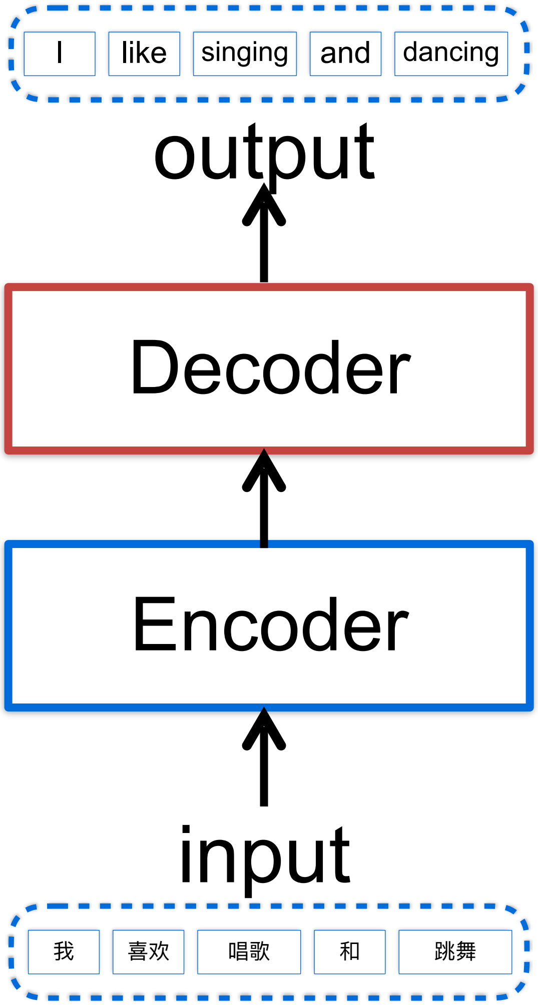
Lecture 8
Transformer

Lei Li and Yuxiang Wang
UCSB

Recap

- Tokenization for Text
  - Byte-Pair Encoding
- CNN language model
  - temporal convolution
- Recurrent Neural Network
- Long-short term memory
  - input, forget, and output gates
- Gated recurrent units

Encoder-Decoder Paradigm



A generic formulation for many tasks

Encoder-Decoder Paradigm

我喜欢唱歌和跳舞。 Machine Translation → I like singing and dancing.

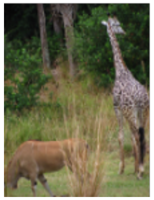


Image Captioning → A giraffe standing next to forest

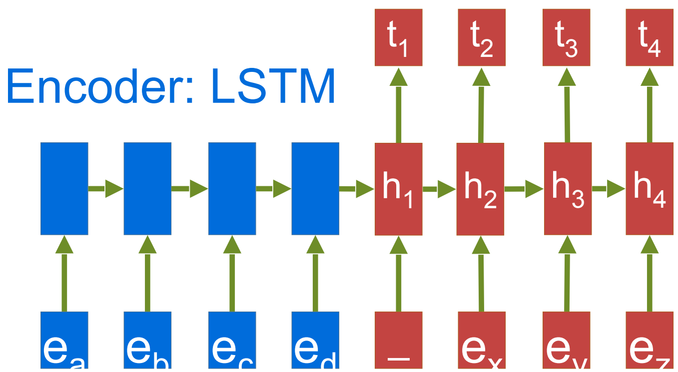
Automatic Speech Recognition → “Alexa, turn off the lights”

Graduate student reading papers on beach Text-to-Image Generation →

Sequence-toSeq Learning

- Machine translation as directly learning a function mapping from source sequence to target sequence

target:
The weather is nice



Source: 天 气 很 好 Decoder: LSTM

$$P(Y|X) = \prod P(y_t | y_{<t}, x)$$

Training loss: Cross-Entropy

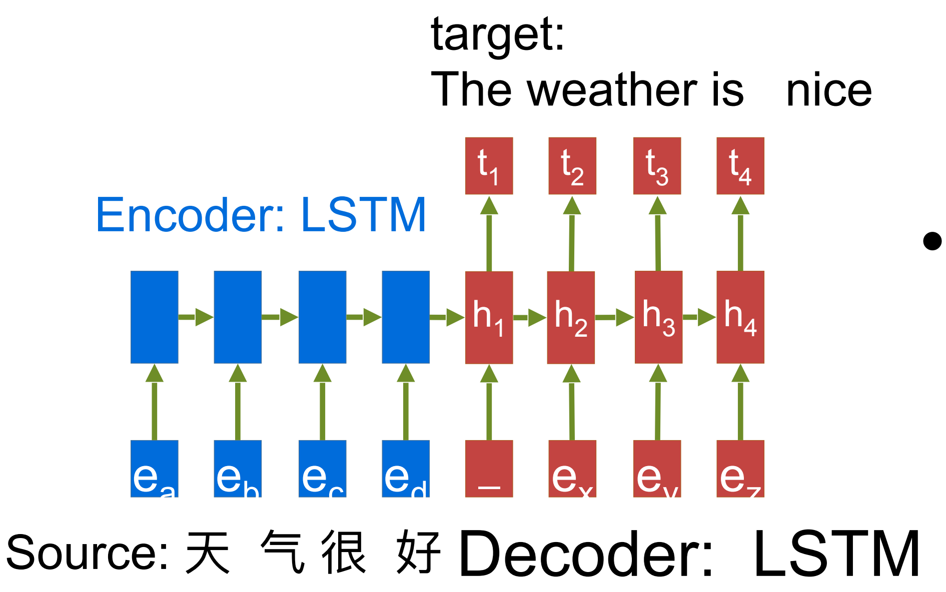
$$l = -\sum_n \sum_t \log f_\theta(x_n, y_{n,1}, \ldots, y_{n,t-1})$$

Teacher-forcing during training.

(pretend to know groundtruth for prefix)

Sutskever et al. Sequence to Sequence Learning with Neural Networks. 2014

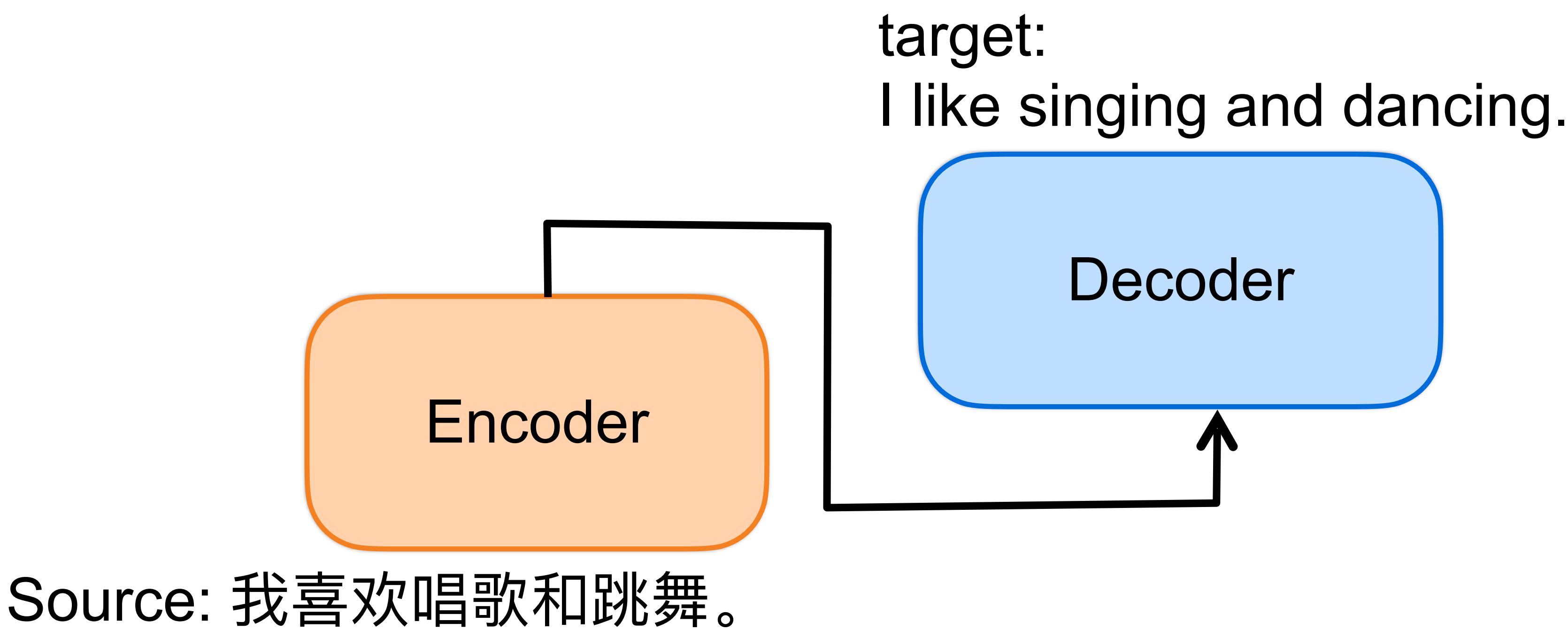
Limitation of RNN/LSTM



- No full context (only one-side)
  - Bidirectional LSTM encoder could alleviate
  - But still no long context
- Sequential computation in nature (encoder)
  - not possible to parallelize the computation
- Vanishing gradient

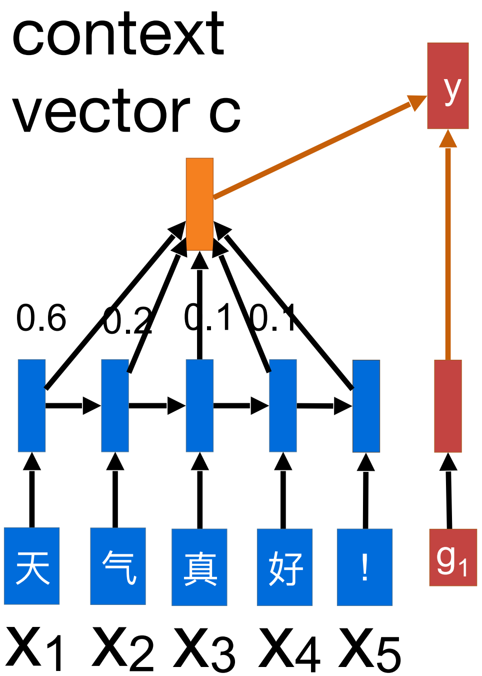
Motivation for New Network Architecture

- Full context and parallel: use Attention in both encoder and decoder
- no recurrent



Attention

Each output token depends on input tokens differently



A context vector c represents the related source context for current predicting word.

$$\alpha_{mj} = \mathrm{Softmax}(D(g_m, h_{1\ldots n})) = \frac{\exp(D(g_m, h_j))}{\sum_k \exp(D(g_m, h_k))}$$

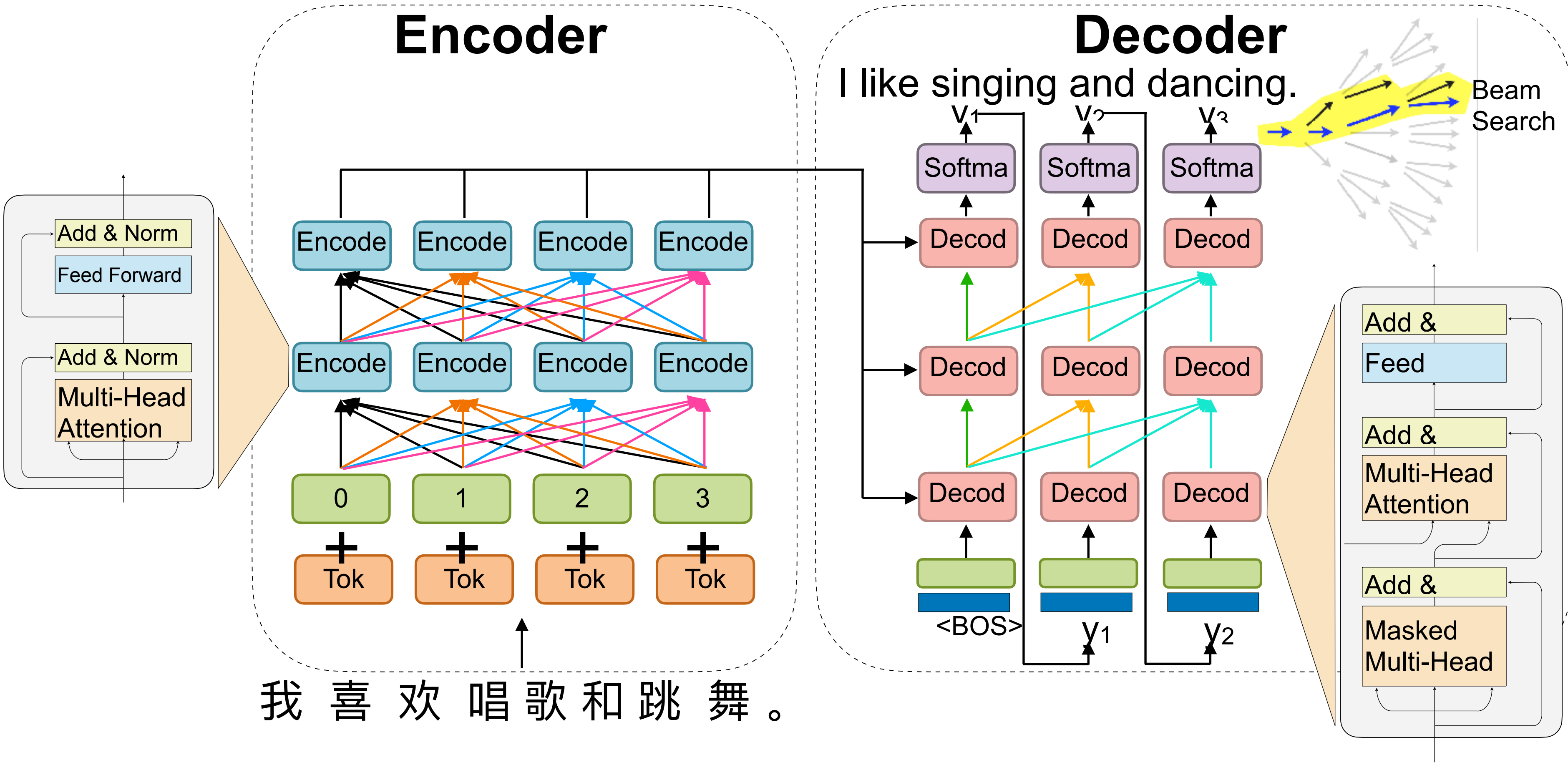
$$c_m = \sum_j \alpha_{mj} h_j$$

$$D(g_m, h_j) = g_m \cdot h_j$$

The probability of word y_i is computed as:

$$p(y_m) = \mathrm{Softmax}(W \cdot \begin{bmatrix} g_m \\ c_m \end{bmatrix} + b)$$

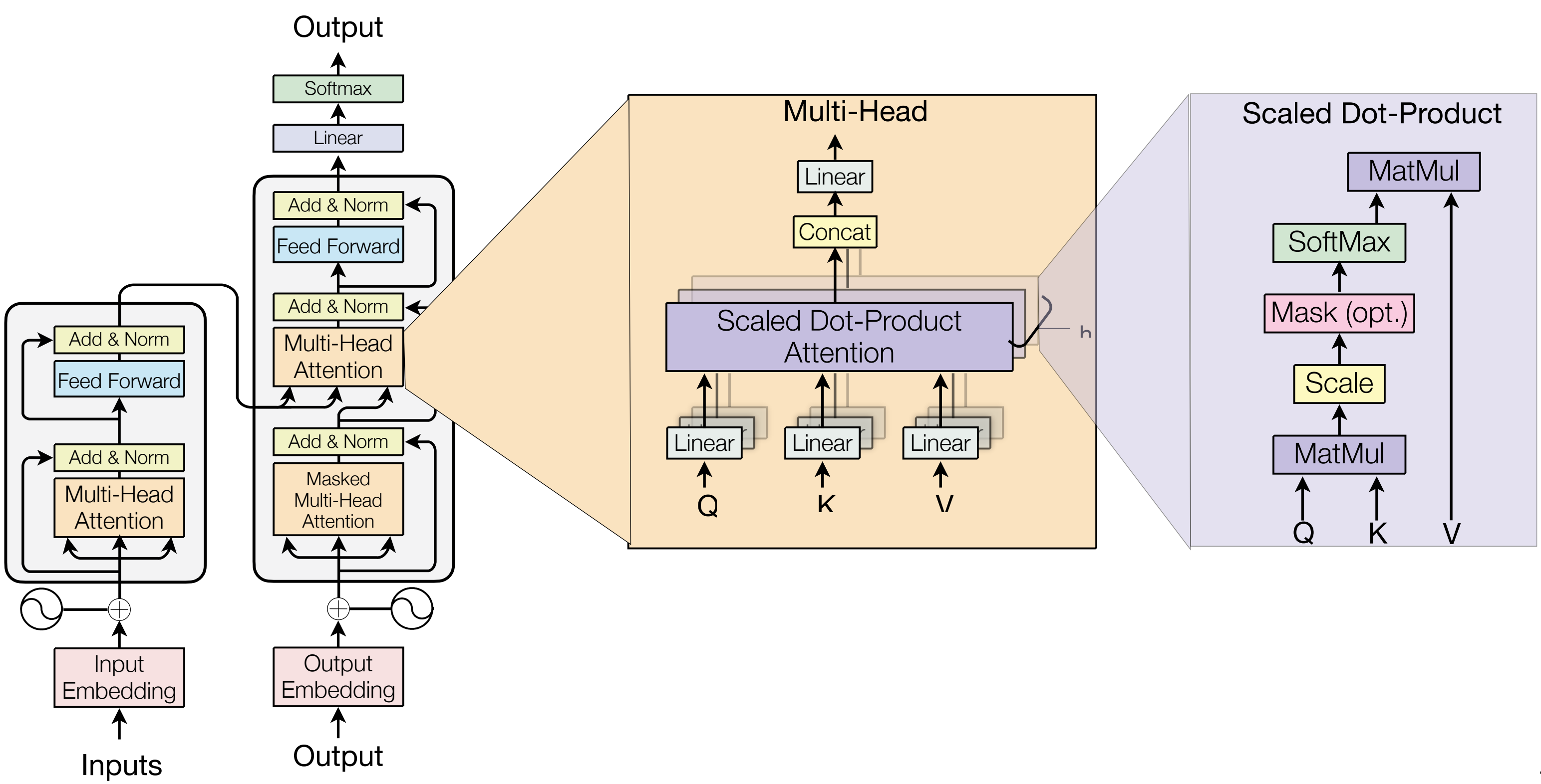
Transformer



Vaswani et al. Attention is All You Need. 2017

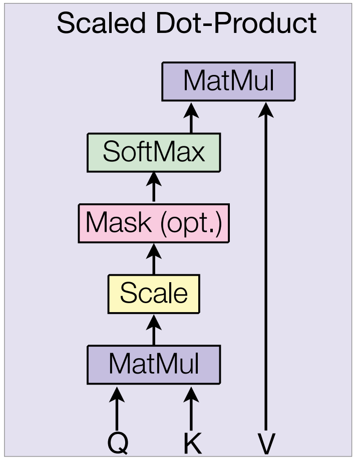
Transformer Multi-head Attention

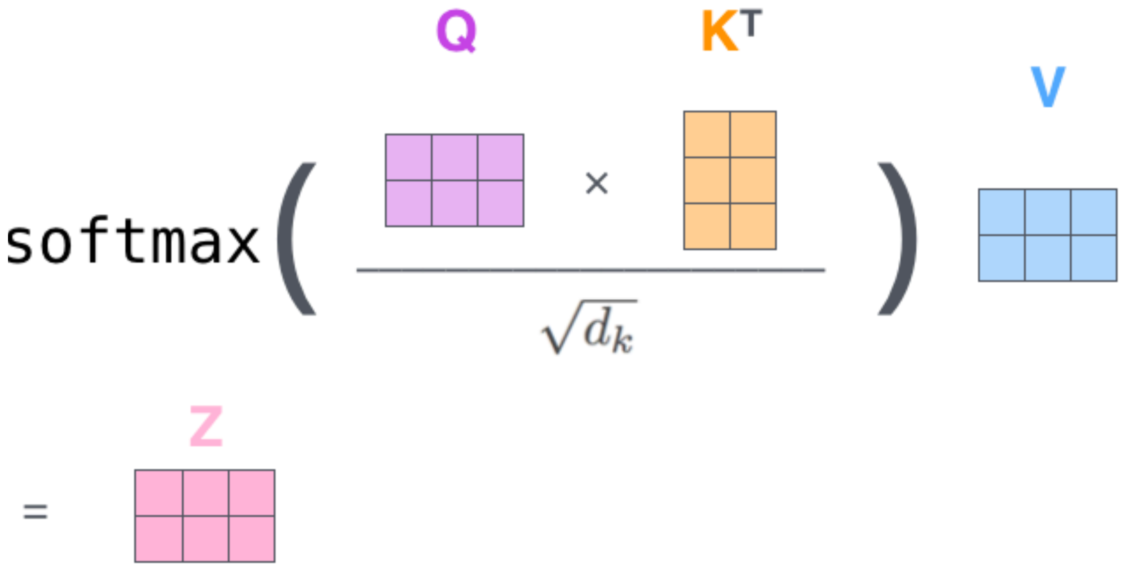
- C layers of encoder (=6)
- D layers of decoder (=6)



Scaled Dot-Product Attention

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d}})V$$



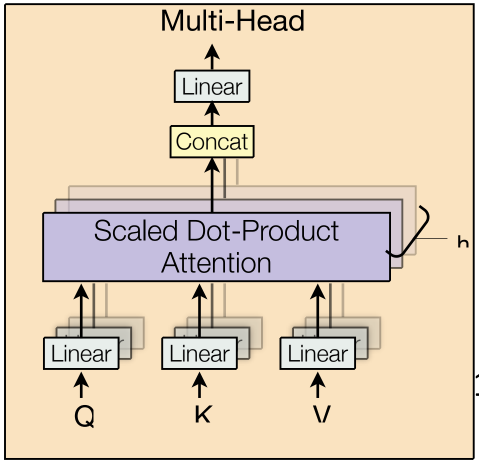


Multi-head Attention

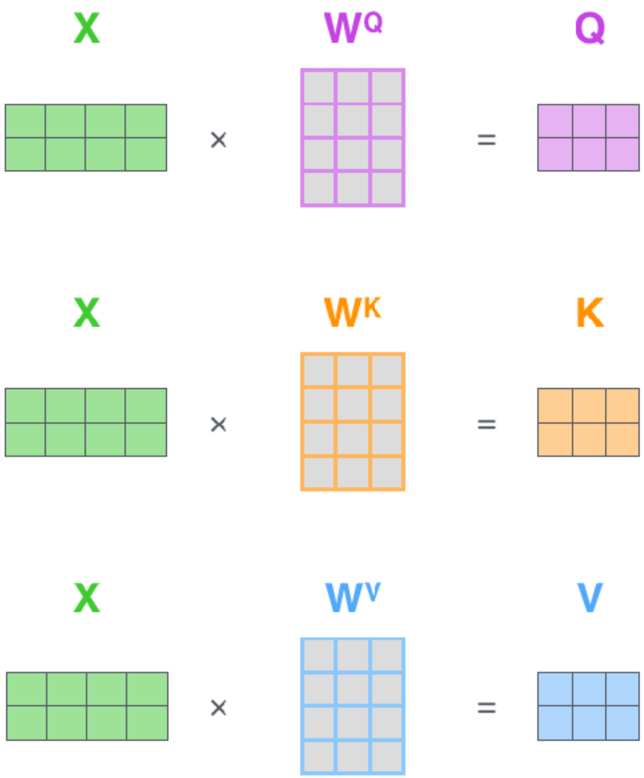
- Instead of one vector for each token
- break into multiple heads
- each head perform attention

$$\text{Head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

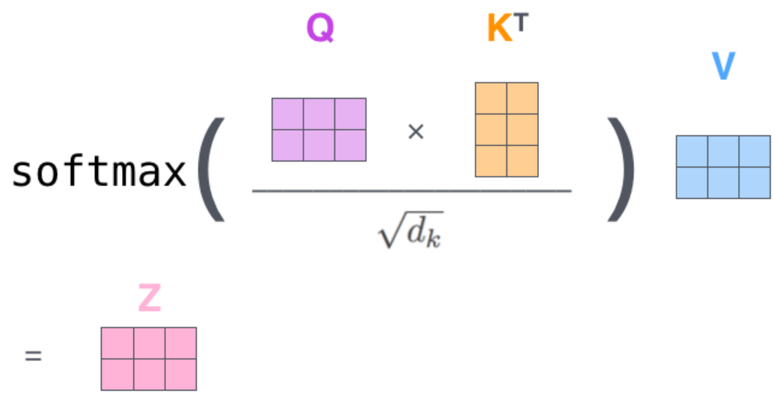
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{Head}_1, \text{Head}_2, \ldots, \text{Head}_h)W^o$$



Multi-head Attention



sent len x sent len

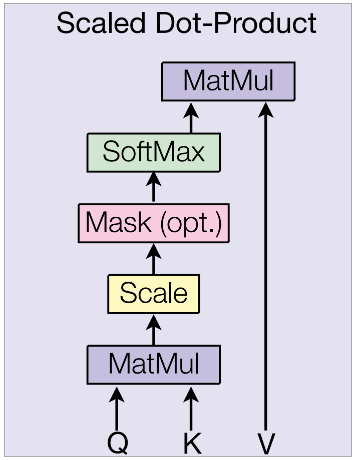


sent len x dim

Alammar, The Illustrated Transformer

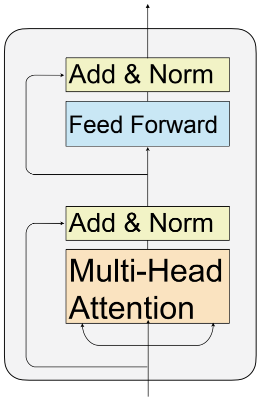
Self-Attention for Decoder

- Maskout right side before softmax (-inf)



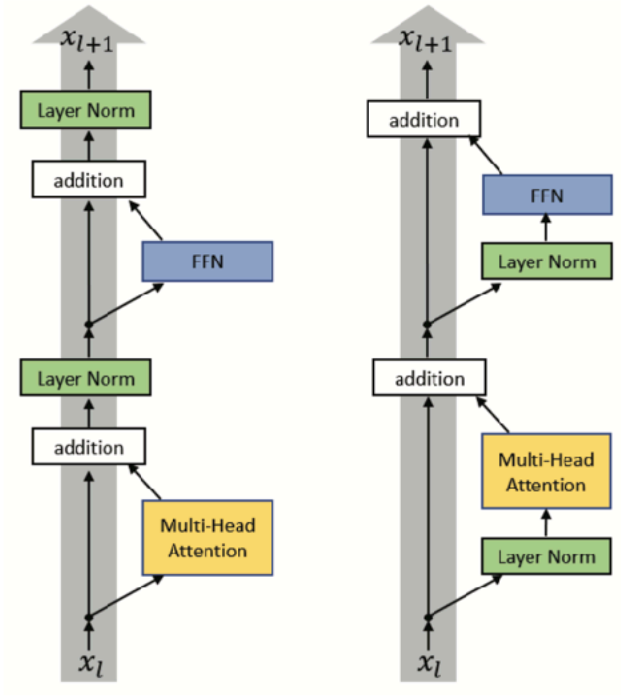
Feedforward Net

- $\mathrm{FFN}(x) = \max(0, x \cdot W_1 + b_1) \cdot W_2 + b_2$
- internal dimension size = 2048 (in Vaswani 2017)



Residual Connection and Layer Normalization

- Residual Connection
- Make it zero mean and unit variance within layer
- Post-norm
- Pre-norm

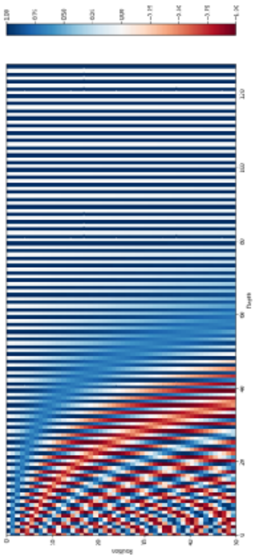


Embedding

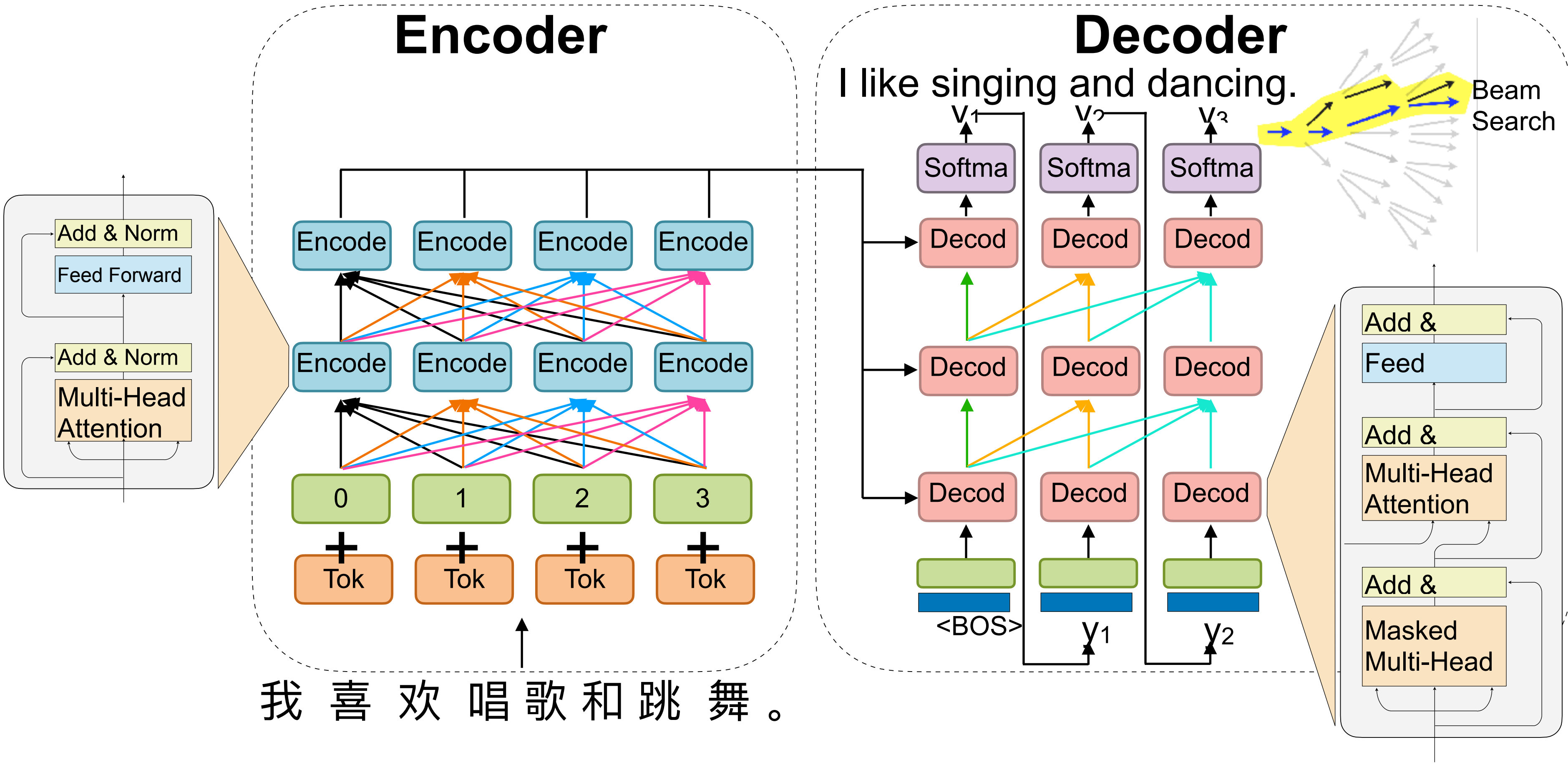
- Token Embedding: 512 (base), 1024 (large)
  - Shared (tied) input and output embedding
- Positional Embedding:
  - to distinguish words in different position, Map position labels to embedding, dimension is same as Tok Emb

$$PE_{pos,2i} = \sin\left(\frac{pos}{1000^{2i/d}}\right)$$

$$PE_{pos,2i+1} = \cos\left(\frac{pos}{1000^{2i/d}}\right)$$



Transformer



Vaswani et al. Attention is All You Need. 2017

Training Loss

$$P(Y|X) = \prod P(y_t | y_{<t}, x)$$

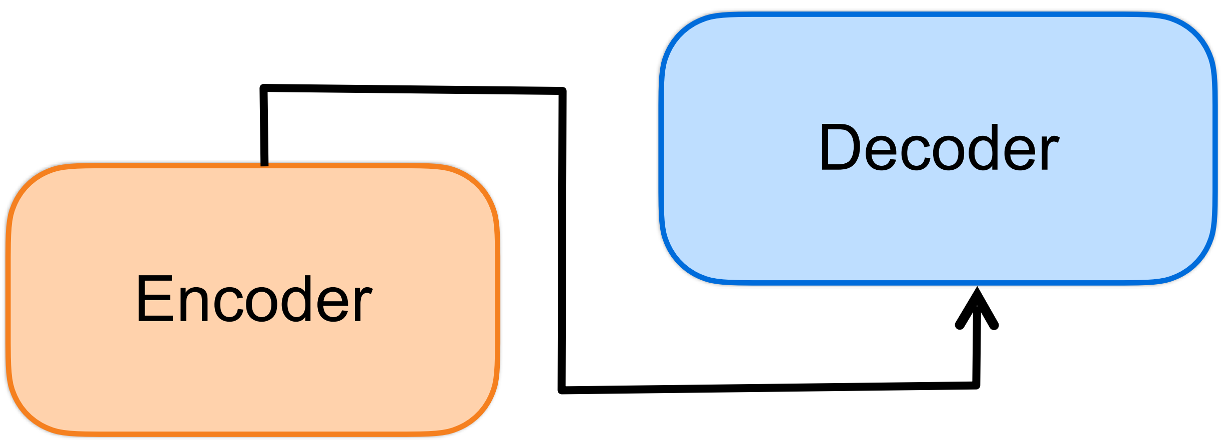
Training loss: Cross-Entropy

$$l = -\sum_n \sum_t \log f_\theta(x_n, y_{n,1}, \ldots, y_{n,t-1})$$

Teacher-forcing during training.

(pretend to know groundtruth for prefix)

target:
I like singing and dancing.



Source: 我喜欢唱歌和跳舞。

Training

- Dropout
  - Applied to before residual
  - and to embedding, pos emb.
  - p=0.1 ~ 0.3
- Label smoothing
  - 0.1 probability assigned to non-truth
- Vocabulary:
  - En-De: 37K using BPE
  - En-Fr: 32k word-piece (similar to BPE)

Label Smoothing

- Assume $\mathbf{y} \in \mathbb{R}^n$ is the one-hot encoding of label

$$y_i = \begin{cases} 1 & \text{if belongs to class } i \\ 0 & \text{otherwise} \end{cases}$$

- Approximating 0/1 values with softmax is hard
- The smoothed version

$$y_i = \begin{cases} 1 - \epsilon & \text{if belongs to class } i \\ \epsilon/(n-1) & \text{otherwise} \end{cases}$$

  – Commonly use $\epsilon = 0.1$

Training

- Batch
  - group by approximate sentence length
  - still need shuffling
- Hardware
  - one machine with 8 GPUs (in 2017 paper)
  - base model: 100k steps (12 hours)
  - large model: 300k steps (3.5 days)
- Adam Optimizer
  - increase learning rate during warmup, then decrease

$$\eta = \frac{1}{\sqrt{d}} \min\left(\frac{1}{\sqrt{t}}, \frac{t}{\sqrt{t_0^3}}\right)$$

ADAM

$$m_{t+1} = \beta_1 m_t - (1 - \beta_1) \nabla \ell(x_t)$$
$$v_{t+1} = \beta_2 v_t + (1 - \beta_2)(\nabla \ell(x_t))^2$$
$$\hat{m}_{t+1} = \frac{m_{t+1}}{1 - \beta_1^{t+1}}$$
$$\hat{v}_{t+1} = \frac{v_{t+1}}{1 - \beta_2^{t+1}}$$
$$x_{t+1} = x_t - \frac{\eta}{\sqrt{\hat{v}_{t+1}} + \epsilon} \hat{m}_{t+1}$$

Model Average

- A single model obtained by averaging the last 5 checkpoints, which were written at 10-minute interval (base)
- decoding length: within source length + 50

Machine Translation

Many possible translation, which is better?

SpaceX周三晚间进行了一次发射任务，将四名毫无航天经验的业余人士送入太空轨道。

SpaceX launched a mission Wednesday night to put four amateurs with no space experience into orbit.

SpaceX conducted a launch mission on Wednesday night, sending four amateurs with no aerospace experience into space orbit.

SpaceX conducted a launch mission Wednesday night that sent four amateurs with no spaceflight experience into orbit.

SpaceX carried out a launch mission on Wednesday night to put four amateurs without Aerospace experience into orbit.

BLEU

- Measuring the precision of n-grams
  - Precision of n-gram: percentage of tokens in output sentences

$$\_\, p_n = \frac{num\,.\,of\,.\,correct\,.\,token\,.\,ngram}{total\,.\,output\,.\,ngram}$$

- Penalize for brevity
  - if output is too short
  - $bp = min(1, e^{1-r/c})$
- BLEU=$bp \cdot (\prod p_i)^{\frac{1}{4}}$
- Notice BLEU is computed over the whole corpus, not on one sentence

Example

Ref: A SpaceX rocket was launched into a space orbit Wednesday evening.

System A: SpaceX launched a mission Wednesday evening into a space orbit.

System B: A rocket sent SpaceX into orbit Wednesday.

Example

Ref: A SpaceX rocket was launched into a space orbit Wednesday evening.

System A: SpaceX launched a mission Wednesday evening into a space orbit.

	Precision
Unigram	9/11
Bigram	4/10
Trigram	2/9
Four-gram	1/8

$$bp=e^{1-12/11}=0.91$$

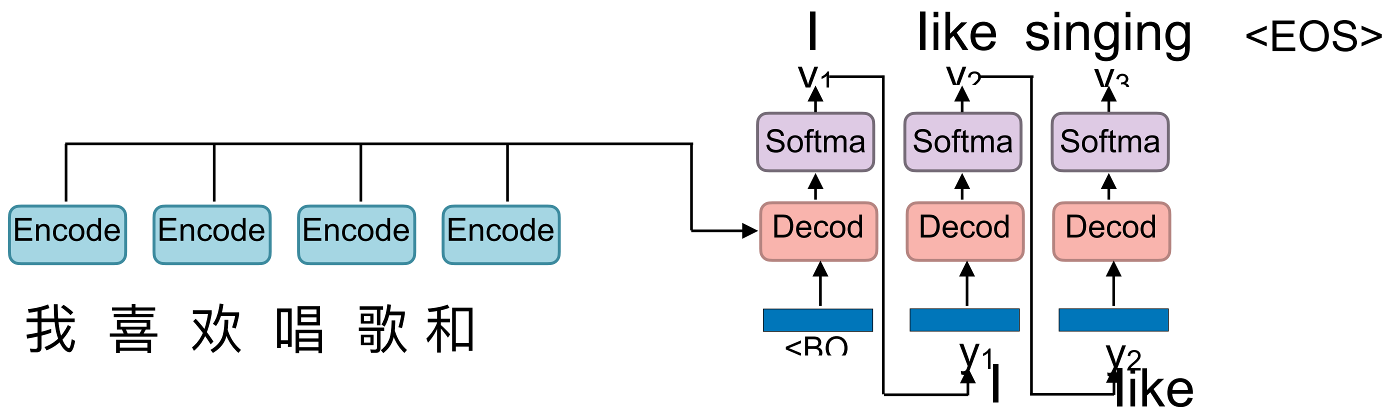
$$BLEU=0.91*(9/11 * 4/10 * 2/9 * 1/8)^{1/4}$$

$$=28.1\%$$

Sequence Decoding

Autoregressive Generation

greedy decoding: output the token with max next token prob



But, this is not necessary the best

Inference

- Now already trained a model θ
- Decoding/Generation: Given an input sentence x, to generate the target sentence y that maximize the probability $P(y \mid x; \theta)$
- $\underset{y}{\operatorname{argmax}} P(y \mid x) = f_\theta(x, y)$
- Two types of error
  - the most probable translation is bad → fix the model
  - search does not find the most probably translation → fix the search
- Most probable translation is not necessary the highest BLEU one!

Decoding

- $\underset{y}{\operatorname{argmax}} P(y \mid x) = f_\theta(x, y)$
- naive solution: exhaustive search
  - too expensive
- Beam search
  - (approximate) dynamic programming

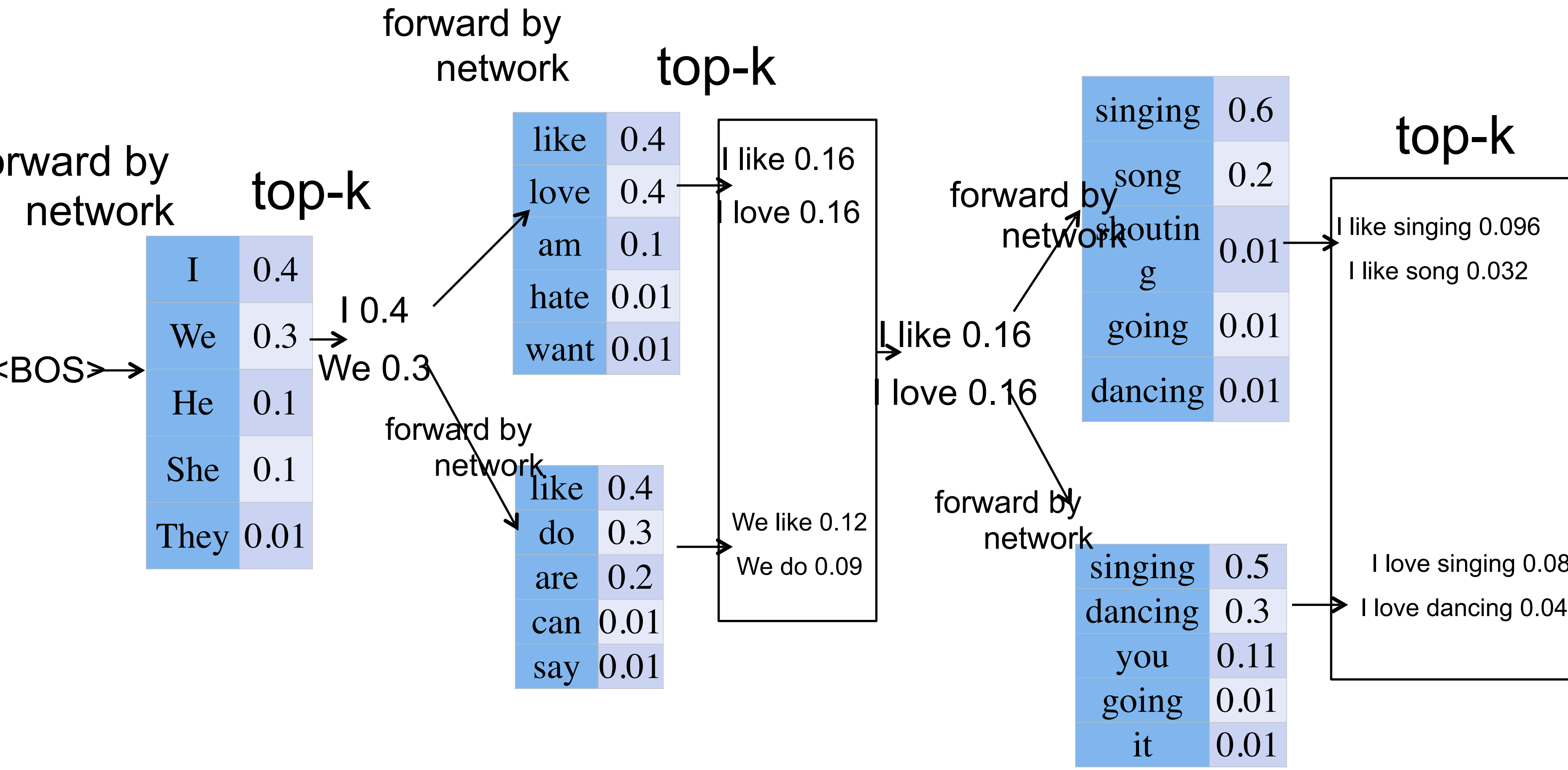
Beam Search

- start with empty S
- at each step, keep k best partial sequences
- expand them with one more forward generation
- collect new partial results and keep top-k

Beam Search (pseudocode)

```
best_scores = []
add {[0], 0.0} to best_scores # 0 is for beginning of sentence token
for i in 1 to max_length:
  new_seqs = PriorityQueue()
  for (candidate, s) in best_scores:
    if candidate[-1] is EOS:
        prob = all -inf
        prob[EOS] = 0
      else:
      prob = using model to take candidate and compute next token
probabilities (logp)
    pick top k scores from prob, and their index
    for each score, index in the top-k of prob:
      new_candidate = candidate.append(index)
      new_score = s + score
      if not new_seqs.full():
```

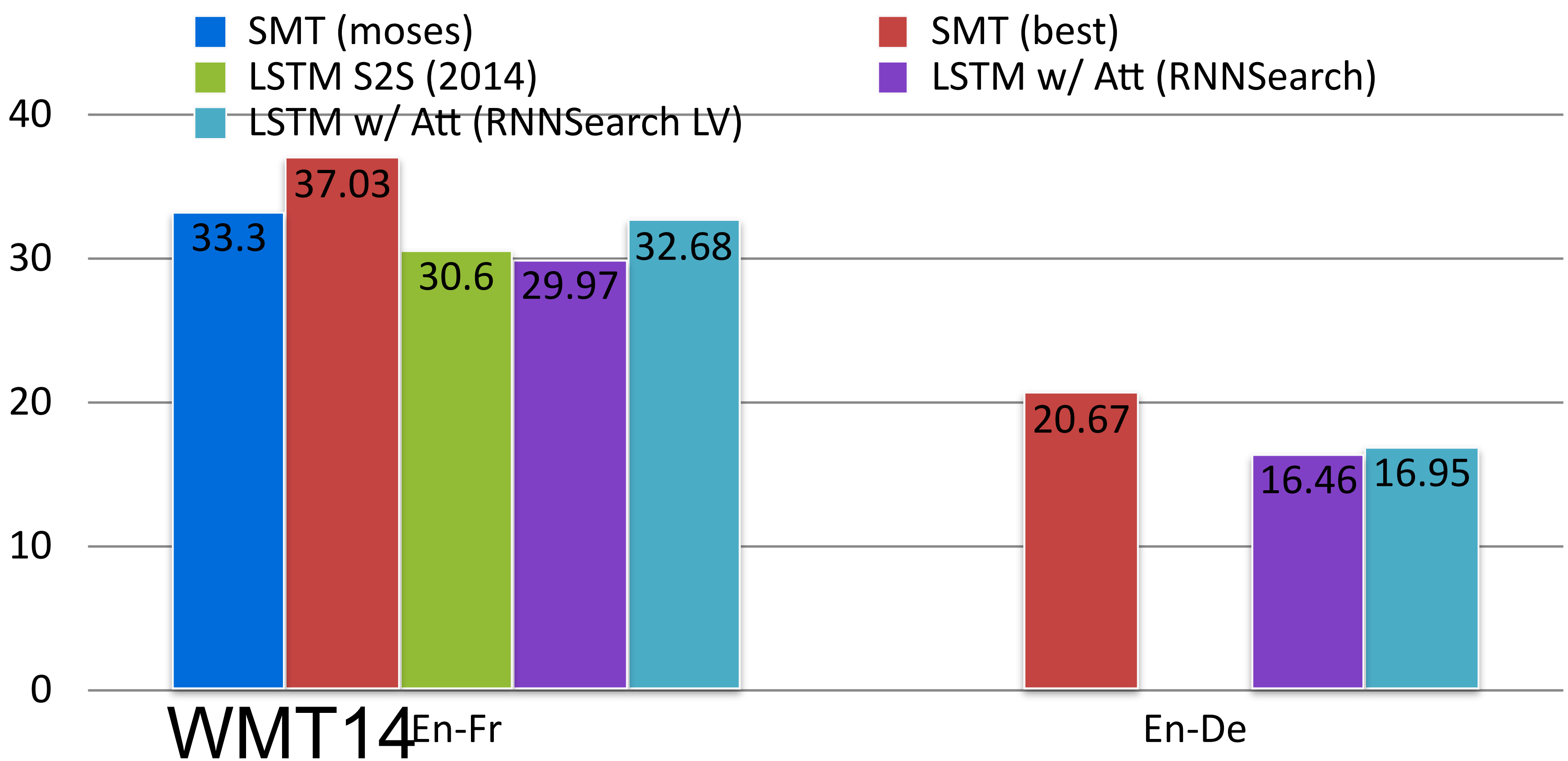
Beam Search



forward by network
top-k
<BOS>
I 0.4
We 0.3
He 0.1
She 0.1
They 0.01
I 0.4
We 0.3
forward by network
top-k
like 0.4
love 0.4
am 0.1
hate 0.01
want 0.01
forward by network
like 0.4
do 0.3
are 0.2
can 0.01
say 0.01
I like 0.16
I love 0.16
We like 0.12
We do 0.09
I like 0.16
I love 0.16
forward by network
singing 0.6
song 0.2
shouting 0.01
going 0.01
dancing 0.01
forward by network
singing 0.5
dancing 0.3
you 0.11
going 0.01
it 0.01
top-k
I like singing 0.096
I like song 0.032
I love singing 0.08
I love dancing 0.048

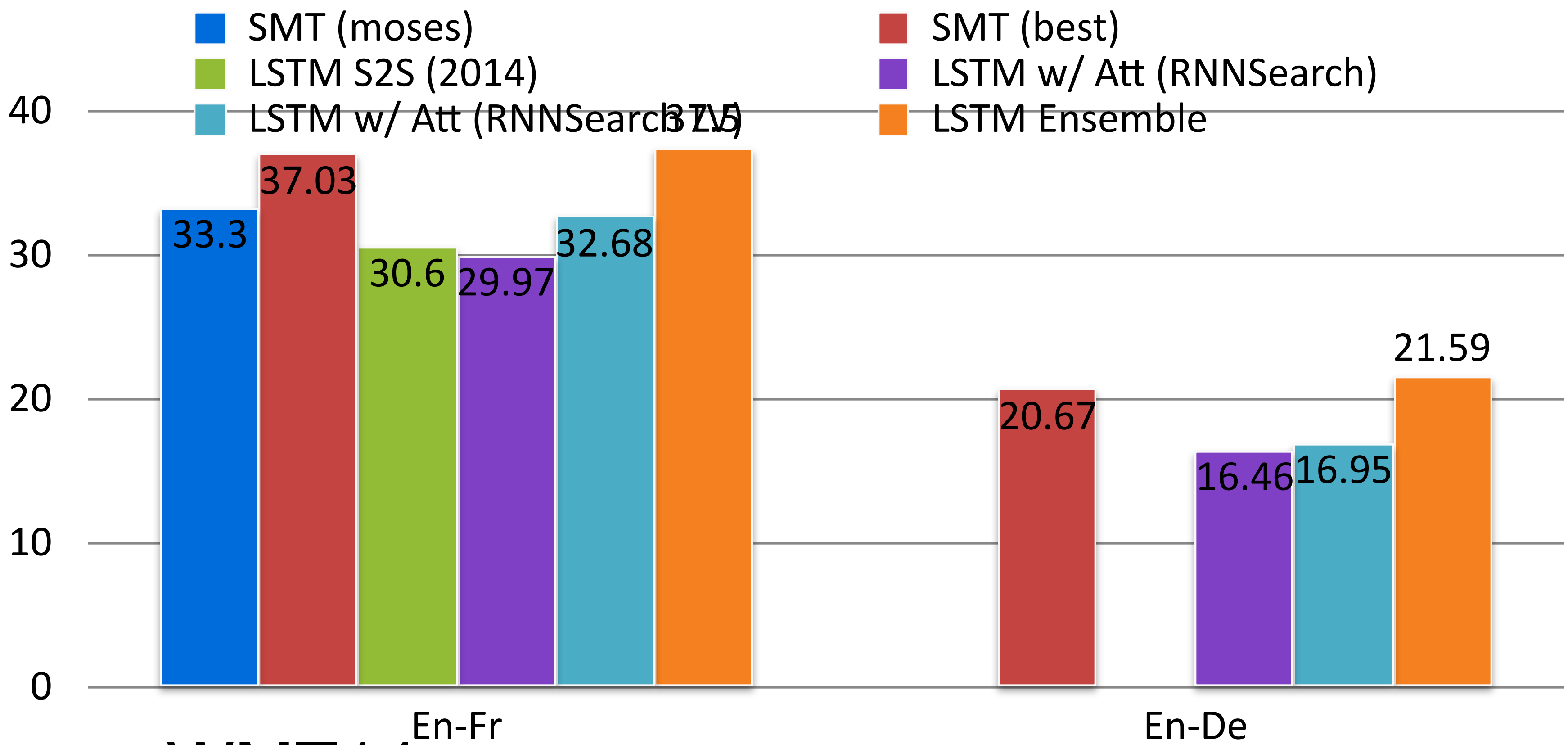
Machine Translation using Seq2seq and Transformer

LSTM Seq2Seq w/ Attention



Jean et al. On Using Very Large Target Vocabulary for Neural Machine Translation. 2015

Performance with Model Ensemble



WMT14

Luong et al. Effective Approaches to Attention-based Neural Machine Translation. 2015

Results on WMT14

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Effectiveness of Choices

- num. head
- dim of key
- num layers
- hid dim
- ffn dim
- dropout
- pos emb

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

Deep Transformer

- 30 ~ 60 encoder
- 12 decoder
- dynamic linear combination of layers (DLCL)
  - or. deeply supervised
  - combine output from all layers

Wang et al. Learning Deep Transformer Models for Machine Translation, 2019.

Model		Param.	Batch (×4096)	Updates (×100k)	†Times	BLEU	Δ
Vaswani et al. (2017) (Base)		65M	1	1	reference	27.3	-
Bapna et al. (2018)-deep (Base, 16L)		137M	-	-	-	28.0	-
Vaswani et al. (2017) (Big)		213M	1	3	3x	28.4	-
Chen et al. (2018a) (Big)		379M	16	†0.075	1.2x	28.5	-
He et al. (2018) (Big)		†210M	1	-	-	29.0	-
Shaw et al. (2018) (Big)		†210M	1	3	3x	29.2	-
Dou et al. (2018) (Big)		356M	1	-	-	29.2	-
Ott et al. (2018) (Big)		210M	14	0.25	3.5x	29.3	-
post-norm	Transformer (Base)	62M	1	1	1x	27.5	reference
	Transformer (Big)	211M	1	3	3x	28.8	+1.3
	Transformer-deep (Base, 20L)	106M	2	0.5	1x	failed	failed
	DLCL (Base)	62M	1	1	1x	27.6	+0.1
	DLCL-deep (Base, 25L)	121M	2	0.5	1x	29.2	+1.7
pre-norm	Transformer (Base)	62M	1	1	1x	27.1	reference
	Transformer (Big)	211M	1	3	3x	28.7	+1.6
	Transformer-deep (Base, 20L)	106M	2	0.5	1x	28.9	+1.8
	DLCL (Base)	62M	1	1	1x	27.3	+0.2
	DLCL-deep (Base, 30L)	137M	2	0.5	1x	**29.3**	**+2.2**

Wang et al. Learning Deep Transformer Models for Machine Translation, 2019.

Model	Param.	newstest17	newstest18	$\Delta_{avg.}$
Wang et al. (2018a) (post-norm, Base)	102.1M	25.9	-	-
pre-norm Transformer (Base)	102.1M	25.8	25.9	reference
pre-norm Transformer (Big)	292.4M	26.4	27.0	+0.9
pre-norm DLCL-deep (Base, 25L)	161.5M	26.7	27.1	+1.0
pre-norm DLCL-deep (Base, 30L)	177.2M	**26.9**	**27.4**	**+1.3**

Table 4: BLEU scores [%] on WMT'18 Chinese-English translation.

Wang et al. Learning Deep Transformer Models for Machine Translation, 2019.

Hot Topics in MT

- Parallel Decoding (e.g. NAT, GLAT, DAT,…)
- Low-resource MT
- Unsupervised MT
- Multilingual NMT, Zero-shot NMT
- Speech-to-text translation
  - (Offline) ST
  - Streaming ST

Pre-training Language Models

Contextual Representations

- Problem: Word embeddings are applied in a context free manner

open a bank account   on the river bank

[0.3, 0.2, -0.8, …]

- Solution: Train contextual representations on text corpus

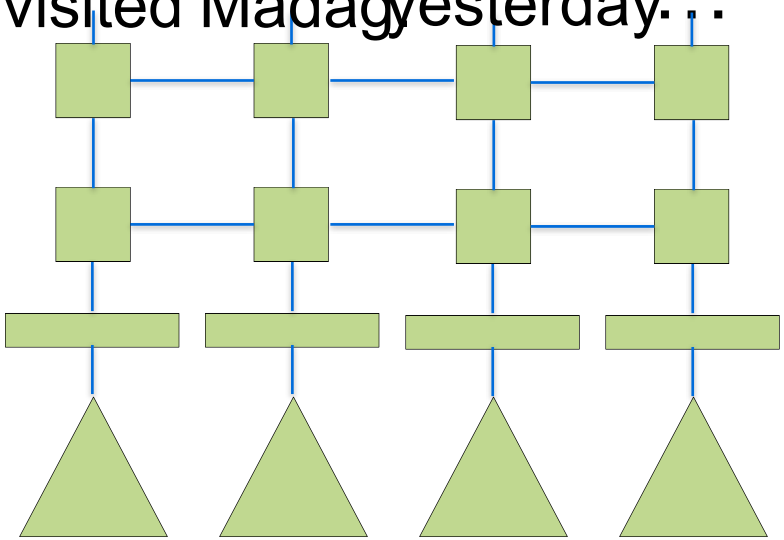
[0.9, -0.2, 1.6, …]   [-1.9, -0.4, 0.1, …]

open a bank account   on the river bank

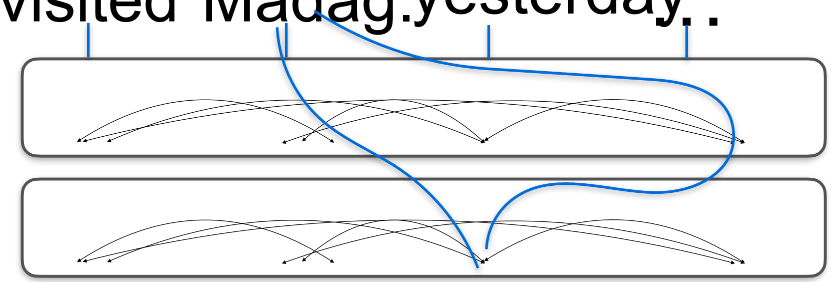
Bidirectional Context

- How to learn a “deeply bidirectional” model? What happens if we just replace an LSTM with a transformer?

ELMo (Language Modeling)



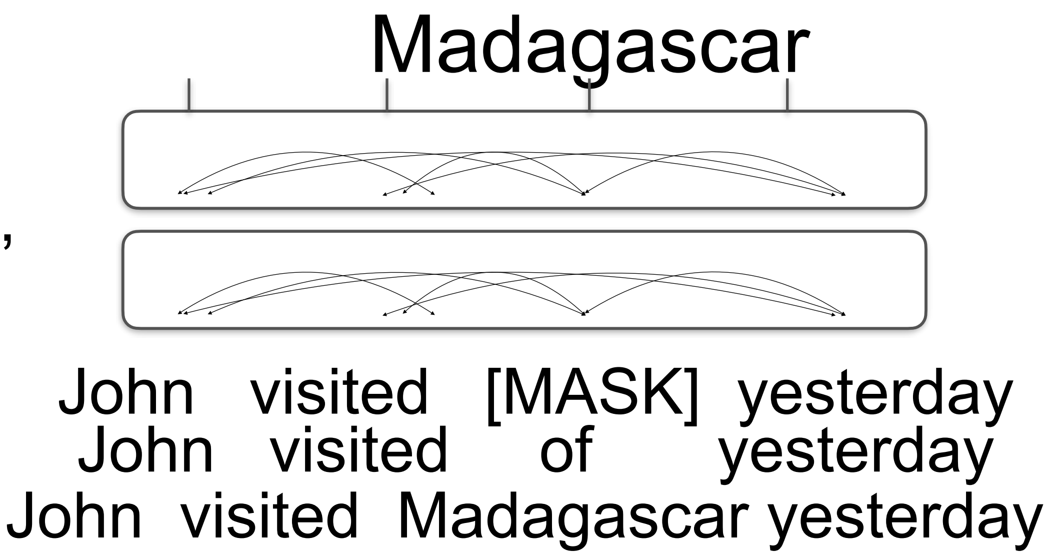
BERT



Transformer LMs have to be “one-sided” (only attend to previous tokens), not what we want

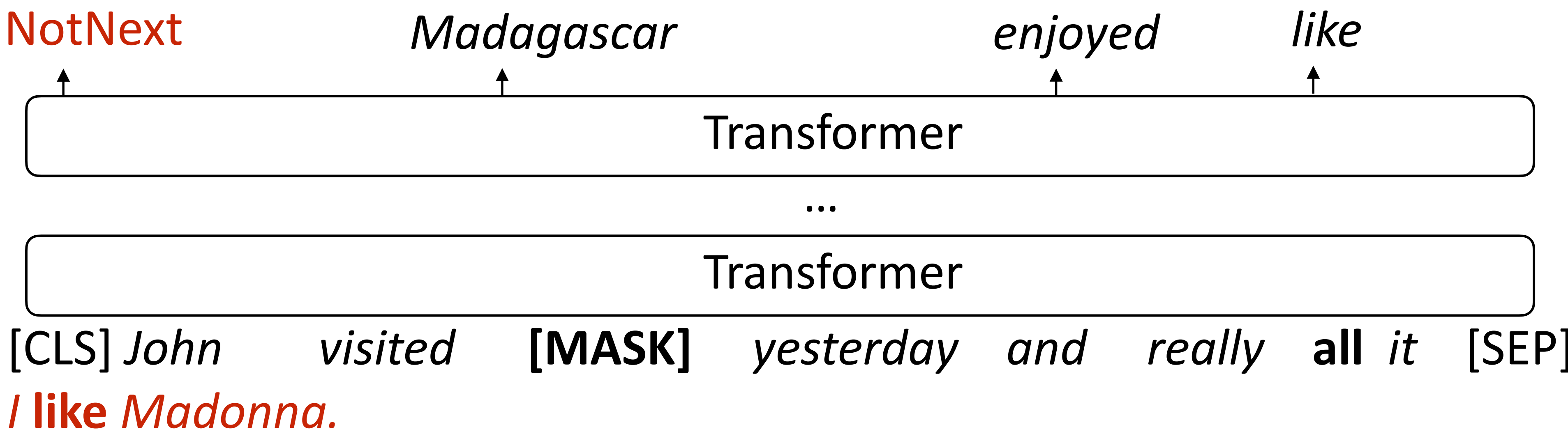
Masked Language Modeling

- How to prevent cheating? Next word prediction fundamentally doesn't work for bidirectional models, instead do masked language modeling
- BERT formula: take a chunk of text, predict 15% of the tokens
  - For 80% (of the 15%), replace the input token with [MASK]
  - For 10%, replace w/ random
  - For 10%, keep same (why?)



Next "Sentence" Prediction

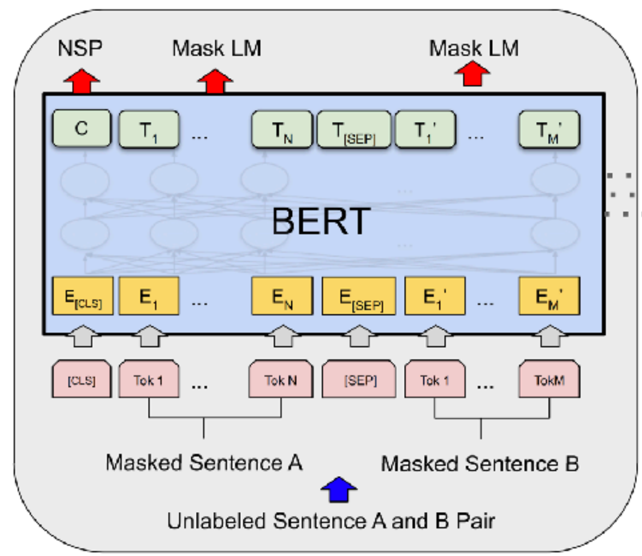
- Input: [CLS] Text chunk 1 [SEP] Text chunk 2
- 50% of the time, take the true next chunk of text, 50% of the time take a random other chunk. Predict whether the next chunk is the "true" next
- BERT objective: masked LM (CE) + next sentence prediction

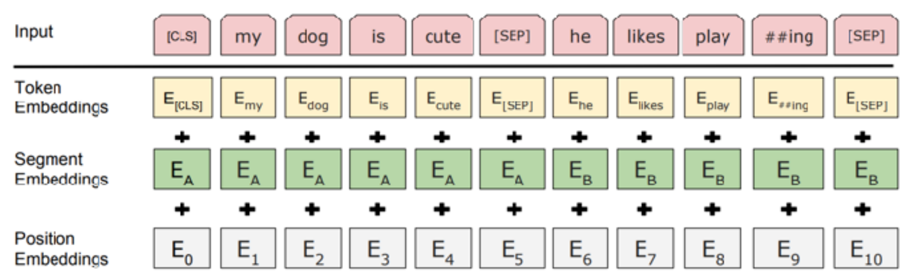


Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019

BERT Architecture

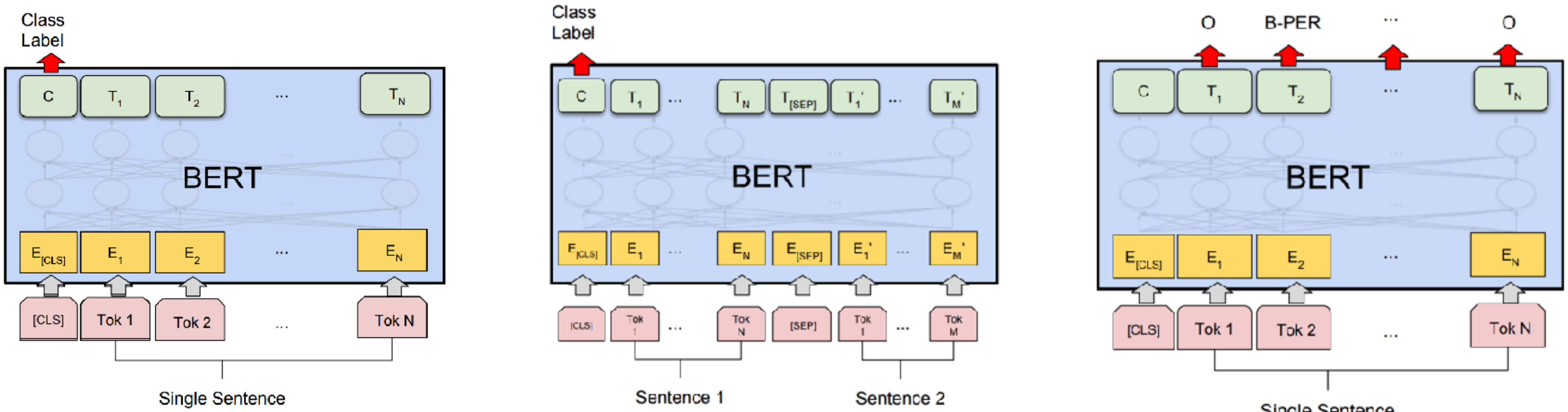
- BERT Base: 12 Transformer encoder layers, 768-dim, 12 heads. Total params = 110M
- BERT Large: 24 layers, 1024-dim, 16 heads. Total params = 340M
- Vocabulary: 30k wordpiece
- Positional embeddings and segment embeddings
- Data: Wikipedia (2.5B words + BookCorpus (800M words)





Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019

Unified model across NLP Tasks



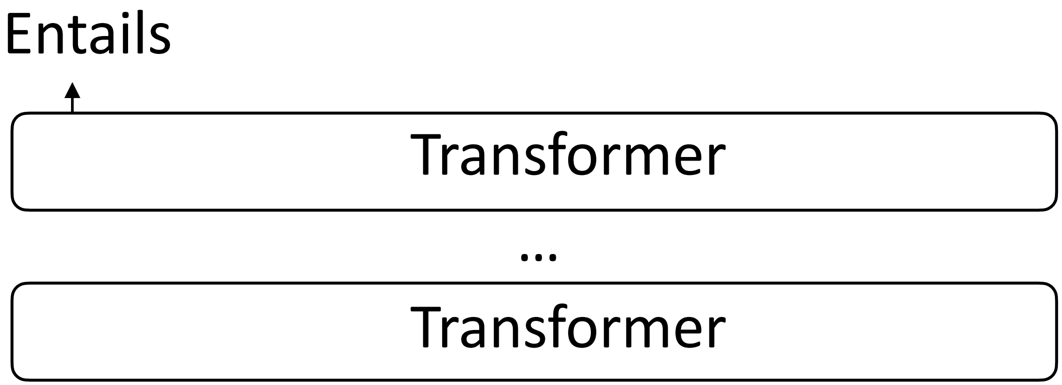
(b) Single Sentence Classification Tasks: SST-2, CoLA

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

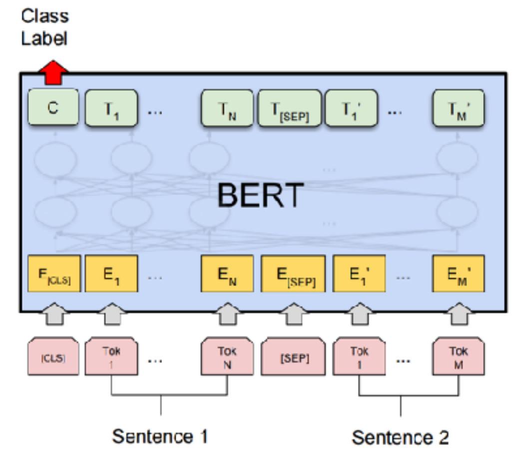
(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

- CLS token is used to provide classification decisions
- Sentence pair tasks (entailment): feed both sentences into BERT
- BERT can also do tagging by predicting tags at each word piece

What can BERT do?



[CLS] A boy plays in the snow [SEP] A boy is outside



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

- How does BERT model this sentence pair stuff?
- Transformers can capture interactions between the two sentences, even though the NSP objective doesn't really cause this to happen

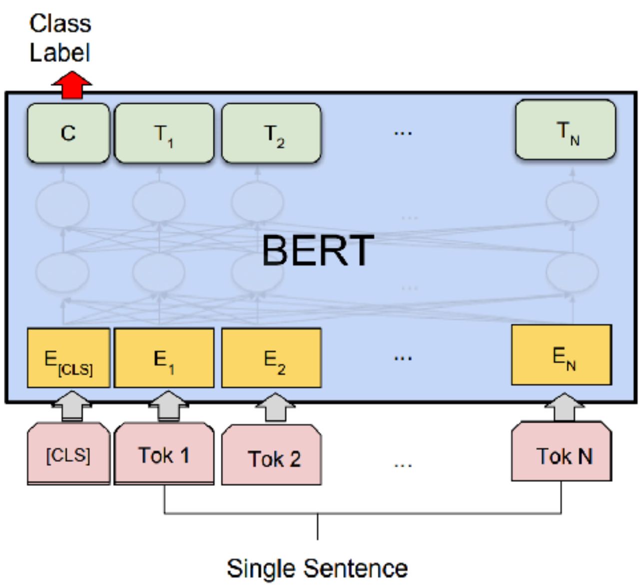
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019

What can BERT NOT do?

- Does not give sentence probability
- BERT cannot generate text (at least not in an obvious way)
  - Not an autoregressive model, can do weird things like stick a [MASK] at the end of a string, fill in the mask, and repeat
- Masked language models are intended to be used primarily for “understanding/ analysis” tasks (NLU)

Fine-tuning BERT

- Fine-tune for 1-3 epochs, batch size 2-32, learning rate 2e-5 - 5e-5



(b) Single Sentence Classification Tasks: SST-2, CoLA

- Large changes to weights up here (particularly in last layer to route the right information to [CLS])
- Smaller changes to weights lower down in the transformer
- Small LR and short fine-tuning schedule mean weights don't change much
- More complex "triangular learning rate" schemes exist

Fine-tuning BERT

Pretraining	Adaptation	NER	SA	Nat. lang. inference		Semantic textual similarity		
		CoNLL 2003	SST-2	MNLI	SICK-E	SICK-R	MRPC	STS-B
Skip-thoughts	❄️	-	81.8	62.9	-	86.6	75.8	71.8
ELMo	❄️	91.7	**91.8**	**79.6**	**86.3**	**86.1**	**76.0**	**75.9**
	🔥	**91.9**	91.2	76.4	83.3	83.3	74.7	75.5
	Δ=🔥-❄️	0.2	-0.6	-3.2	-3.3	-2.8	-1.3	-0.4
BERT-base	❄️	92.2	93.0	**84.6**	84.8	86.4	78.1	82.9
	🔥	**92.4**	**93.5**	**84.6**	**85.8**	**88.7**	**84.8**	**87.1**
	Δ=🔥-❄️	0.2	0.5	0.0	1.0	2.3	6.7	4.2

- BERT is typically better if the whole network is fine-tuned, unlike ELMo

Peters, Ruder, Smith. To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks (2019)

Evaluation: GLUE

Corpus	|Train|	|Test|	Task	Metrics	Domain
			Single-Sentence Tasks		
CoLA	8.5k	**1k**	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
			Similarity and Paraphrase Tasks		
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	**391k**	paraphrase	acc./F1	social QA questions
			Inference Tasks		
MNLI	393k	**20k**	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	**146**	coreference/NLI	acc.	fiction books

Wang et al. GLUE. 2019

Results

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

- Huge improvements over prior work (even compared to ELMo)
- Effective at “sentence pair” tasks: textual entailment (does sentence A imply sentence B), paraphrase detection

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019

Improving BERT

- Dynamic masking: standard BERT uses the same MASK scheme for every epoch, RoBERTa recomputes them

epoch 2

epoch 1

… John visited Madagascar yesterday …

- Whole word masking: don't mask out parts of words

… _John _visited _Mada gas car yesterday …

Liu et al. (2019)

RoBERTa

- "Robustly optimized BERT" incorporating some of these tricks
- 160GB of data instead of 16 GB

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7

- New training + more data = better performance

Liu et al. (2019)

BERT/MLMs

- There are lots of ways to train these models!
- Key factors:
  - Big enough model
  - Big enough data
  - Well-designed "self-supervised" objective (something like language modeling). Needs to be a hard enough problem!

Other Pre-trained LM

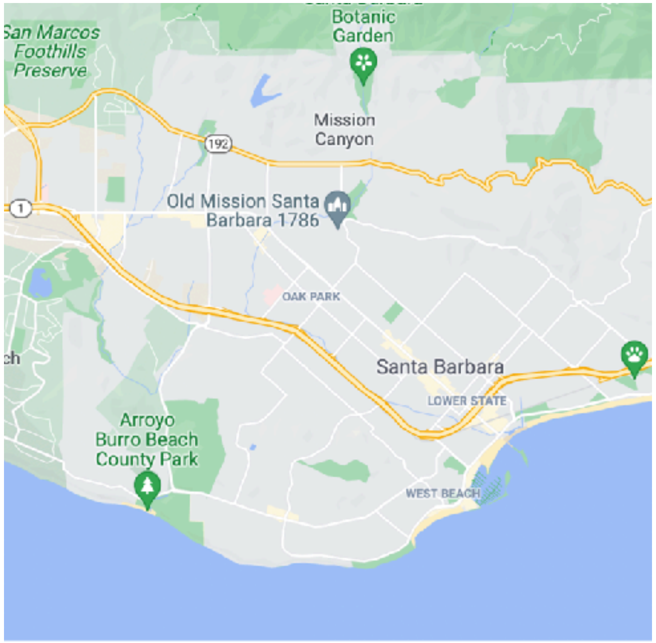
- GPT (GPT-2, GPT-3)
  - Transformer decoder only
- T5
  - Transformer encoder-decoder
  - with many tasks
- BART
  - Transformer encoder-decoder, with denoising training

Sequence Labelling

Understanding Query Intention

Noodle house near Santa Barbara
[Keyword] [Location]

How to go from Santa Barbara to Log Angeles ?
[Origin] [Destination]



Sequence Labelling

Named entity recognition

date   Location
In **April 1775** fighting broke out between **Massachusetts**
militia units and **British** regulars at **Lexington** and **Concord** .
                    Geo-Political

Sequence Labelling

- Named entity recognition

  In **April 1775** fighting broke out between **Massachusetts** militia units and **British** regulars at **Lexington** and **Concord** .

- Semantic role labeling

The excess supply pushed gasoline prices down in that period .
subject verb object

- Question Answering: subject parsing

  Who created Harry Potter ?

Represent the Output Labels

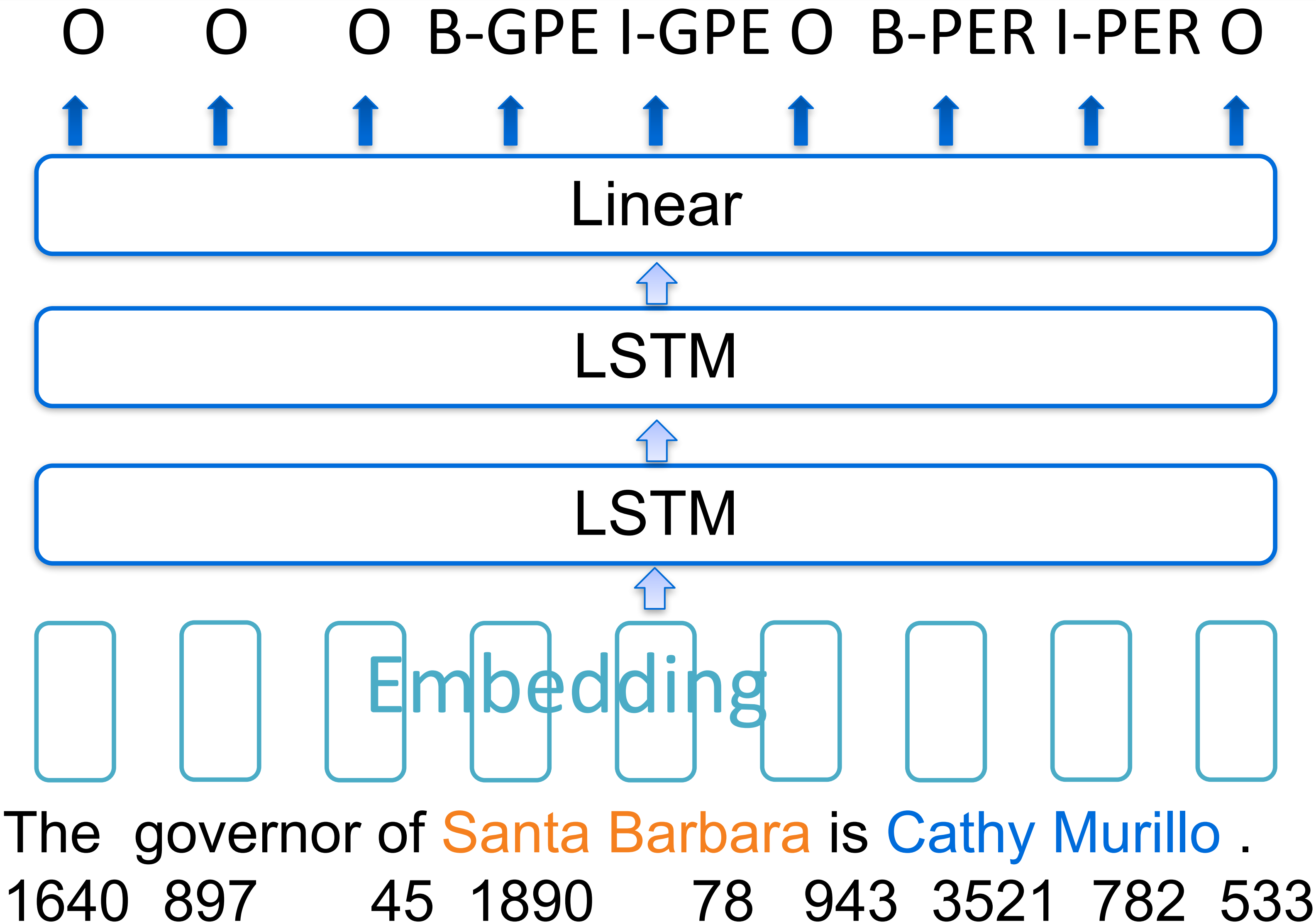
- BIO scheme

O O O B-GPE I-GPE O B-PER I-PER O

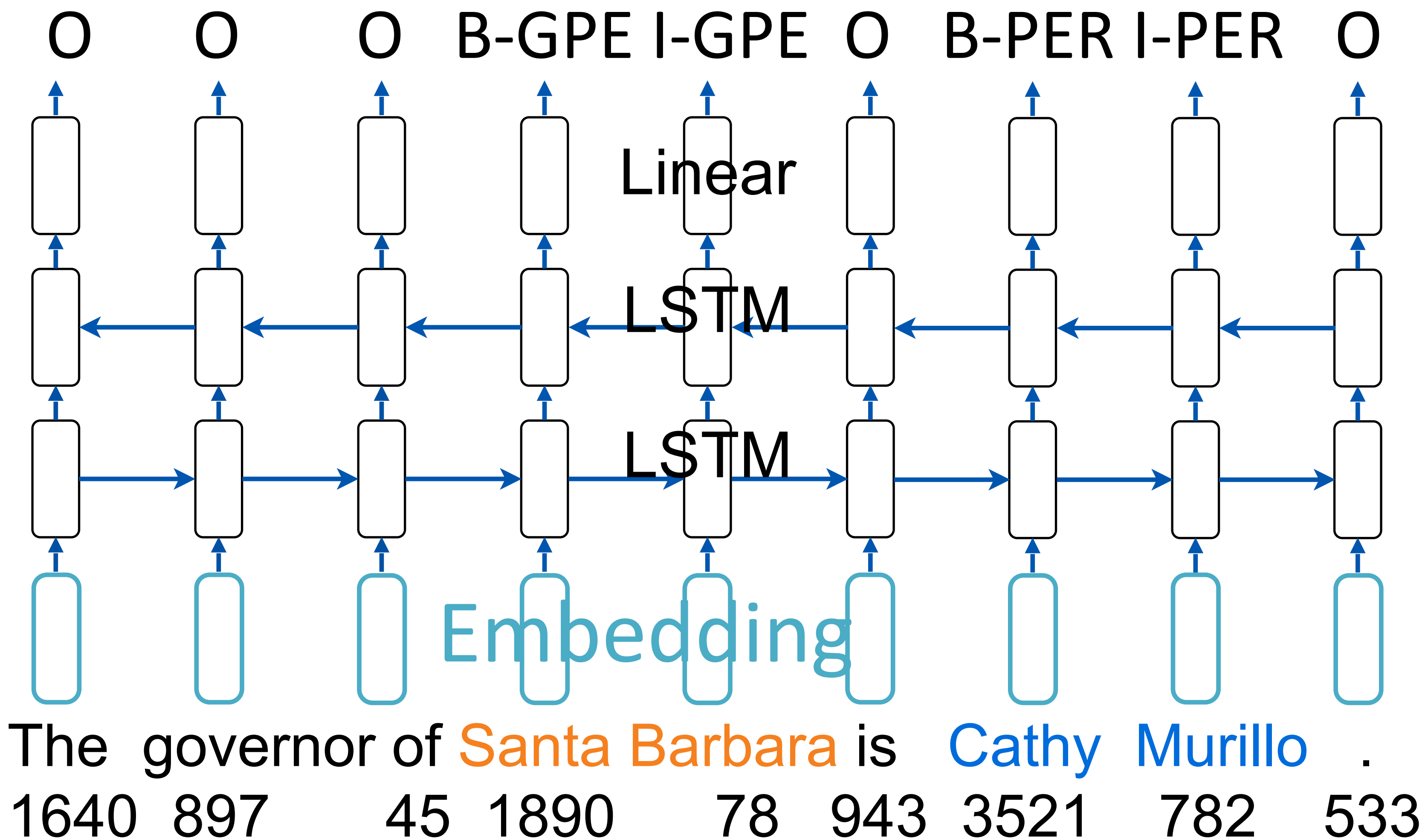
The governor of Santa Barbara is Cathy Murillo .

1640 897 45 1890 78 943 3521 782 533

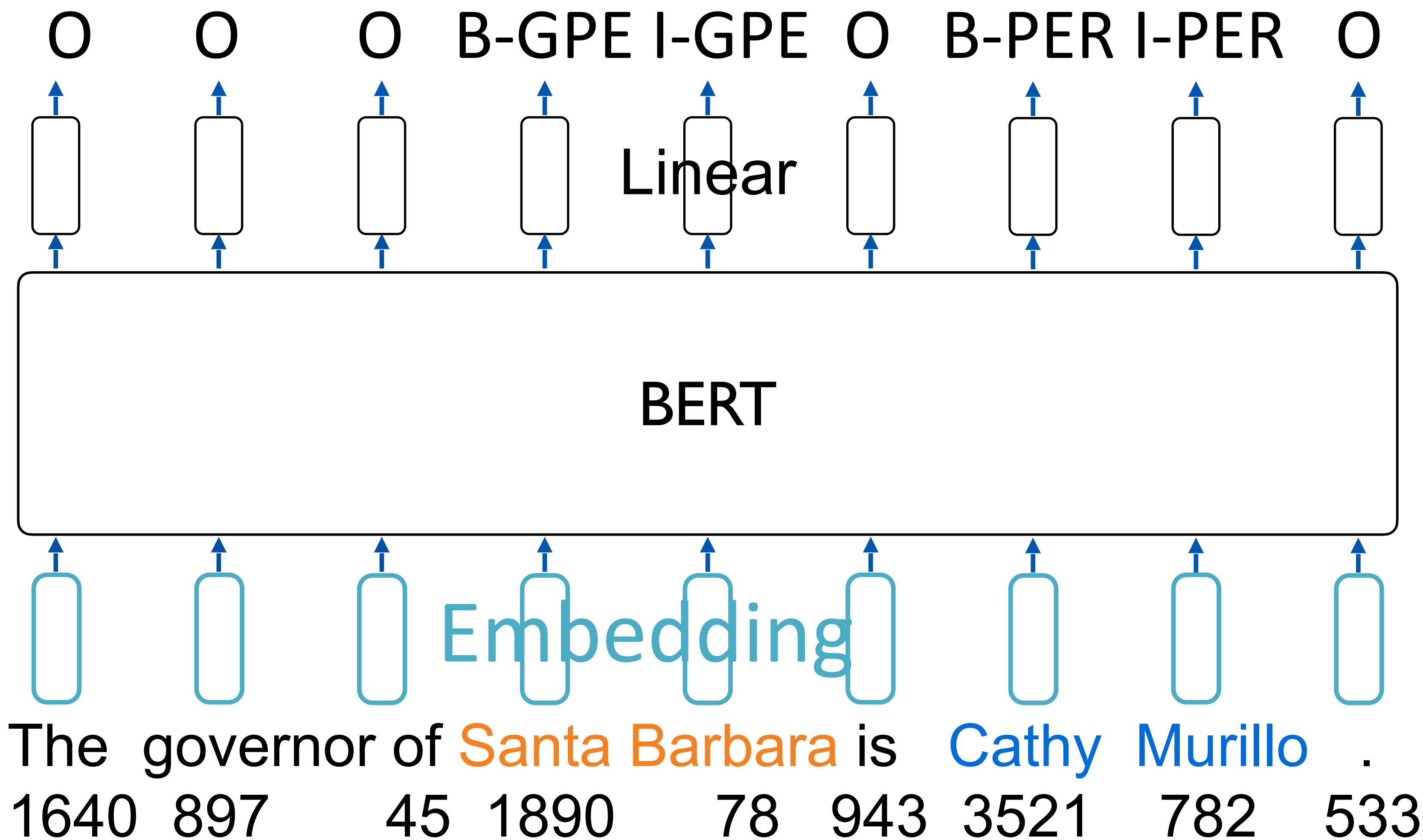
RNN/LSTM for Sequence Labelling



Bi-LSTM


O O O B-GPE I-GPE O B-PER I-PER O
Linear
LSTM
LSTM
Embedding
The governor of Santa Barbara is Cathy Murillo .
1640 897 45 1890 78 943 3521 782 533

BERT for Seq-Labeling



Summary

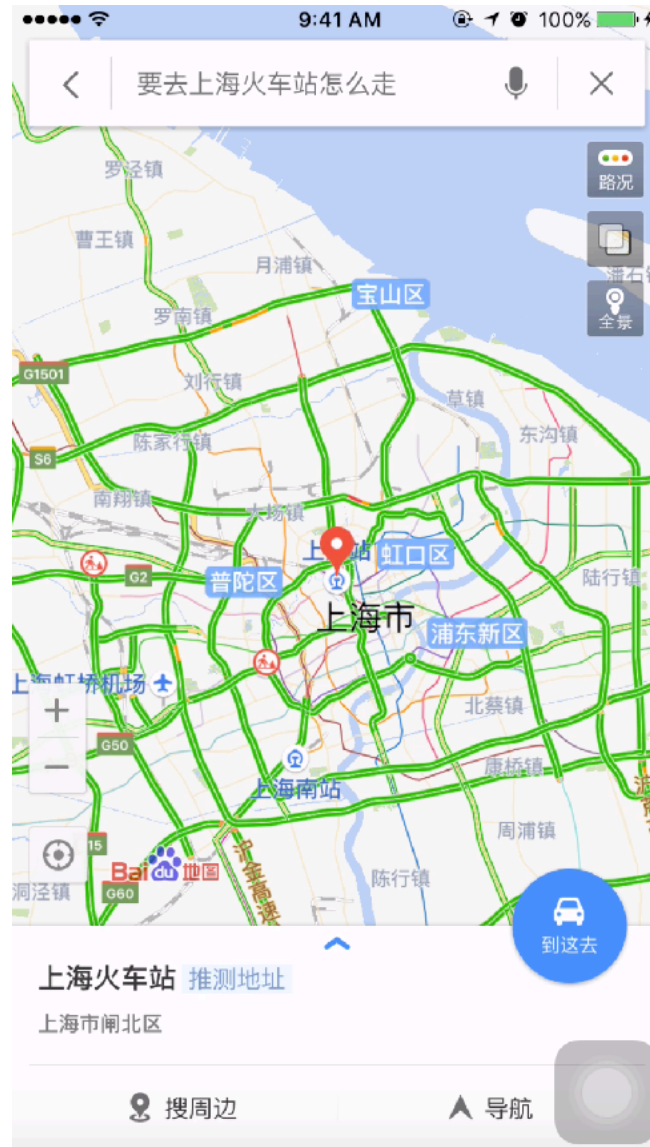
- Key components in Transformer
  - Positional Embedding (to distinguish tokens at different pos)
  - Multihead attention
  - Residual connection
  - layer norm
- Transformer is effective for machine translation, and many other tasks
- Pre-training: using unlabeled raw data to train a model
- BERT: masked pre-training

Next Up

- Probabilistic Graphical Models

Discussion Topic

- Building a voice dialog interface for Baidu/Google Map
- Voice input
  - use ASR sdk to output transcript (80% acc)
- Queries belong to 3 domains
  - lbs_poi, lbs_route, lbs_nav
- Semantic fields for each domain
  - Different fields for domain
  - Intent (search, open)
  - Keywords, origin, destination, etc.
- 8 million query logs to start with



LBS query intention parsing

南宁到防城港白浪滩
Domain: lbs_route

Origin: 南宁

Destination: 防城港白浪滩

from Nanning to Fangchenggang white beach

武汉理工大学附近的拉面馆
Domain: lbs_poi

Centre:武汉理工大学

Keywords:拉面馆

handmade noodle house near Wuhan Tech University