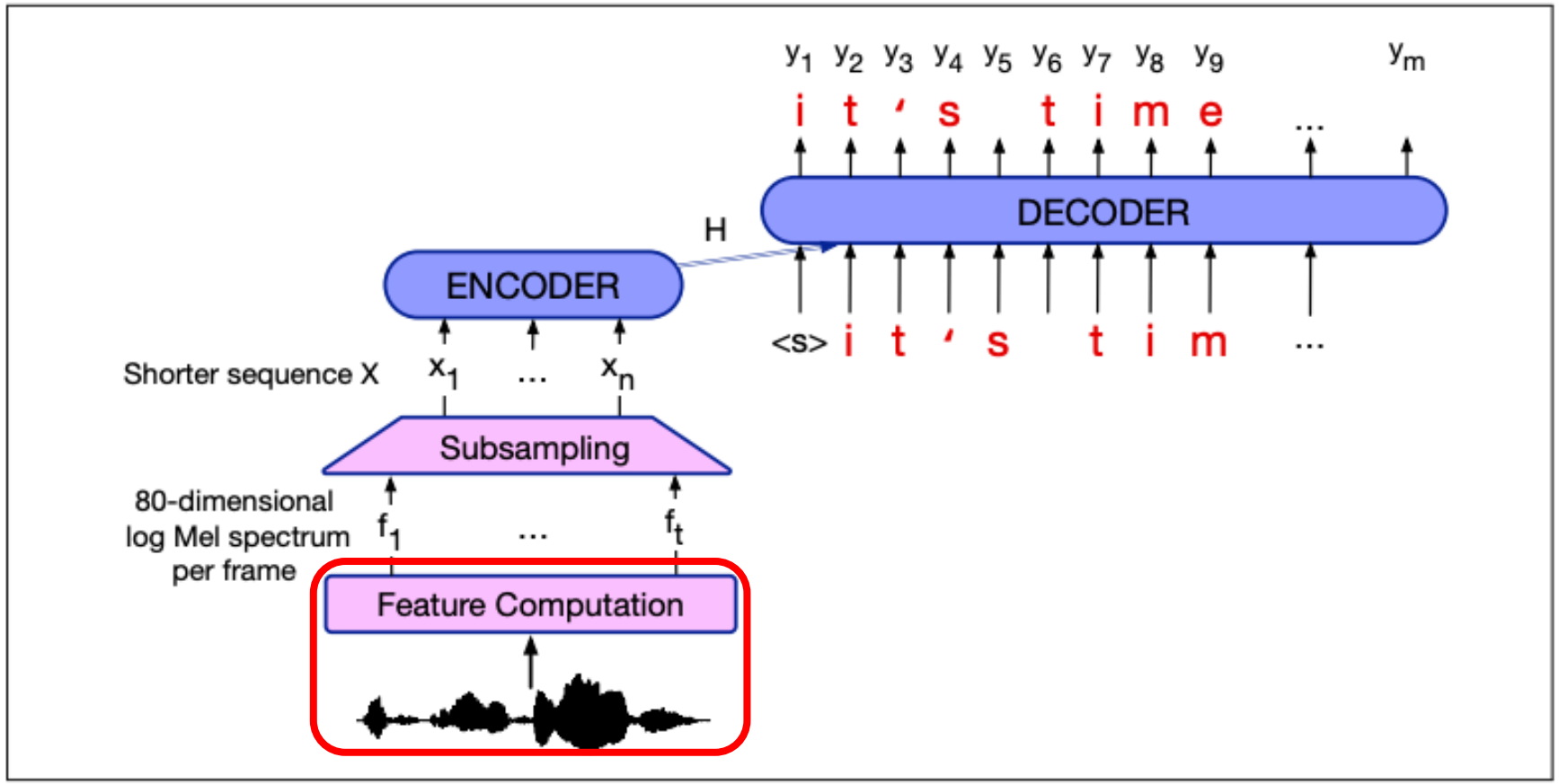


Figure 16.6 Schematic architecture for an encoder-decoder speech recognizer.

Illustration from https://web.stanford.edu/~jurafsky/slp3/16.pdf

Feature Computation

- Windowing
  - Rectangular window
    - Gibbs phenomenon
  - Hamming window

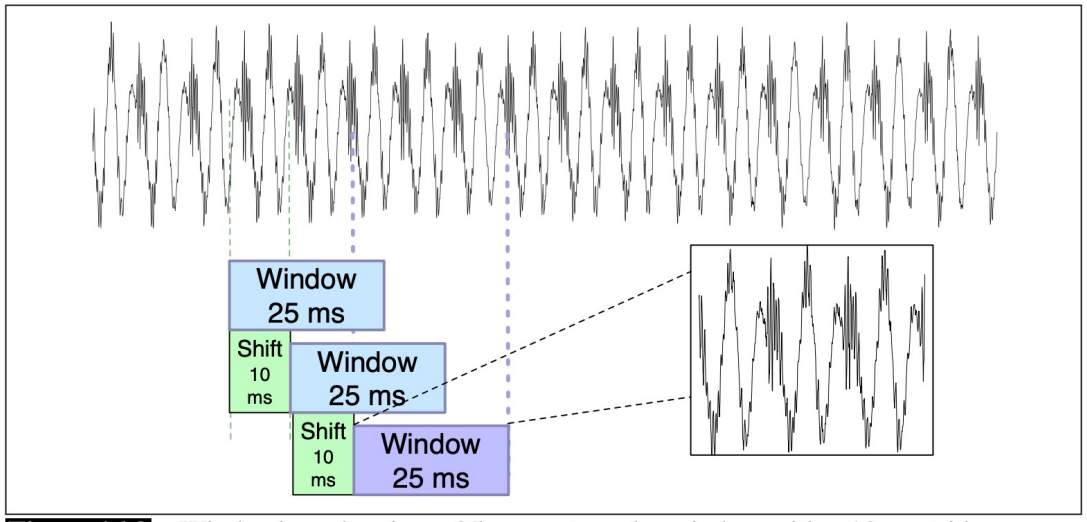
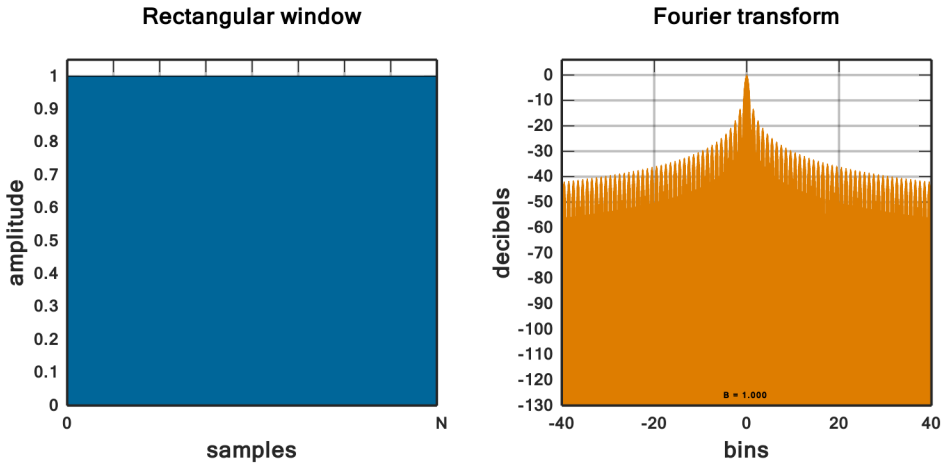


Figure 16.2 Windowing, showing a 25 ms rectangular window with a 10ms stride.



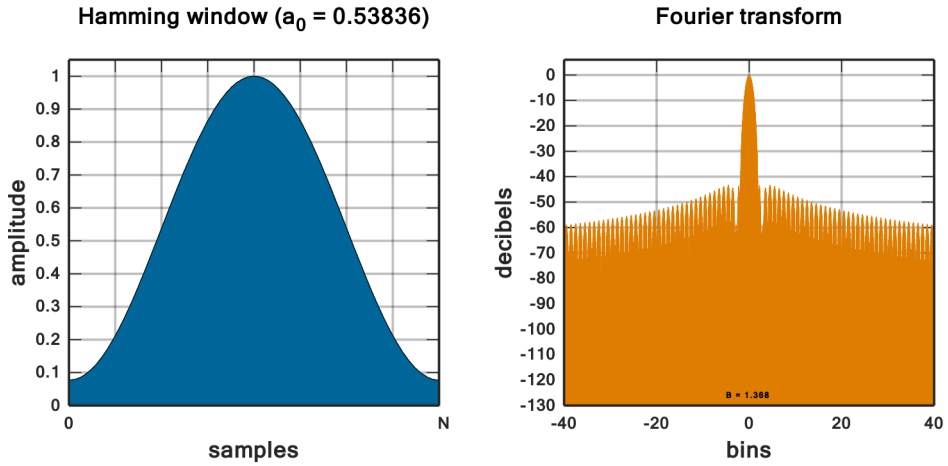


Illustration from https://web.stanford.edu/~jurafsky/slp3/16.pdf and https://en.wikipedia.org/wiki/Window_function

Convert to Frequency domain

- Apply FFT inside each window
- Apply Mel Filter banks
  - Why? Human hearing is less sensitive at higher frequency

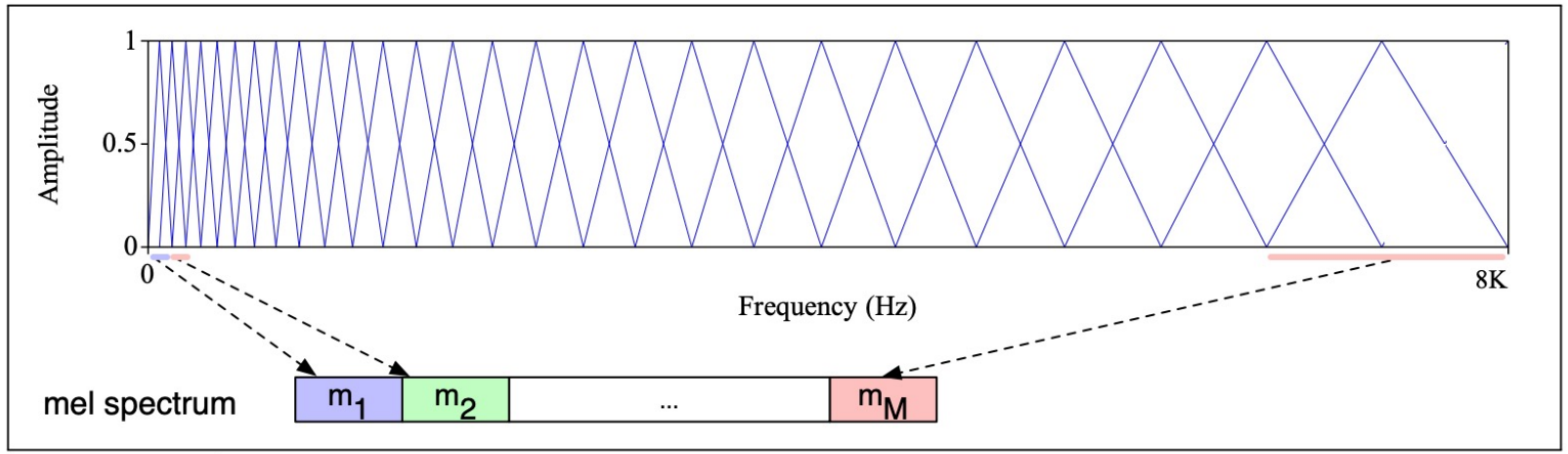
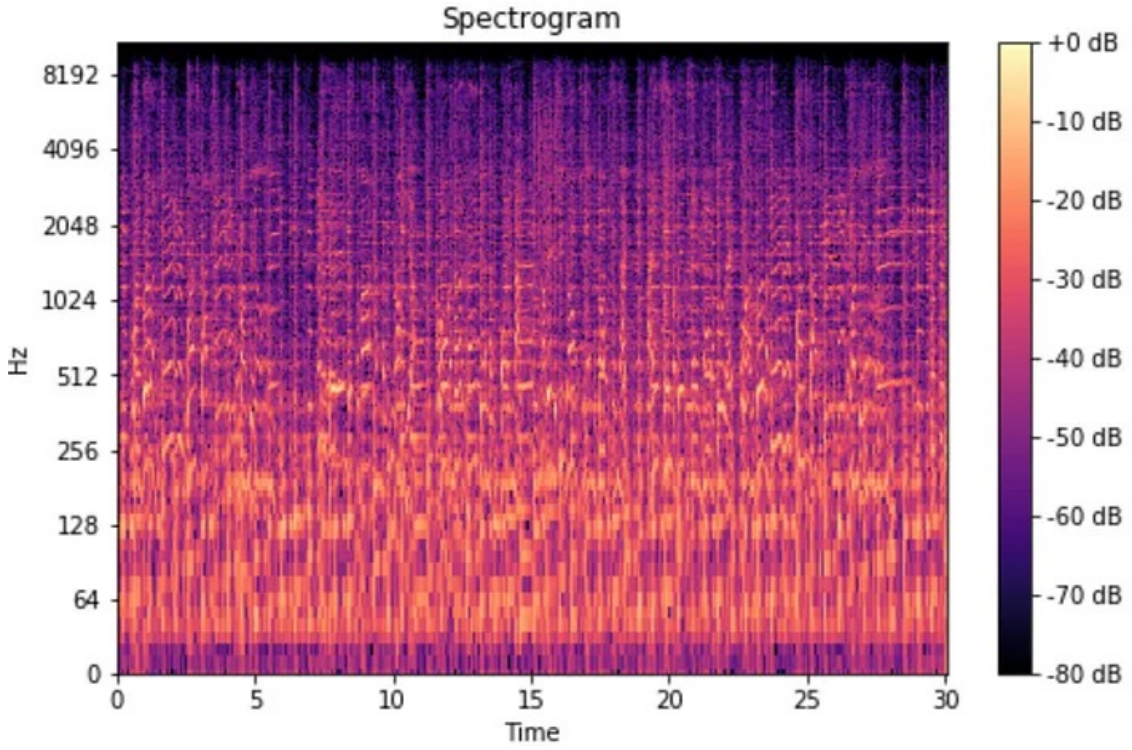


Figure 16.5 The mel filter bank (Davis and Mermelstein, 1980). Each triangular filter, spaced logarithmically along the mel scale, collects energy from a given frequency range.

Illustration from https://web.stanford.edu/~jurafsky/slp3/16.pdf

Before and after Mel



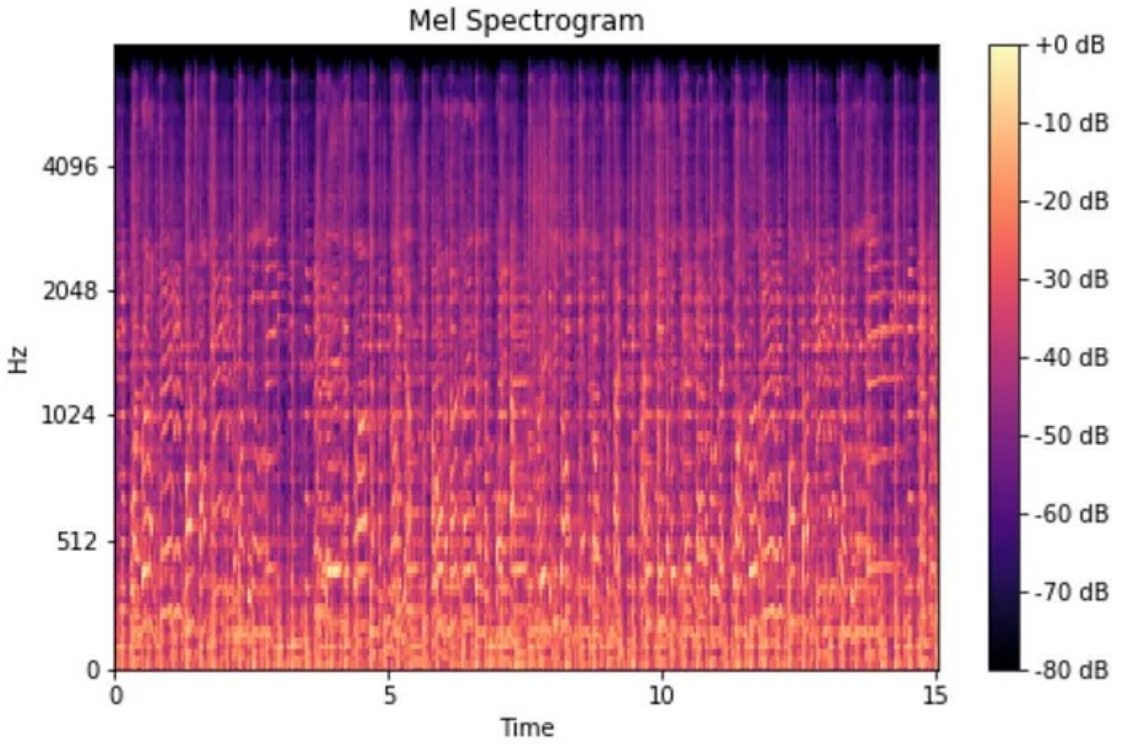


Illustration from https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

End-to-end ASR

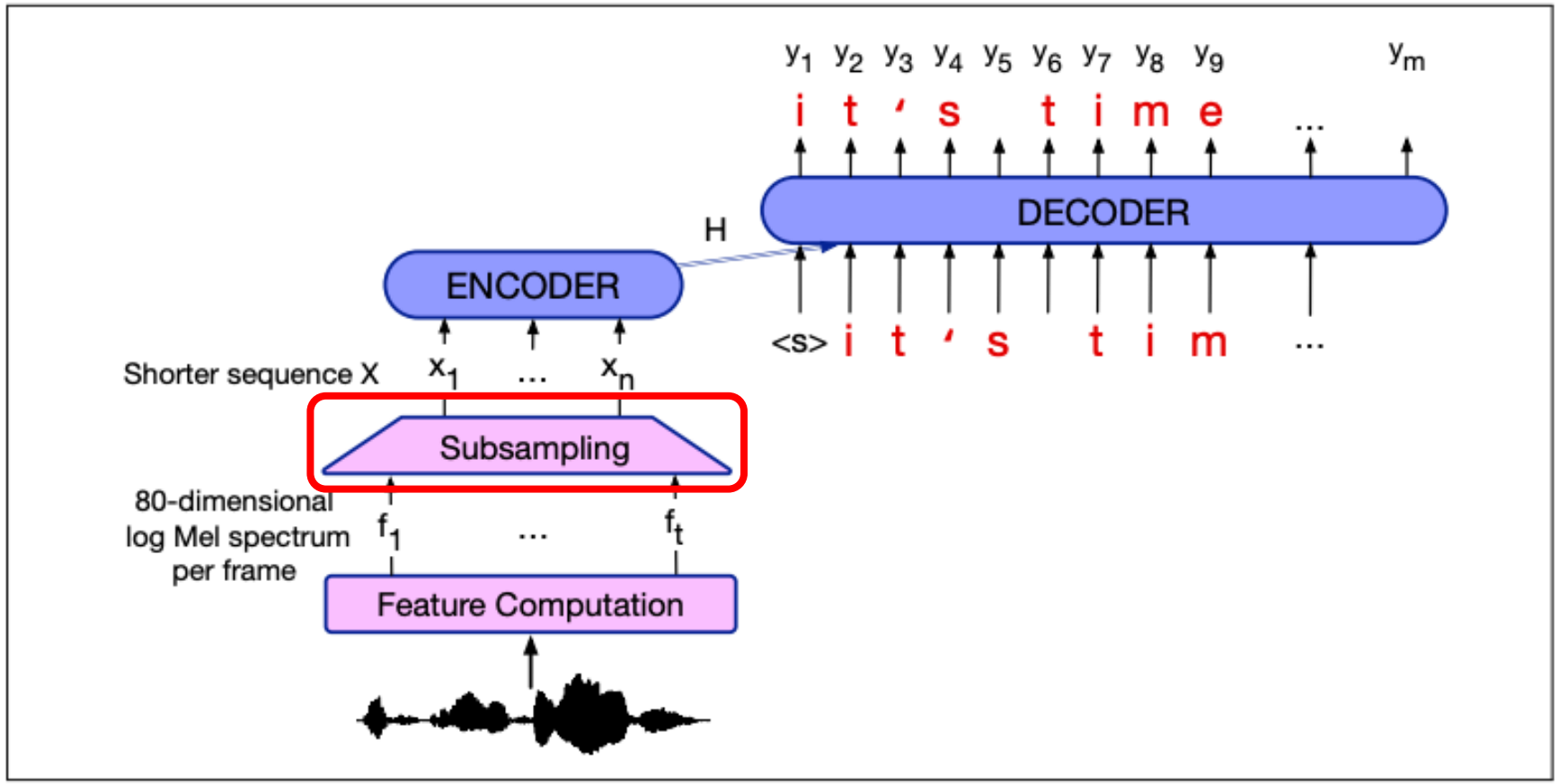


Figure 16.6 Schematic architecture for an encoder-decoder speech recognizer.

Illustration from https://web.stanford.edu/~jurafsky/slp3/16.pdf

Subsampling

- Input is very long sequence, e.g.,
  - 2s audio is 200 frames, assuming 10ms stride at windowing
- Ways to lower the frame rate:
  - Stack adjacent frames
  - 1D filter along time axis

End-to-end ASR

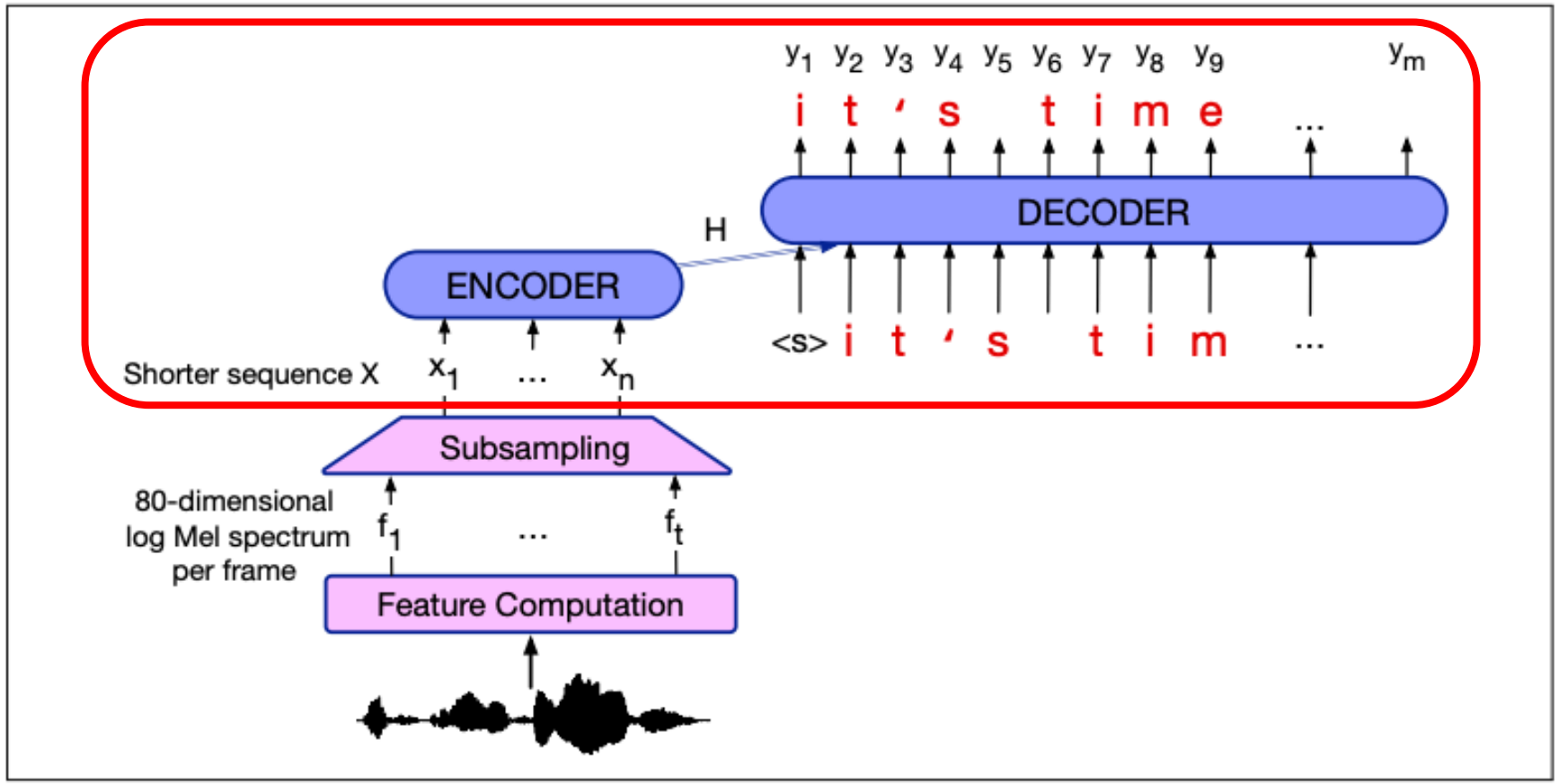
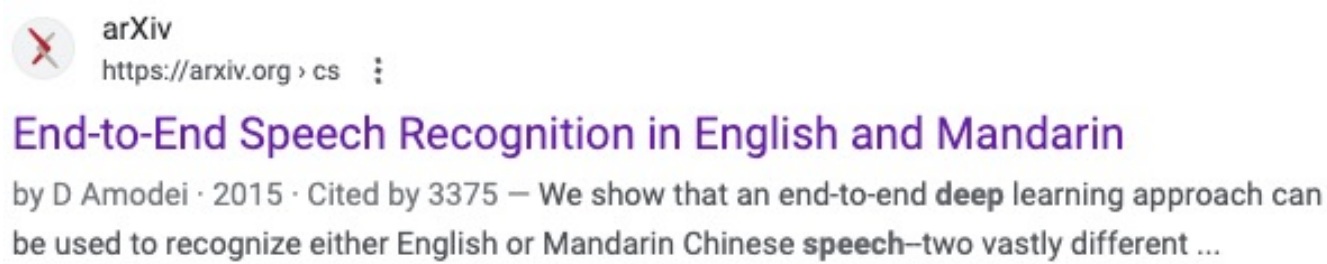


Figure 16.6 Schematic architecture for an encoder-decoder speech recognizer.

Illustration from https://web.stanford.edu/~jurafsky/slp3/16.pdf

What are the y’s

- Characters



- Words, less common

- Subwords, popular now

LSTM based Encoder-Decoder

LISTEN, ATTEND AND SPELL: A NEURAL NETWORK FOR LARGE VOCABULARY CONVERSATIONAL SPEECH RECOGNITION

William Chan — Carnegie Mellon University

Navdeep Jaitly, Quoc Le, Oriol Vinyals — Google Brain

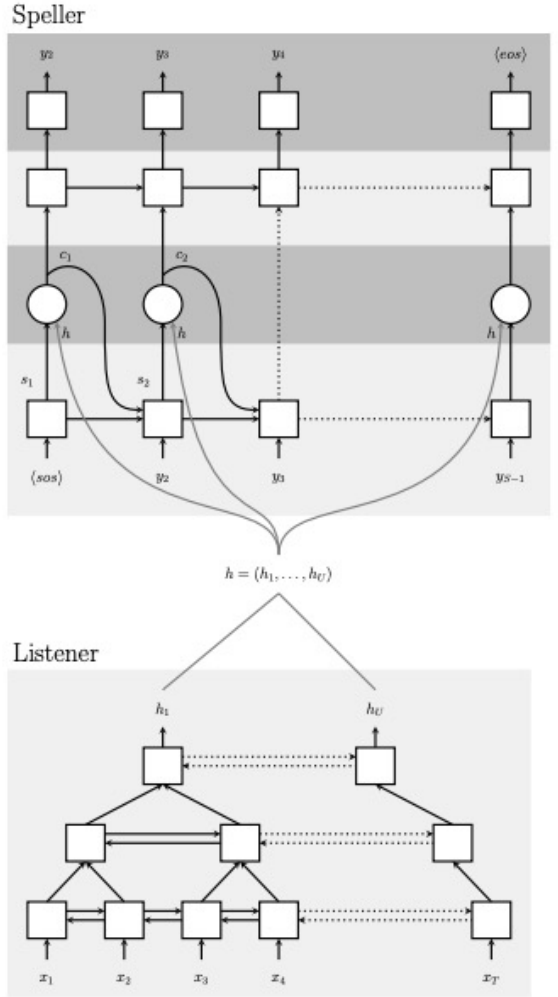


Fig. 1: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence $\mathbf{x}$ into high level features $\mathbf{h}$, the speller is an attention-based decoder generating the $\mathbf{y}$ characters from $\mathbf{h}$.

Transformer based Encoder-Decoder

Conformer: Convolution-augmented Transformer for Speech Recognition

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, Ruoming Pang

Google Inc.

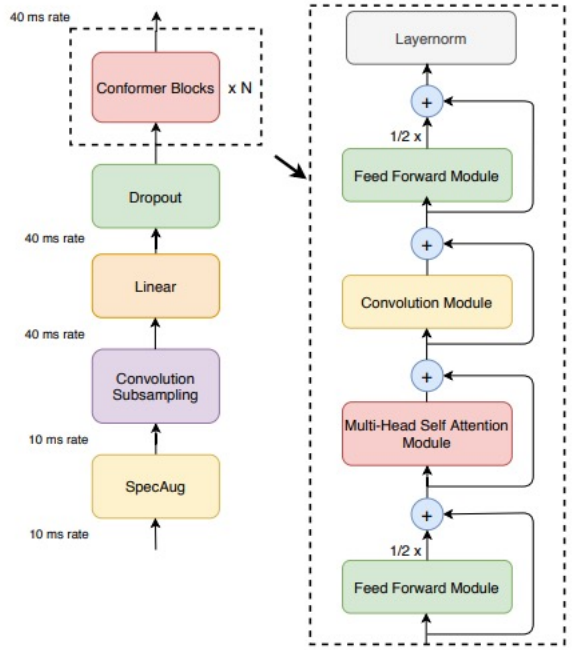


Figure 1: ***Conformer encoder model architecture.*** *Conformer comprises of two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed self-attention and convolution modules. This is followed by a post layernorm.*

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

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

Use Language Model

- “two” and “to” sounds alike
- How to make sure we decode the right one?
- Use Language Model
  - Easy way: rescoring N-best list
  - Hard way: add LM score at decoding
- We can finetune the language model for ASR

Large Margin Neural Language Model

Jiaji Huang[1] **Yi Li**[1] **Wei Ping**[1] **Liang Huang**[1,2*]
[1] Baidu Research, Sunnyvale, CA, USA
[2] School of EECS, Oregon State University, Corvallis, OR, USA

Evaluation Metric: Word Error Rate

- Edit (Levenshtein) distance between reference and decoded

$$\text{Word Error Rate} = 100 \times \frac{\text{Insertions} + \text{Substitutions} + \text{Deletions}}{\text{Total Words in Correct Transcript}}$$

- Implementation based on Dynamic programming

		H	Y	U	N	D	A	I
	0	1	2	3	4	5	6	7
H	1	0	1	2	3	4	5	6
O	2	1	1	2	3	4	5	6
N	3	2	2	2	2	3	4	5
D	4	3	3	3	3	2	3	4
A	5	4	4	4	4	3	2	3

$$\text{lev}_{a,b}(i,j) = \begin{cases} 0 & , i = j = 0 \\ i & , j = 0 \text{ and } i > 0 \\ j & , i = 0 \text{ and } j > 0 \\ \min \begin{cases} \text{lev}_{a,b}(i-1,j) + 1 \\ \text{lev}_{a,b}(i,j-1) + 1 \\ \text{lev}_{a,b}(i-1,j-1) + [a_i \neq b_j] \end{cases} & , \text{ else} \end{cases}$$

Example from https://www.cuelogic.com/blog/the-levenshtein-algorithm