CS182/282A: Designing, Visualizing and Understanding Deep Neural Networks

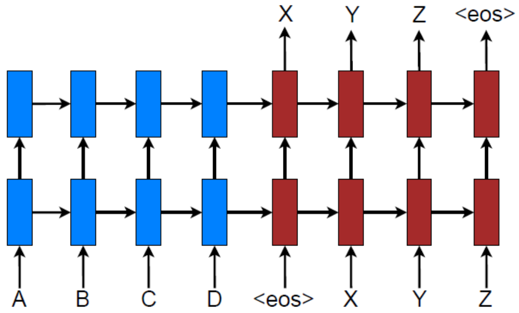
John Canny

Spring 2020

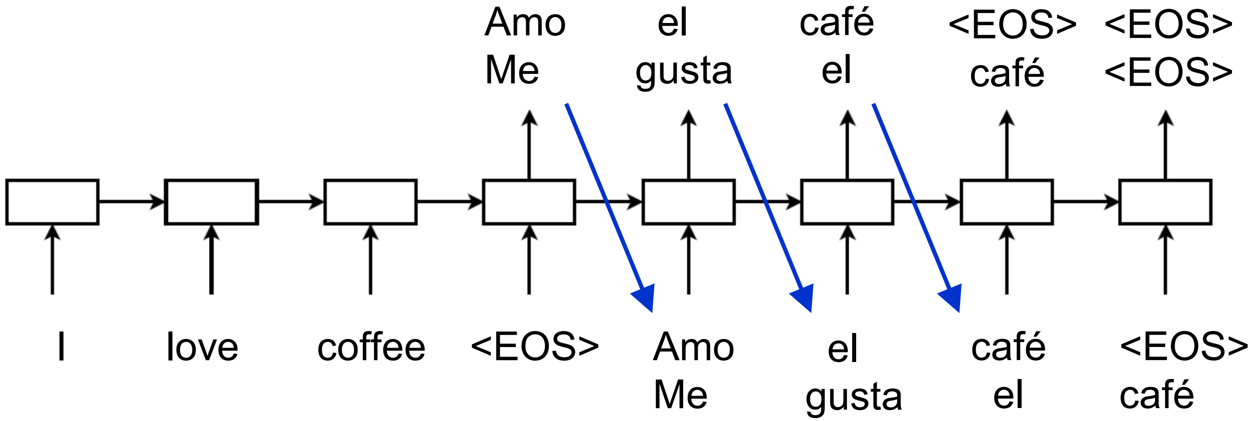
Lecture 14: Transformers and Pre-training

Last Time: Sequence-To-Sequence Translation

Encoder Decoder

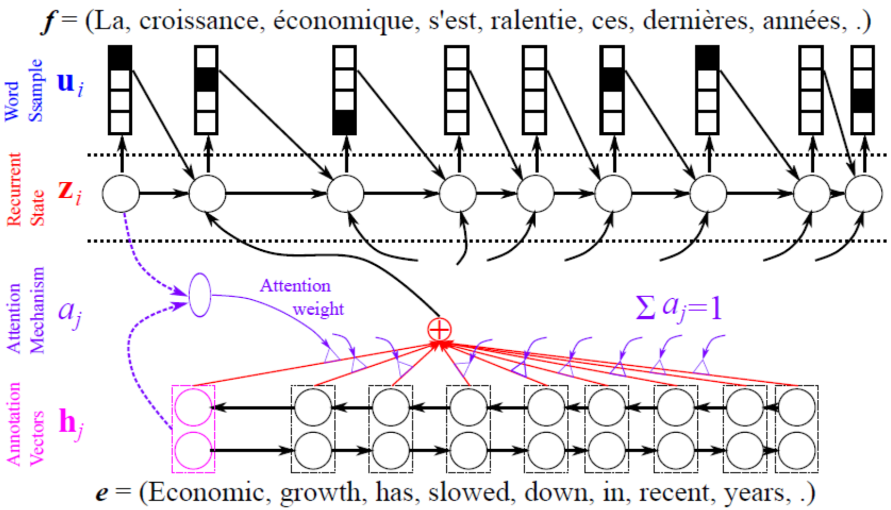


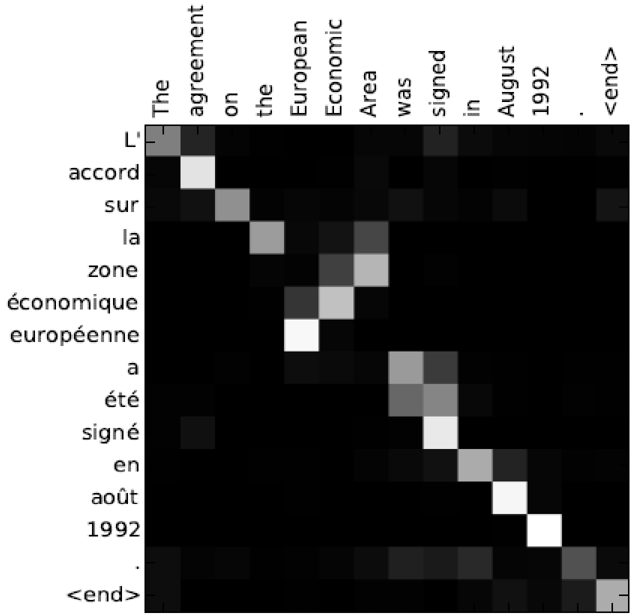
A depth-2 model



Beam search width = 2

Last Time: Soft Attention for Translation





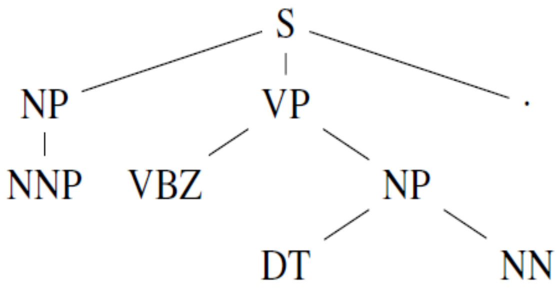
(a)

From Y. Bengio CVPR 2015 Tutorial

Last Time: Parsing as Translation

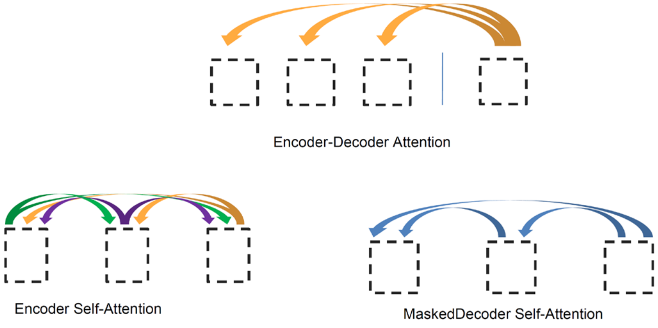
Sequence models generate linear structures, but these can easily encode trees by “closing parens” (prefix tree notation):

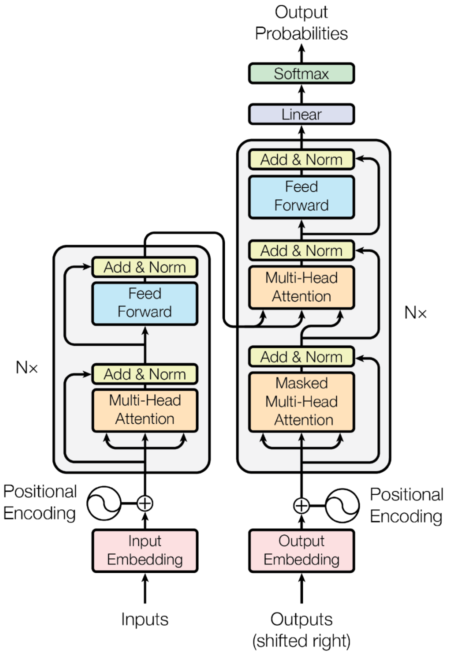
John has a dog . $\rightarrow$

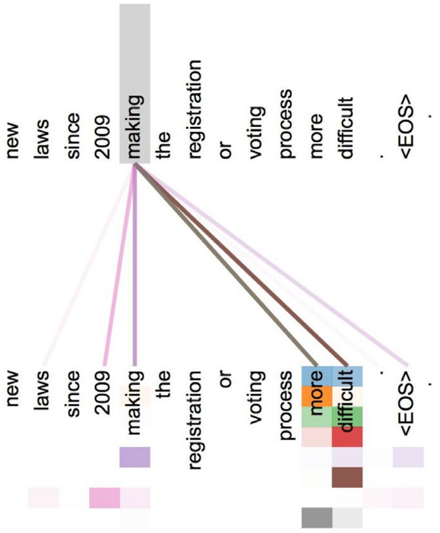


John has a dog . $\rightarrow$ (S (NP NNP $)_{NP}$ (VP VBZ (NP DT NN $)_{NP}$ $)_{VP}$. $)_{S}$

Last Time: Attention only Models: Transformer



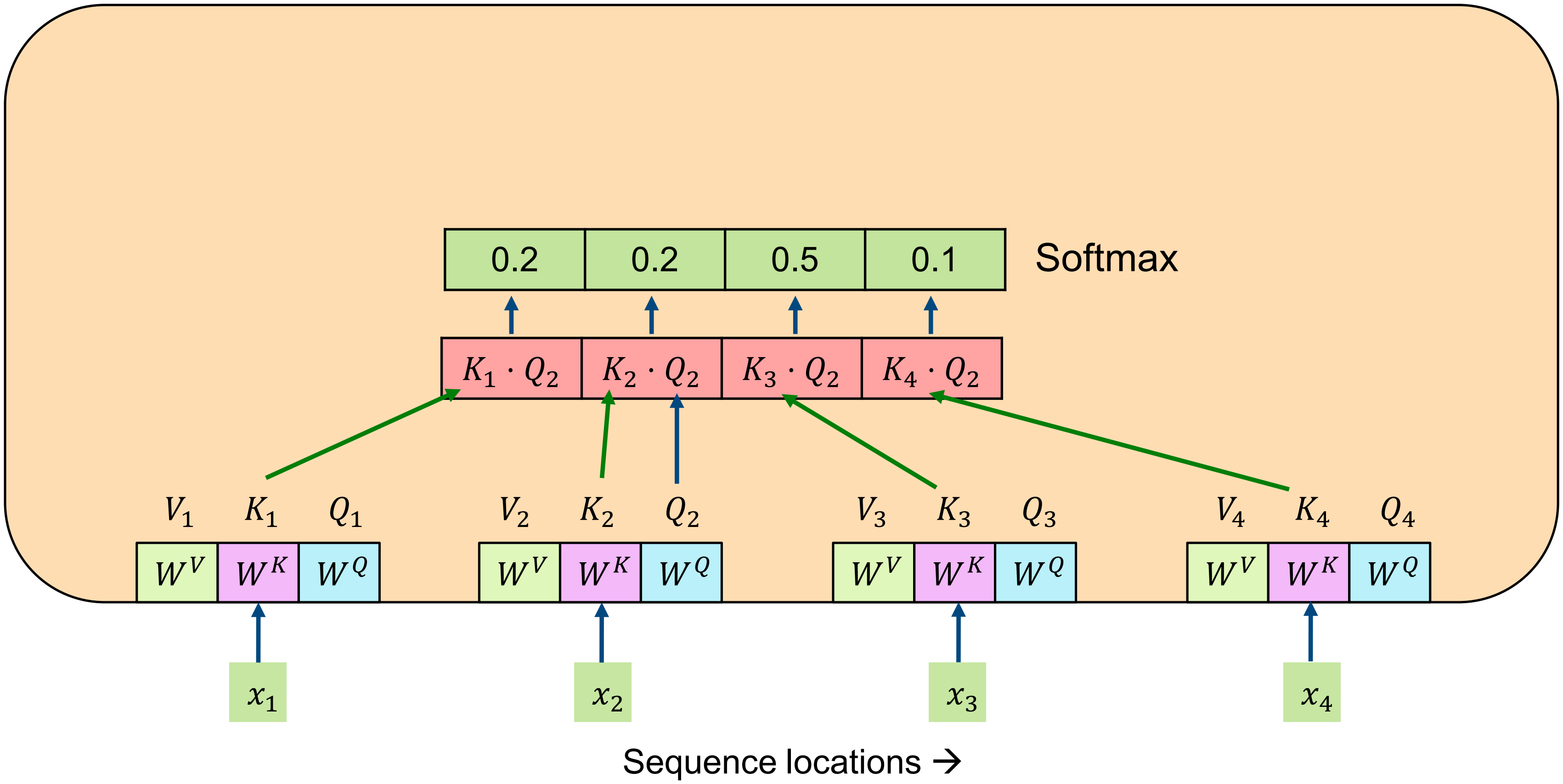




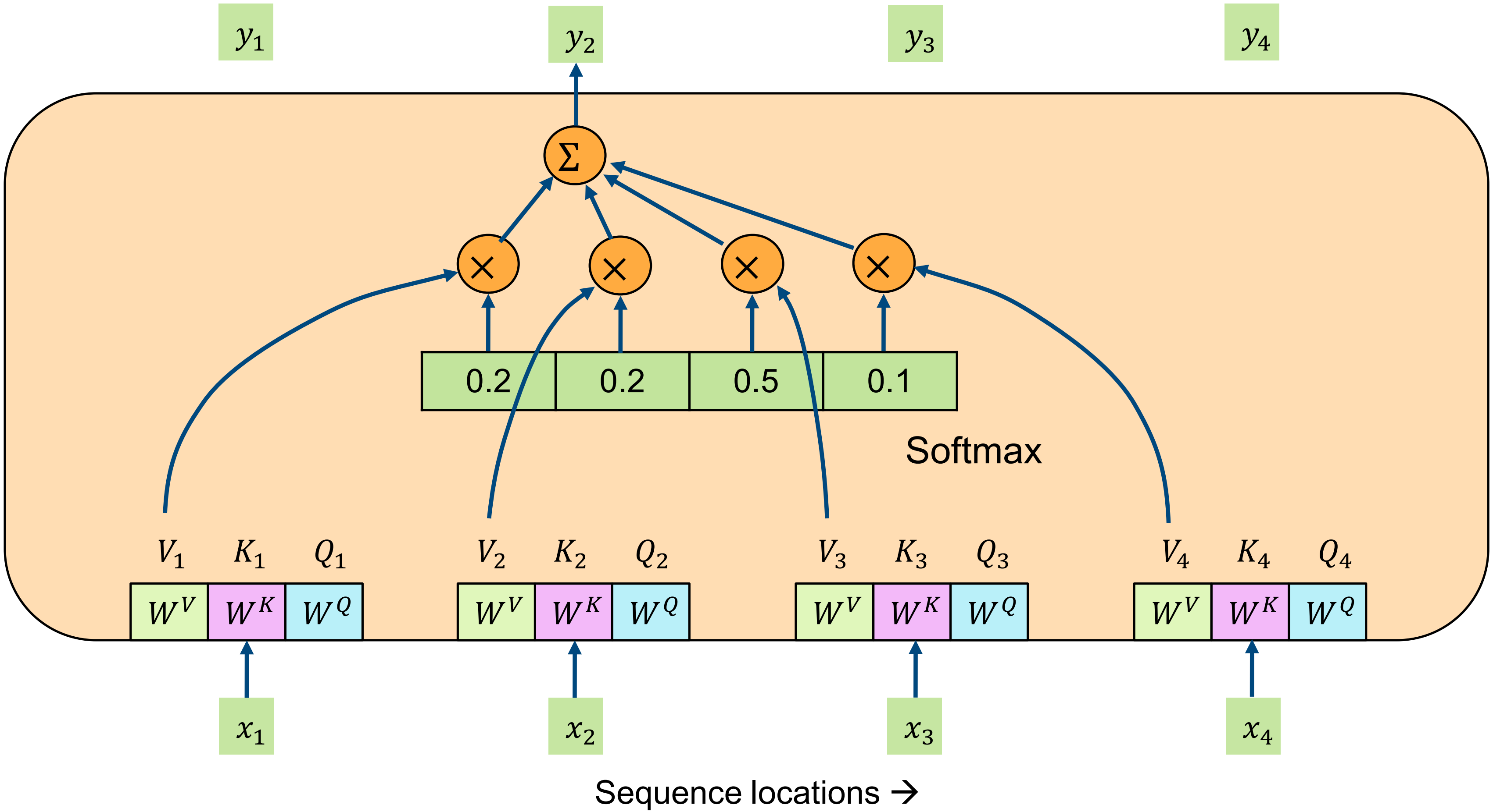
Multi-headed self-attention

image from Lukas Kaiser, Stanford NLP seminar

The Transformer – Self-Attention Layer



The Transformer – Self-Attention Layer



Attention Implementation with matrices

Transformer networks have extreme parallelism by using *matrices* to hold all the vectors in the network:
Q = matrix of all query vectors (as rows)
K = matrix of all keys (as rows)
V = matrix of all values (as rows)

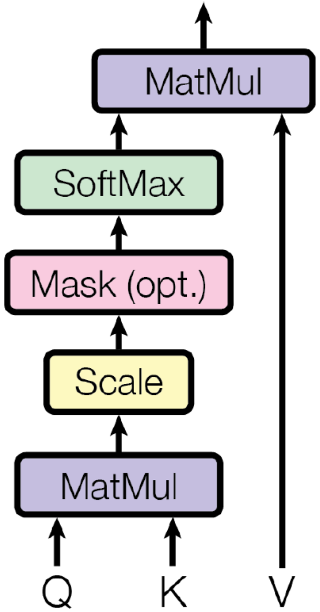
The row index is the position in the sequence.

The entire attention operation can be computed as a *single matrix formula* as:

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where the softmax is applied across rows (not columns).

Scaled Dot-Product Attention

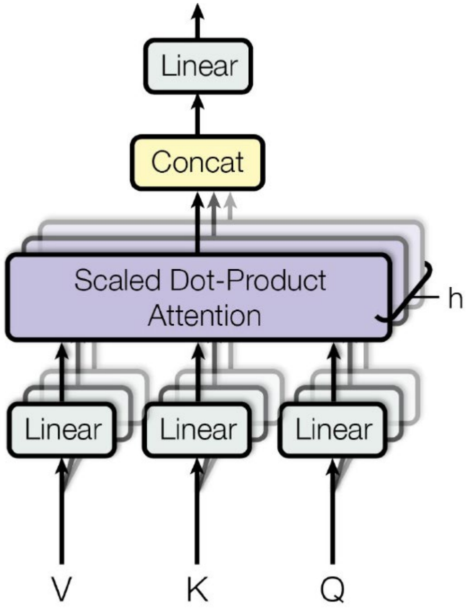


Multi-Headed Attention

- Standard attention allows each location to attend with a single weight/value embedding to another location.

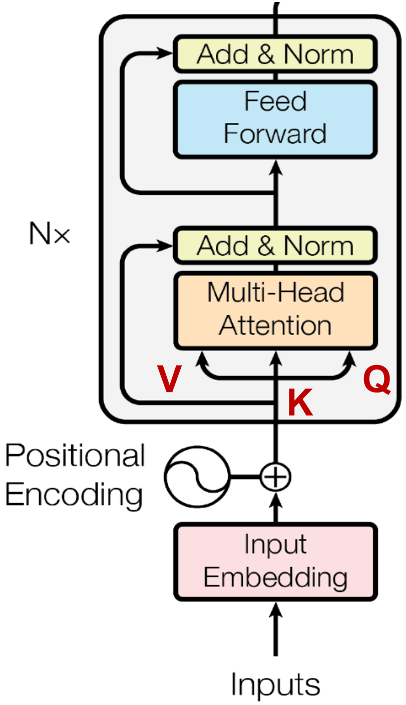
- We can extend this with “multi-headed” attention by breaking inputs and outputs into ranges, and applying different embeddings for each range.

- The figure to the right shows h heads.



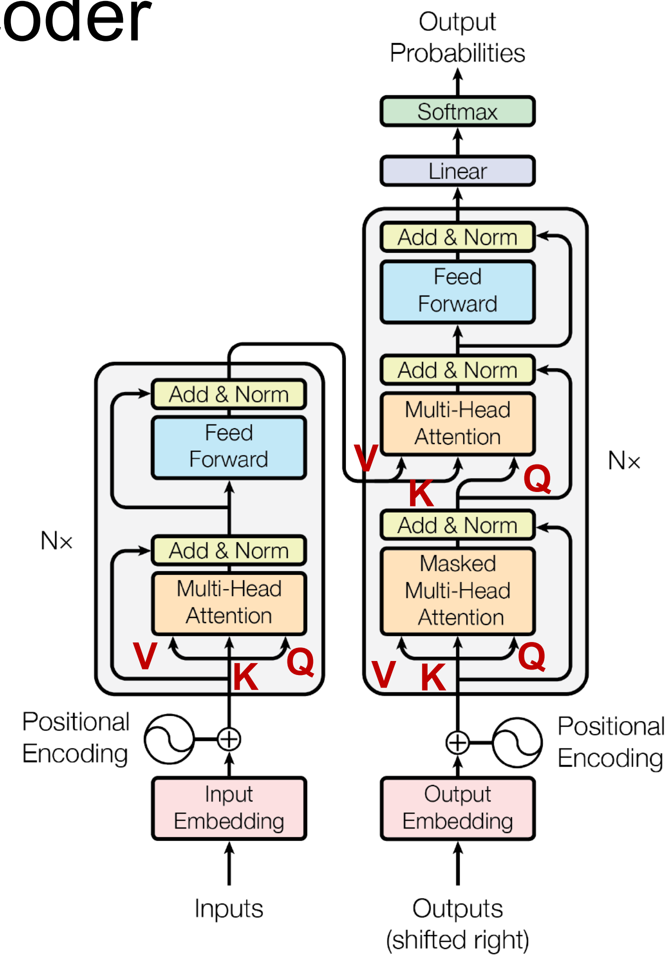
Transformer Encoder

- Basic unit shown at right.
- The input is a sequence of symbols at the bottom.
- Because different positions are encoded as matrices, its common not to show the sequence positions.
- Multiple layers can be stacked.

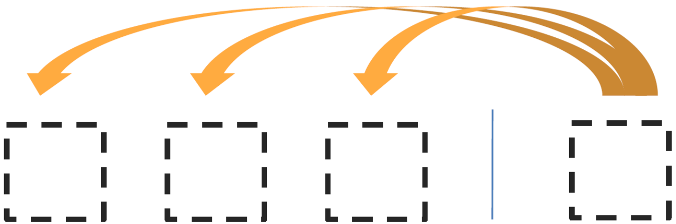


The Transformer Encoder/Decoder

- Basic unit shown at right.
- Now there is both and encoder with self-attention, and a decoder with both masked self-attention and cross-attention.
- In experiments, stacked with N=6.
- Inputs and outputs are embedded in vector spaces of fixed dimension.
- Positional encoding: when words are combined through attention, their location is lost. Positional encoding adds it back.



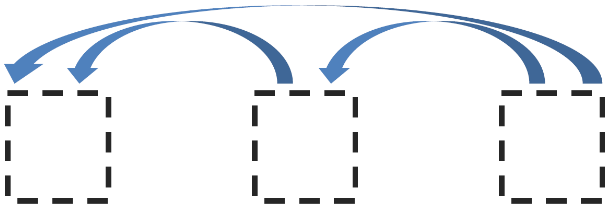
Attention Types in Transformer Networks



We saw this in Bahdanau and Luong models

Encoder-Decoder Attention

Encoder Self-Attention



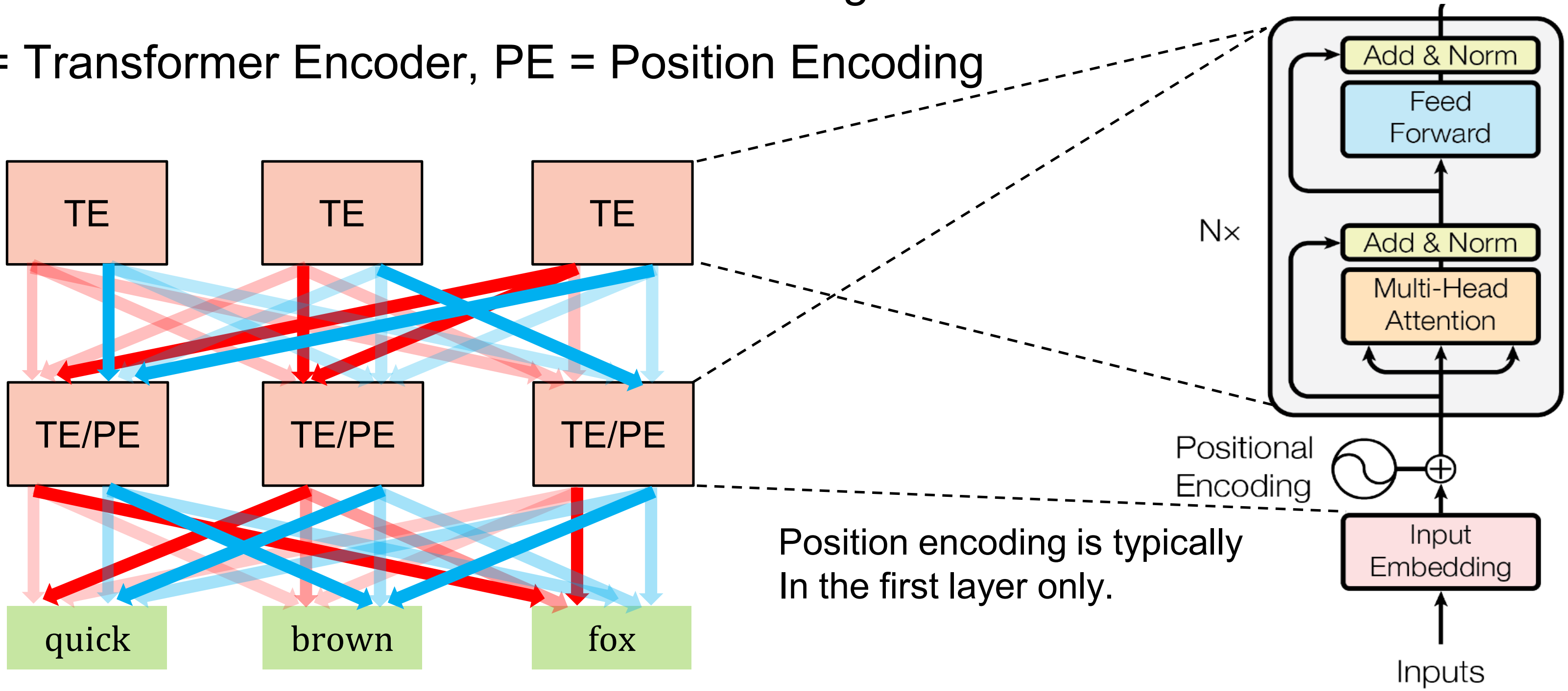
MaskedDecoder Self-Attention

image from Lukas Kaiser, Stanford NLP seminar

The Transformer Encoder

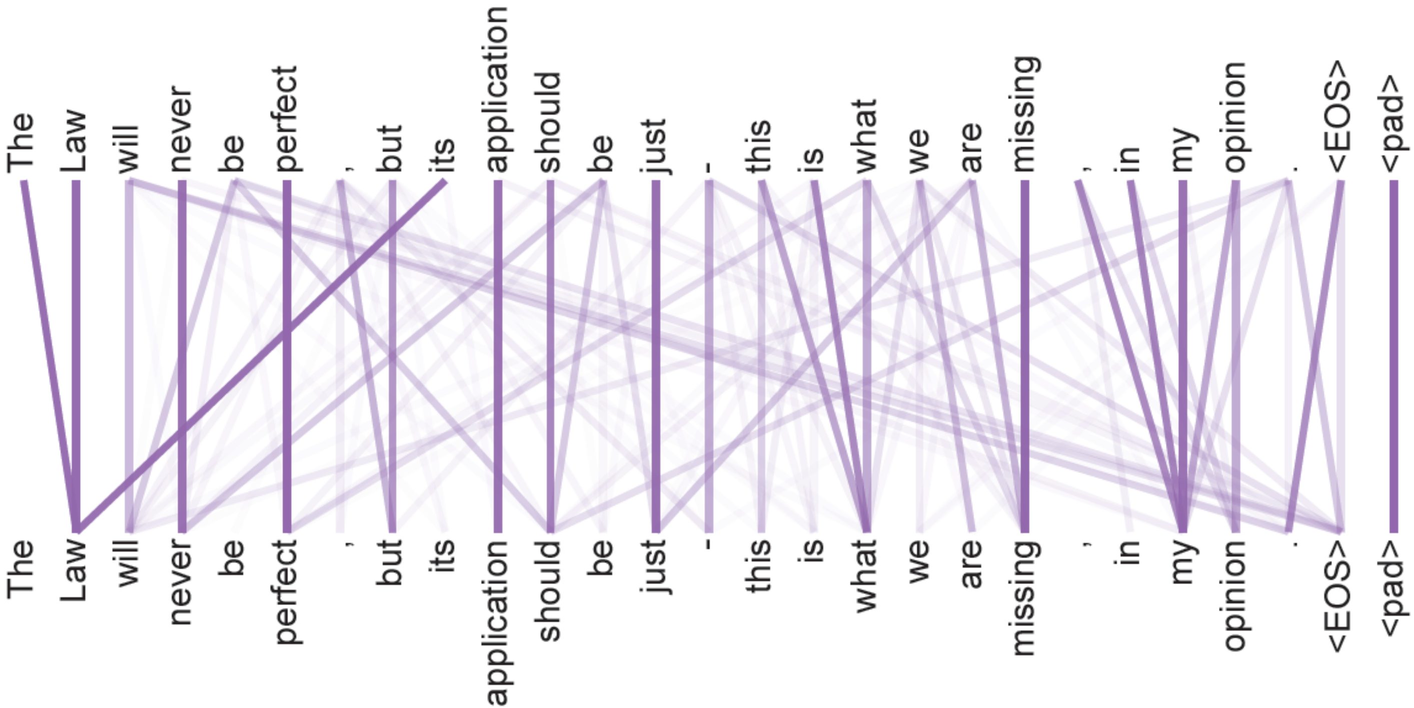
Arrows show red/blue self-attention head weights.

TE = Transformer Encoder, PE = Position Encoding



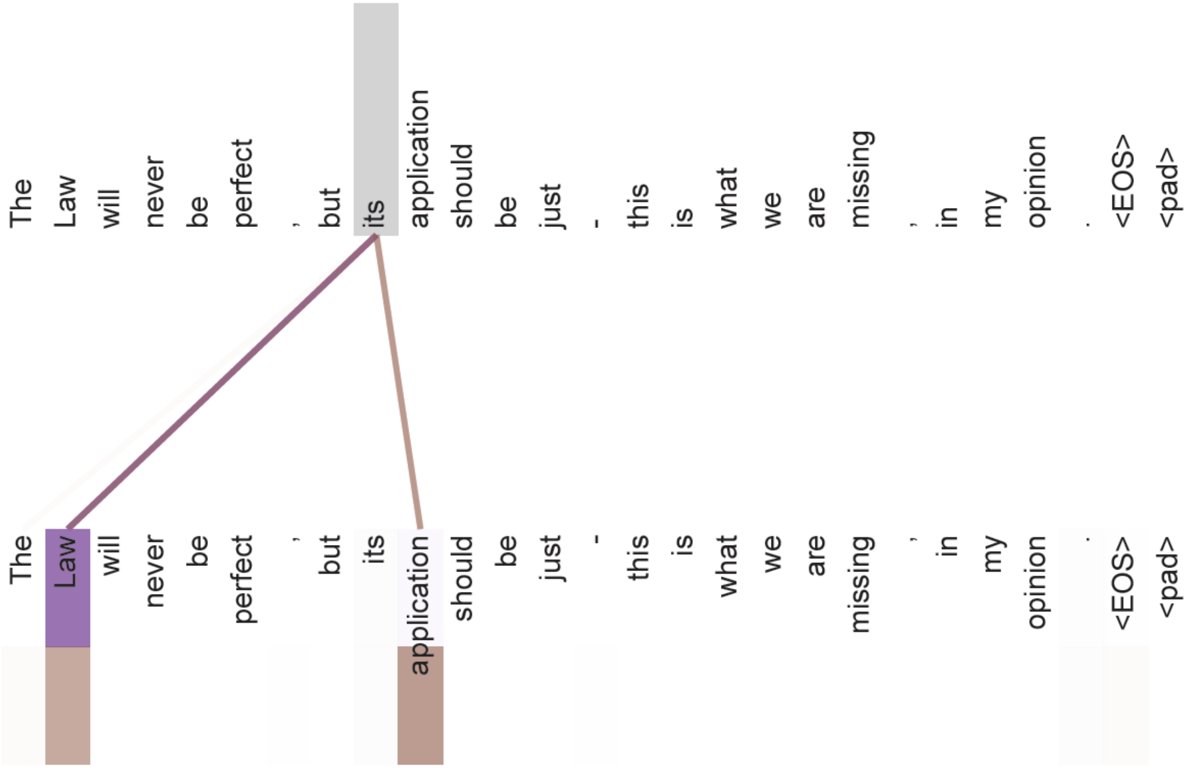
Position encoding is typically
In the first layer only.

Multi-Headed Attention



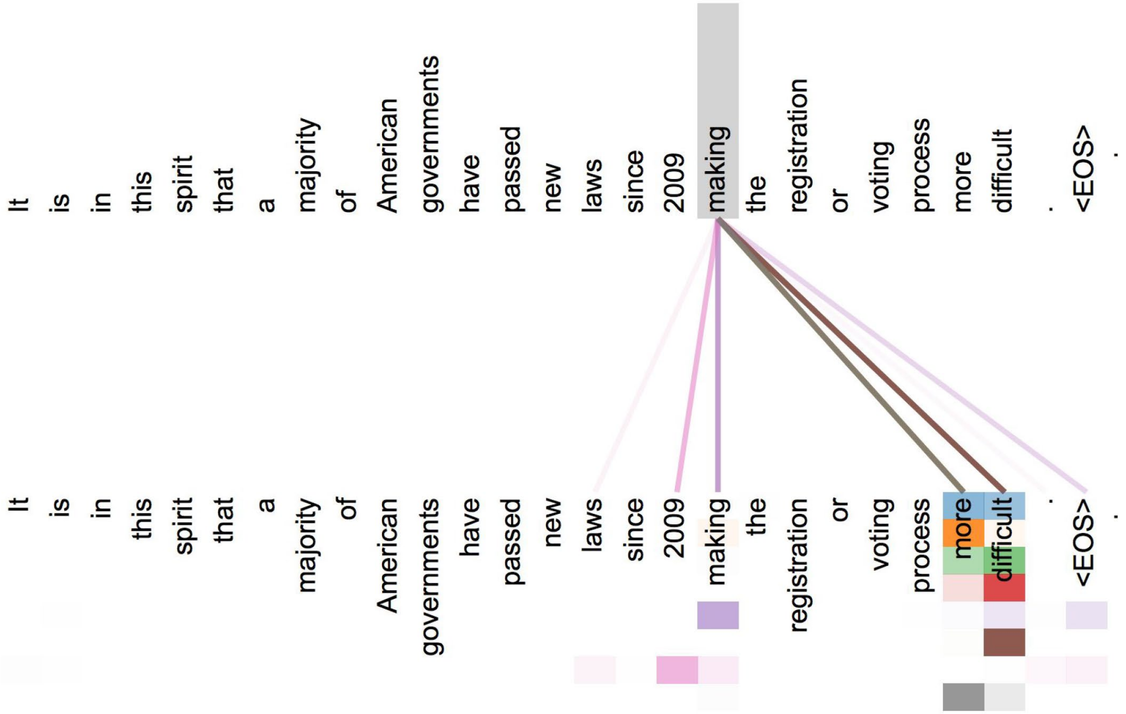
Anaphora (pronoun or article) resolution

Multi-Headed Attention



Anaphora (pronoun or article) resolution

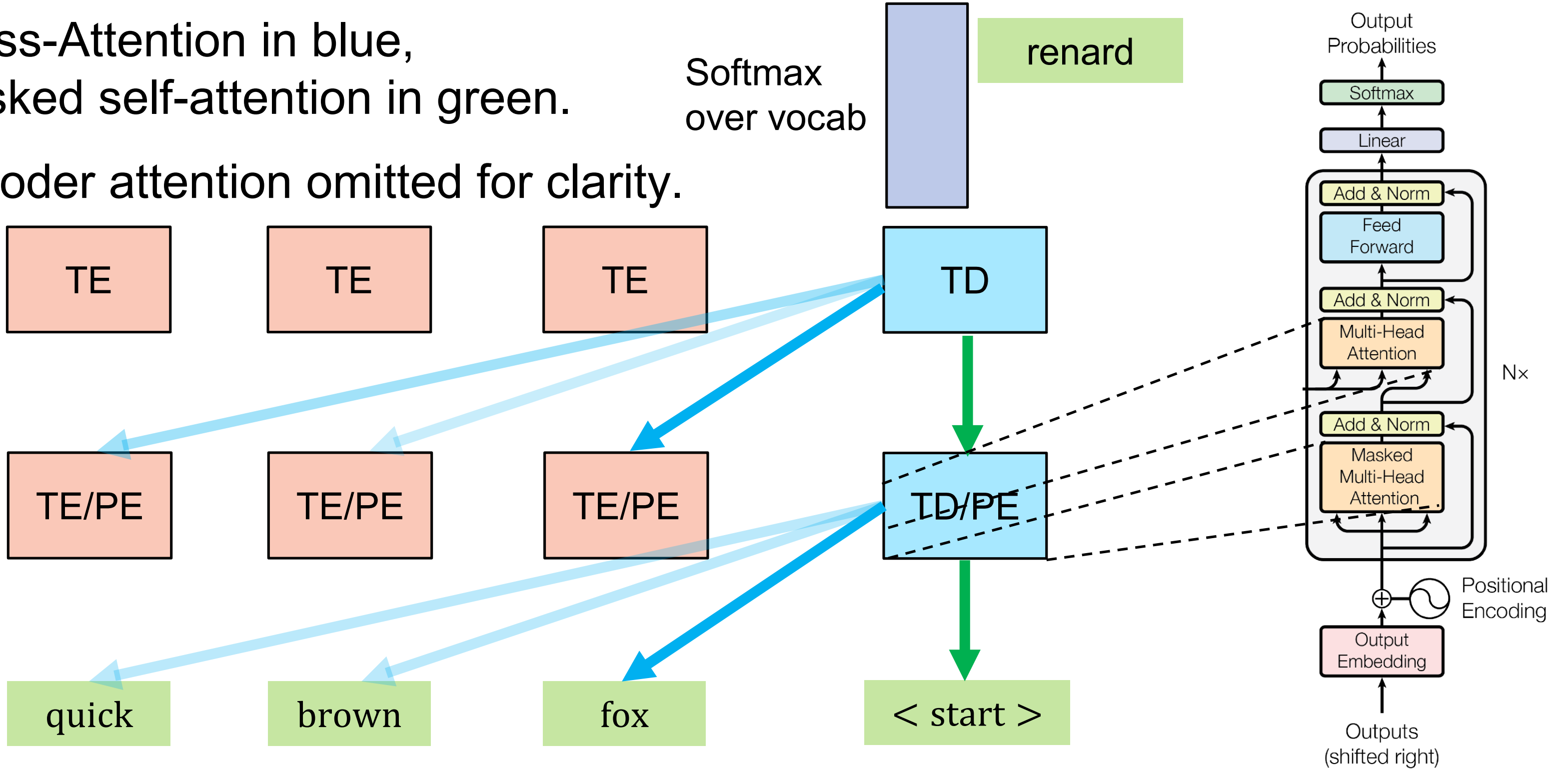
Multi-Headed Attention



Word-by-word Transformer Decoding

Cross-Attention in blue,
masked self-attention in green.

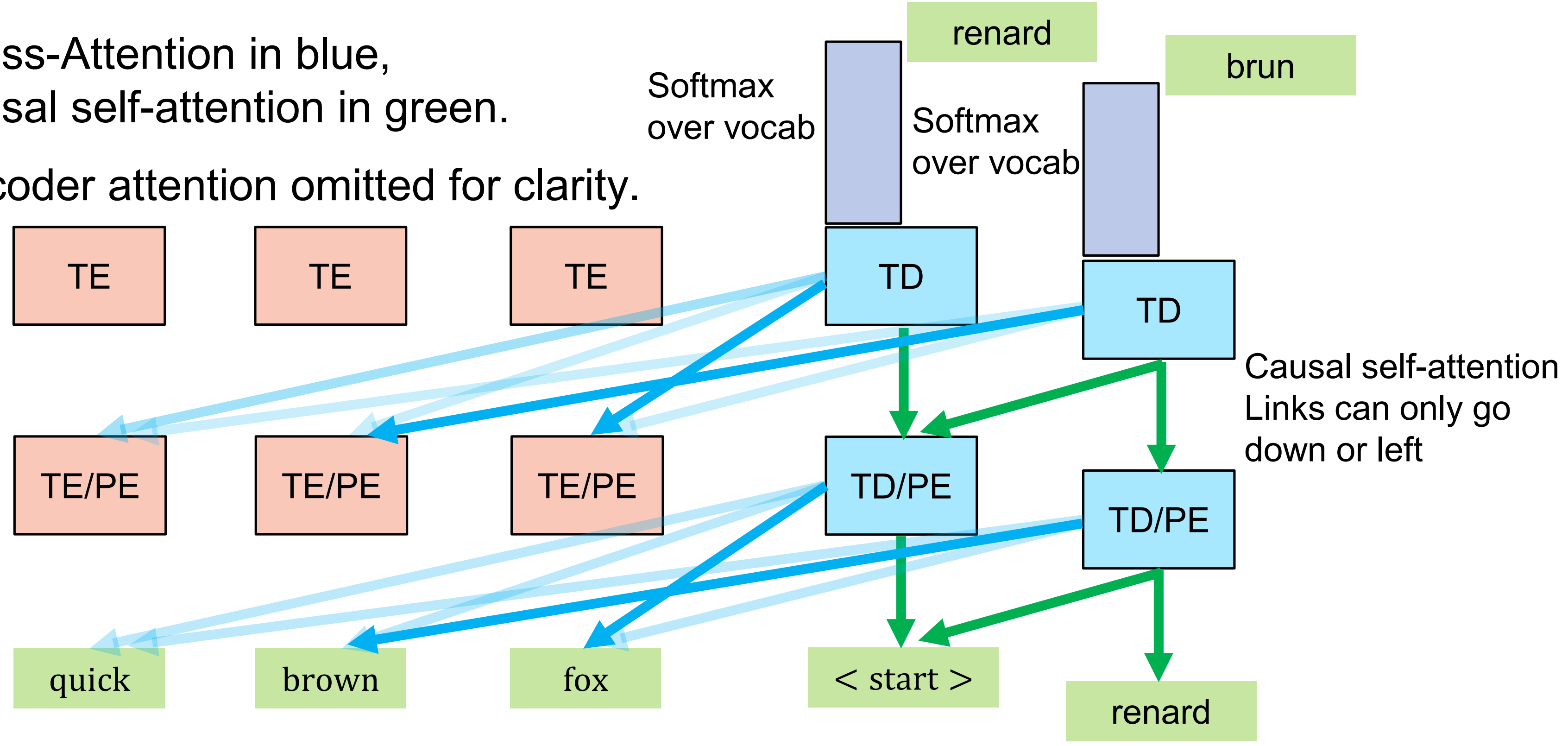
Encoder attention omitted for clarity.



Word-by-word Transformer Decoding

Cross-Attention in blue,
causal self-attention in green.

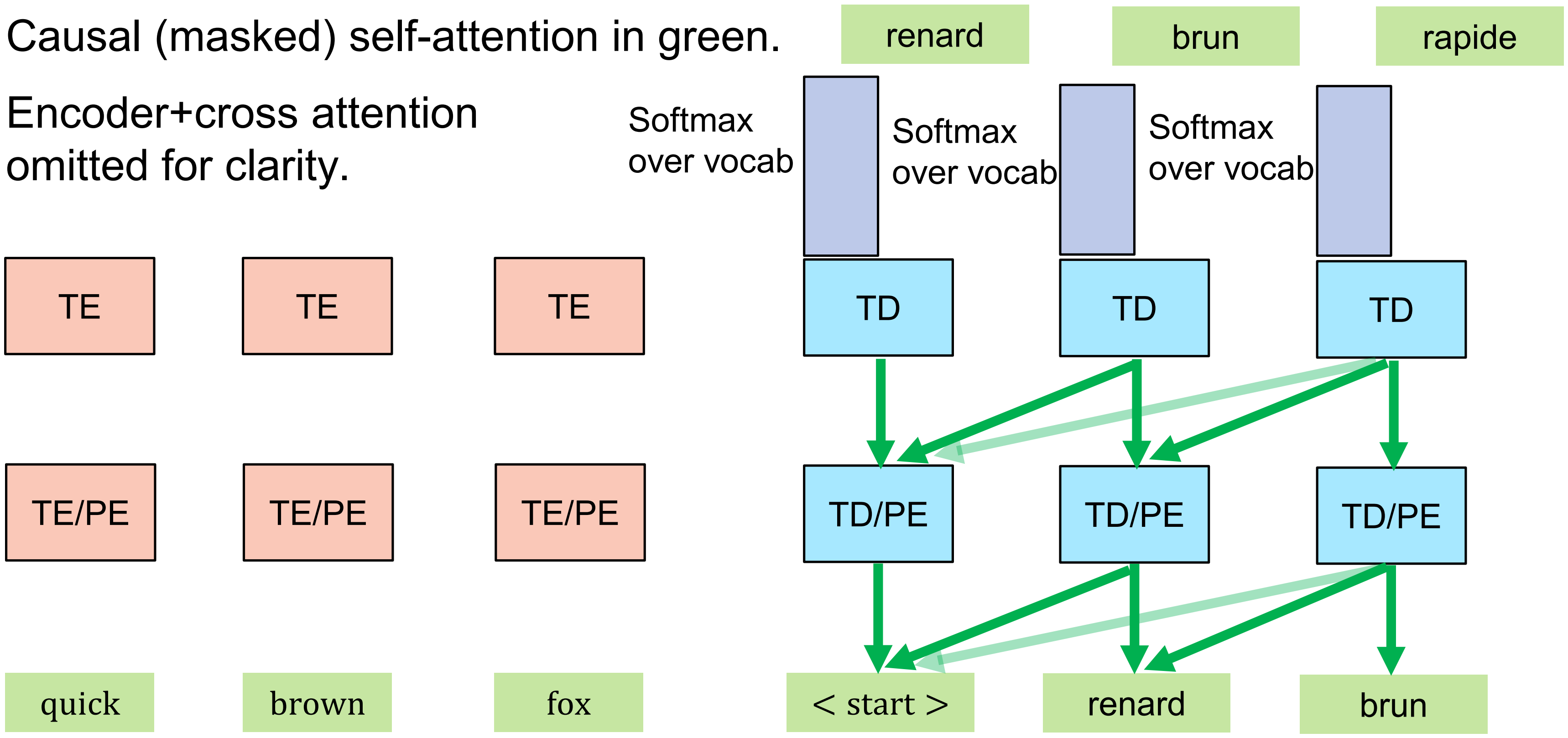
Encoder attention omitted for clarity.



Word-by-word Transformer Decoding

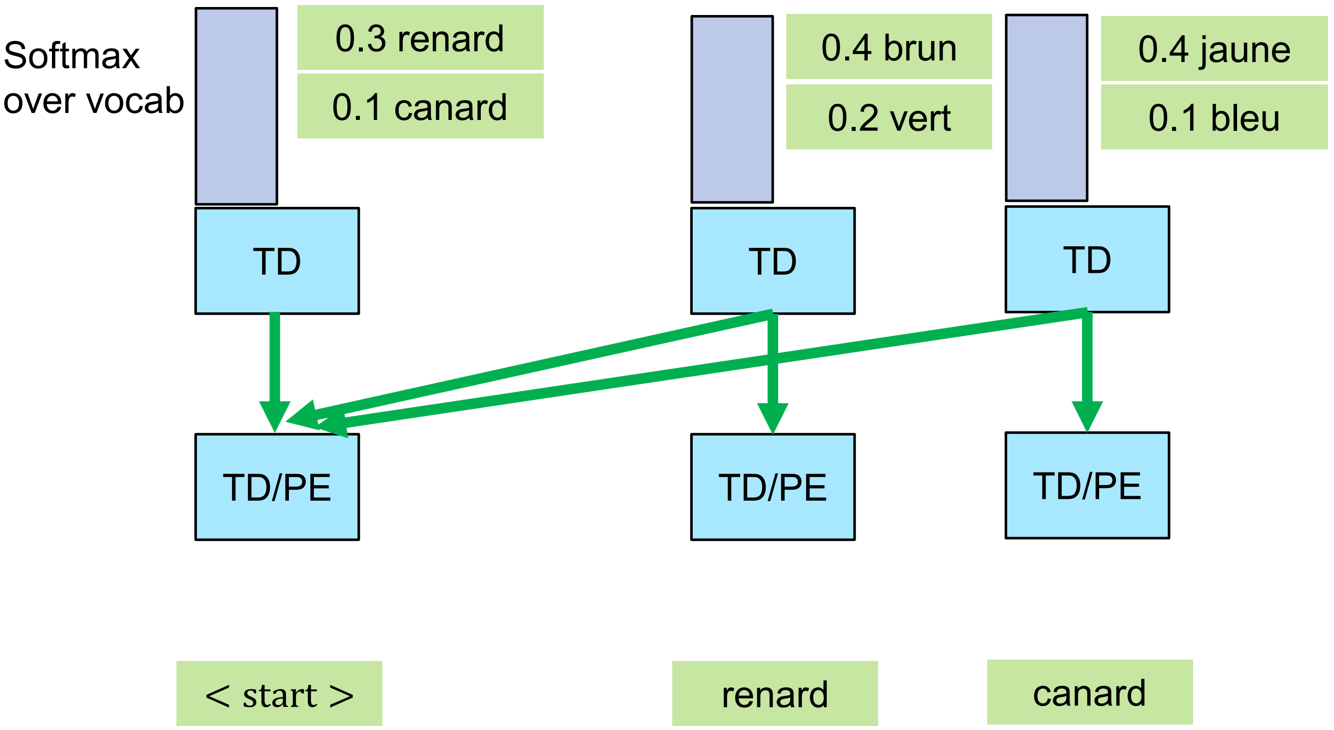
Causal (masked) self-attention in green.

Encoder+cross attention omitted for clarity.



N-Best Transformer Decoding

Two copies of second decoder position:



Two best phrases so far:
0.12 “renard brun”
0.06 “renard vert”

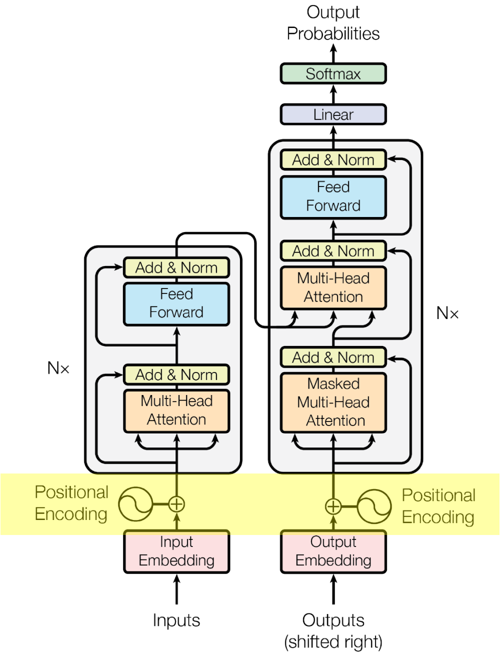
Transformer Position encoding

Every cell in the transformer has the same "view" of the data below. Its important to break this symmetry so different cells do different things. Spatial encoding is usually used:

The encoding vector has the same dimension as the model.

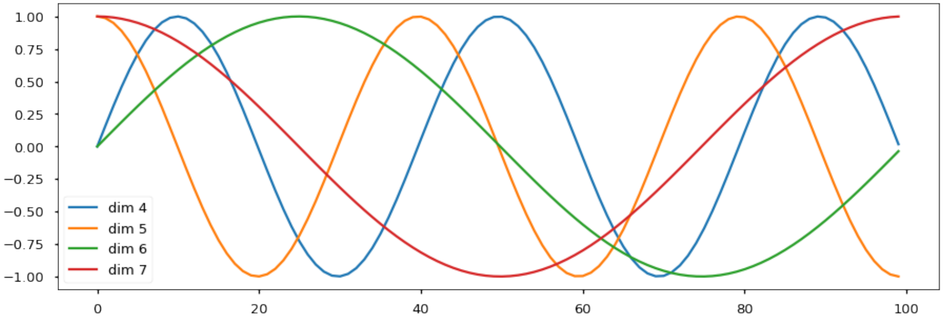
Its components are all sinusoidal functions of position.

The periods of the sinusoids form a geometric series.



Position encoding is $PE(x,k)$ for position x and component k of the encoding.

value of $PE(x,k)$:



Position in the input x

Different colors = different values of k

Position Encoding – Relative Positions

The position encoding is given by $PE(x,k)$ for position x and encoding dimension k.

even dimensions of the encoding vector: $PE(x,2i) = \sin(x/10000^{2i/d})$

odd dimensions of the encoding vector: $PE(x,2i+1) = \cos(x/10000^{2i/d})$

Why ??? – empirical – gives similar performance to a learned encoding.

Its good for measuring relative positions:
the dot product $PE(x,:) \cdot PE(y,:)$ depends only
on $y - x$, the relative displacement between x and y.

$PE(x,:) \cdot PE(y,:)$

0

$y - x$

Position Encoding – Predictive Information Version

The position encoding is given by $PE(x,k)$ for position x and encoding dimension k.

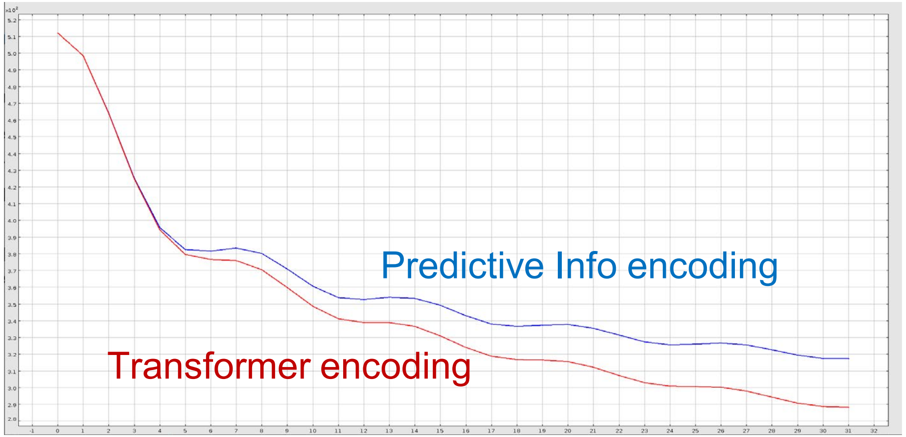
Let $r(i) = \left((2i+1)/d\right)^{1/(1-p)}$ where $p = 0.883$ *

even dimensions of the encoding vector: $PE(x,2i) = \sin(x * r(i))$

odd dimensions of the encoding vector: $PE(x,2i+1) = \cos(x * r(i))$

Why ??? – the dot product between two locations is the expected predictive information between them.

$$PE(x,:) \cdot PE(y,:)$$

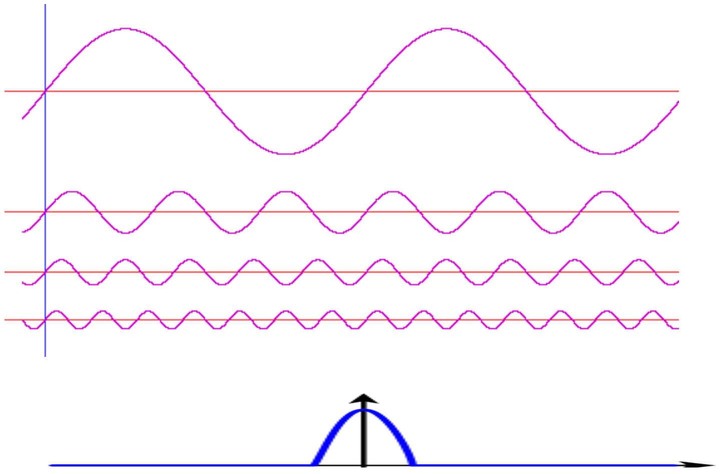


0 $\qquad y - x$

* Info in a text of length l grows as $l^{0.883}$

Position Encoding - Ranges

The advantage of this representation is that the model can learn a linear combination of the sinusoids that is strongest at any particular word position, or a range of positions.



Transformer Results

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	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Tokenization Challenges

Problem:

A fixed vocabulary does not account for **language evolution.**

How do you represent:

"Boeing's new Starliner is built to carry astronauts."

Tokenization Challenges - UNK Token

How do you represent:

"Boeing's new **Starliner** is built to carry astronauts."

Solution 1: Any word not in the vocabulary is replaced by a special "unknown" token.

```
>> toks = ["Boeing", " 's", "new", "UNK", "is", "built", "to", "carry", "astronauts", "."]
```

Limitation: All unknown words are represented by the same word vector to the model, limiting model understanding.

Tokenization Challenges - Large Vocab

How do you represent:

"Boeing's new **Starliner** is built to carry astronauts."

<u>**Solution 2:**</u> Increase vocabulary size. NLP models saw vocabulary size grow from 50k to around 200k (until 2017).

>> toks = ["Boeing", " 's", "new", "Starliner", "is", "built", "to", "carry", "astronauts", "."]

<u>Limitation:</u>

- Larger vocabulary means larger model size.
- Model cannot learn good representations for words it sees a few times.
- UNK can still occur (vocab is not infinite)

Tokenization Challenges - Word Piece

How do you represent:

"Boeing's new Starliner is built to carry astronauts."

Solution 3: Allow for words to be broken down into "pieces" (e.g. syllables) if they are not present in the vocabulary.

>> toks = ["boeing", " 's", "new", "star", "##liner", "is", "built", "to", "carry", "astronauts", "."] # (output of the BERT uncased tokenizer)

There are several algorithms for WordPiece tokenization:

Byte-Pair Encoding (Sennrich et al. 2016), Unigram LM (Kudo 2018)

Word Piece - Advantages and Limitations

Advantages:

- Small vocabulary (a parameter of the Word Piece model)
- No more UNK tokens (unless new unicode characters are introduced)
- Can have a multilingual Word Piece model (e.g. XLM)
- Sub-word units can have semantic meaning the model can learn.

  For instance, the Word Piece tokenization of "california", "californian", "californians" is:

  ["california"], ["california", "##n"], ["california", "##ns"]

Word Piece - Advantages and Limitations

Limitations

- Increases sequence lengths: our 7-word example has a 10 word-piece representation.
- Tokenization is no longer a reversible operation: a string can be represented by several sequences: "carrot" = ["carrot"] but "carrot" = ["car", "##rot"]
- Need to train the Word Piece model ahead of time. Added complexity.

Nowadays, most Transformer architectures are coupled with a Word Piece model.

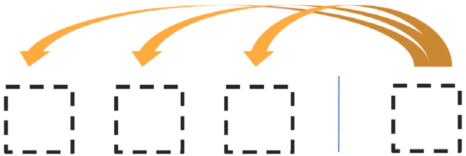
English-to-English Translation ?!

Yes, it does make sense. a.k.a. summarization.

Liu et al, “GENERATING WIKIPEDIA BY SUMMARIZING LONG SEQUENCES” arXiv 2018

M = input length, N = output length

Summarization: M >> N

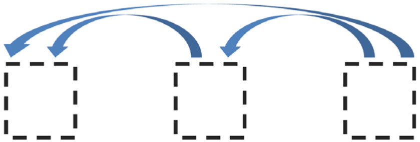


$O(MN)$

Encoder-Decoder Attention

$O(M^2)$

Encoder Self-Attention



$O(N^2)$

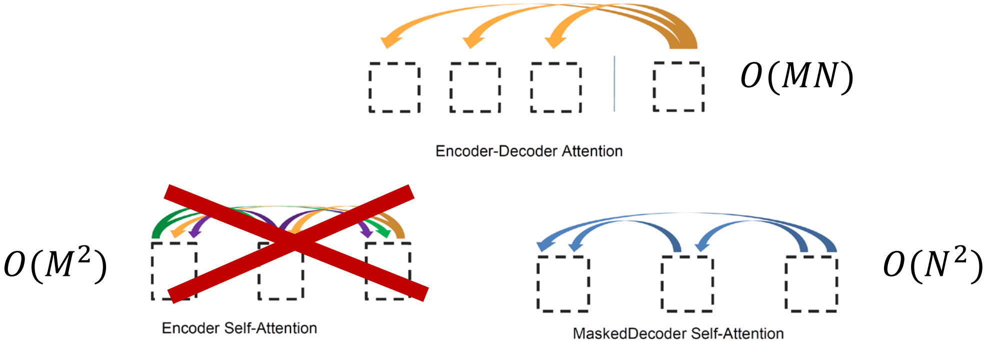
MaskedDecoder Self-Attention

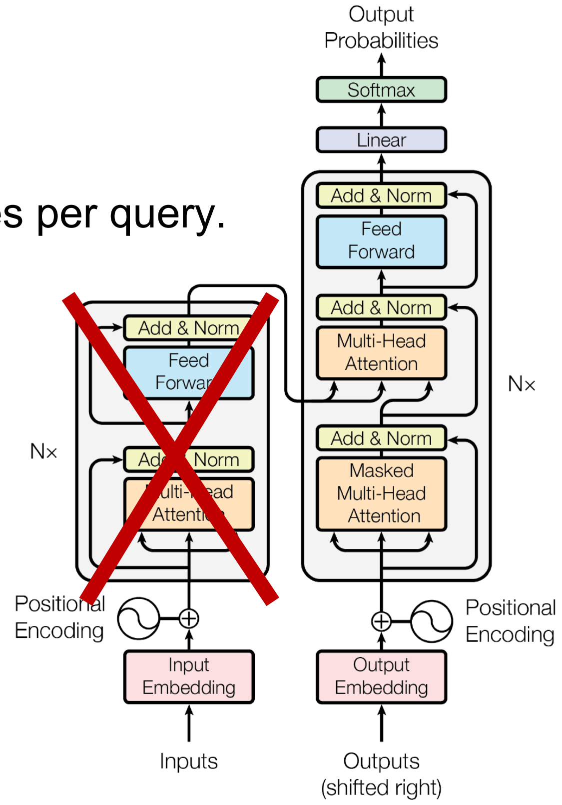
Large-scale Summarization (Wikipedia)

Like translation, but we completely remove the encoder.

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.





Large-scale Summarization (Wikipedia)

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.

For a passage of length N and a summary of length M, the complexity of the attention is:

A. $O(N) + O(M)$

B. $O(N) + O(M) + O(NM)$

C. $O(N^2) + O(M^2) + O(NM)$

D. $O(N^2) + O(M^2)$

Oops!

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.

For a passage of length N and a summary of length M, the complexity of the attention is:

A. $O(N) + O(M)$

No, self attention is all-to-all and so quadratic.

Try Again

Continue

Oops!

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.

For a passage of length N and a summary of length M, the complexity of the attention is:

B. $O(N) + O(M) + O(NM)$

No, self attention is all-to-all and so quadratic in M and N.

Try Again

Continue

Correct!

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.

For a passage of length N and a summary of length M, the complexity of the attention is:

C. $O(N^2) + O(M^2) + O(NM)$

Yes. The three terms are respectively the Encoder self-attention, Decoder self-attention, and Cross attention.

Try Again

Continue

Oops!

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.

For a passage of length N and a summary of length M, the complexity of the attention is:

D. $O(N^2) + O(M^2)$

No, cross attention is missing.

Try Again

Continue

Wikipedia

Source data (large!):

- The references for a Wikipedia article.
- Web search using article section titles, ~ 10 web pages per query.

When N >> M as it is for summarization, we drop the Encoder self-attention $O(N^2)$.

Large-scale Summarization

Results:

Model	Test perplexity	ROUGE-L
seq2seq-attention, $L = 500$	5.04952	12.7
Transformer-ED, $L = 500$	2.46645	34.2
Transformer-D, $L = 4000$	2.22216	33.6
Transformer-DMCA, no MoE-layer, $L = 11000$	2.05159	36.2
Transformer-DMCA, MoE-128, $L = 11000$	1.92871	37.9
Transformer-DMCA, MoE-256, $L = 7500$	1.90325	38.8

L = input window length.
ED = encoder-decoder.
D = decoder only.
DMCA = a memory compression technique (strided convolution).
MoE = mixture of experts layer.

Liu et al, “GENERATING WIKIPEDIA BY SUMMARIZING LONG SEQUENCES” arXiv 2018

BERT and Pre-Trained Transformer Models

Woman

Queen

Man

King

RNN

ELMo

Embeddings from
Language Models

Transformer

GPT

BERT

Bidirectional Encoder
Representations from
Transformers

BERT

BERT is a language model (next word predictor) trained on a large dataset of natural language that is fine-tuned for particular tasks.

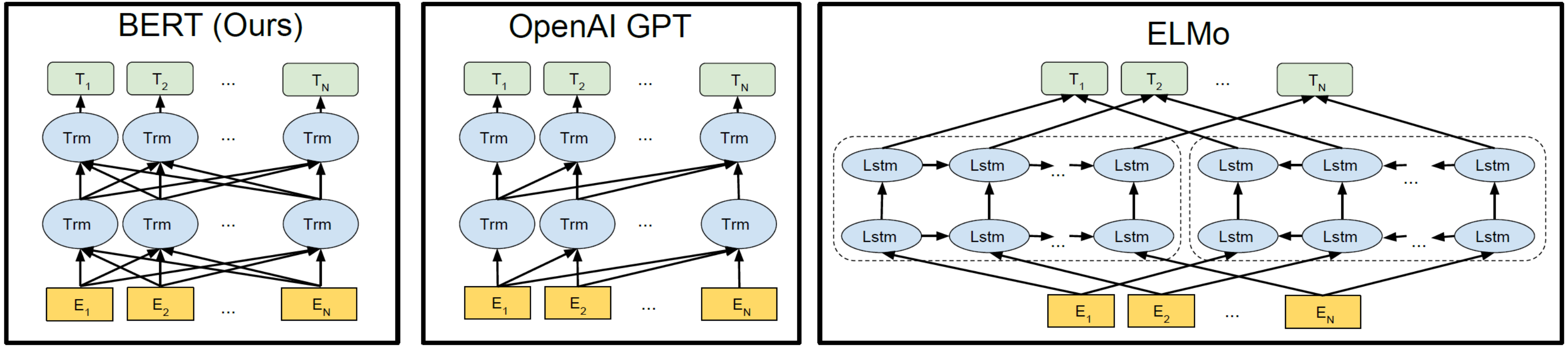
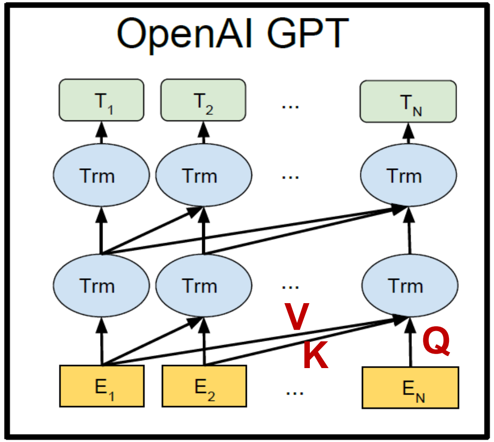


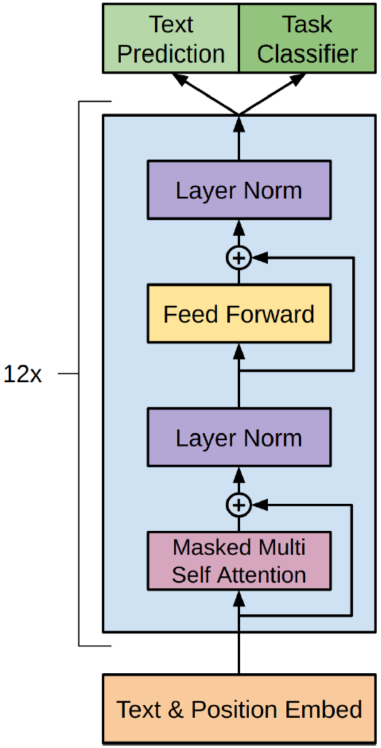
Figure 1: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTM to generate features for downstream tasks. Among three, only BERT representations are jointly conditioned on both left and right context in all layers.

But note that only GPT is a generative model.

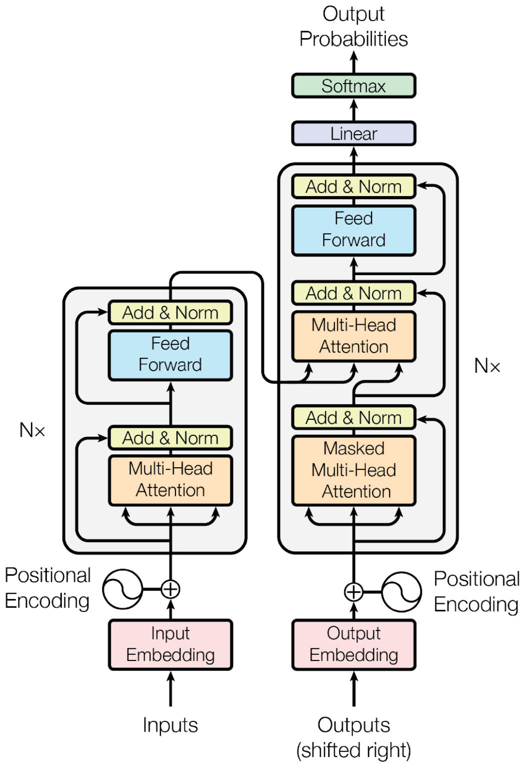
GPT

Generative Pre-Training (OpenAI) is a transformer-based generator using only a simplified transformer decoder stage:





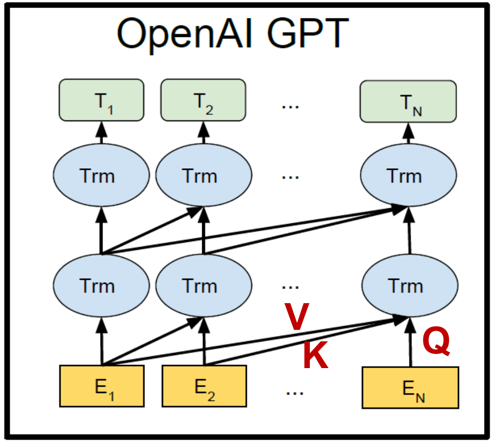
GPT

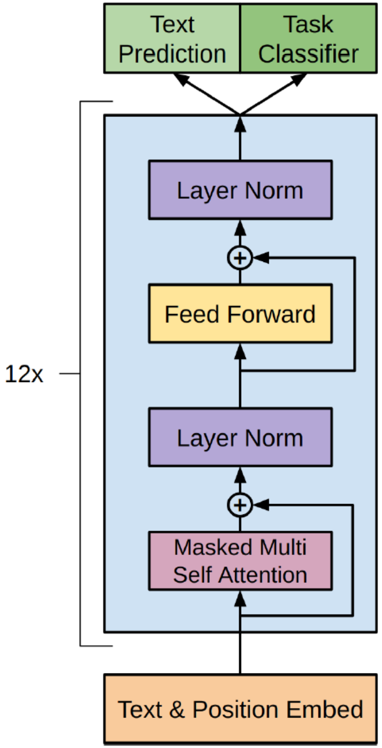


Full transformer

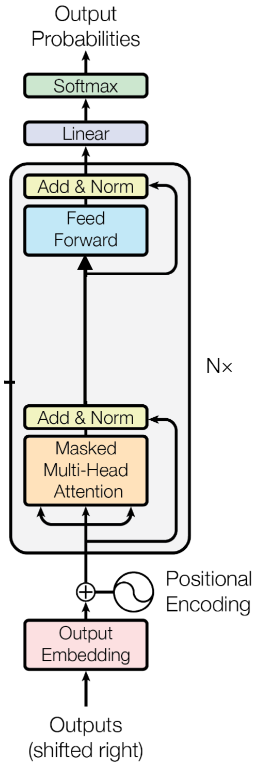
GPT

Generative Pre-Training (OpenAI) is a transformer-based generator using only a simplified transformer decoder stage:





GPT



Full transformer

GPT

GPT is trained initially on a large text corpus to minimize a language modeling loss

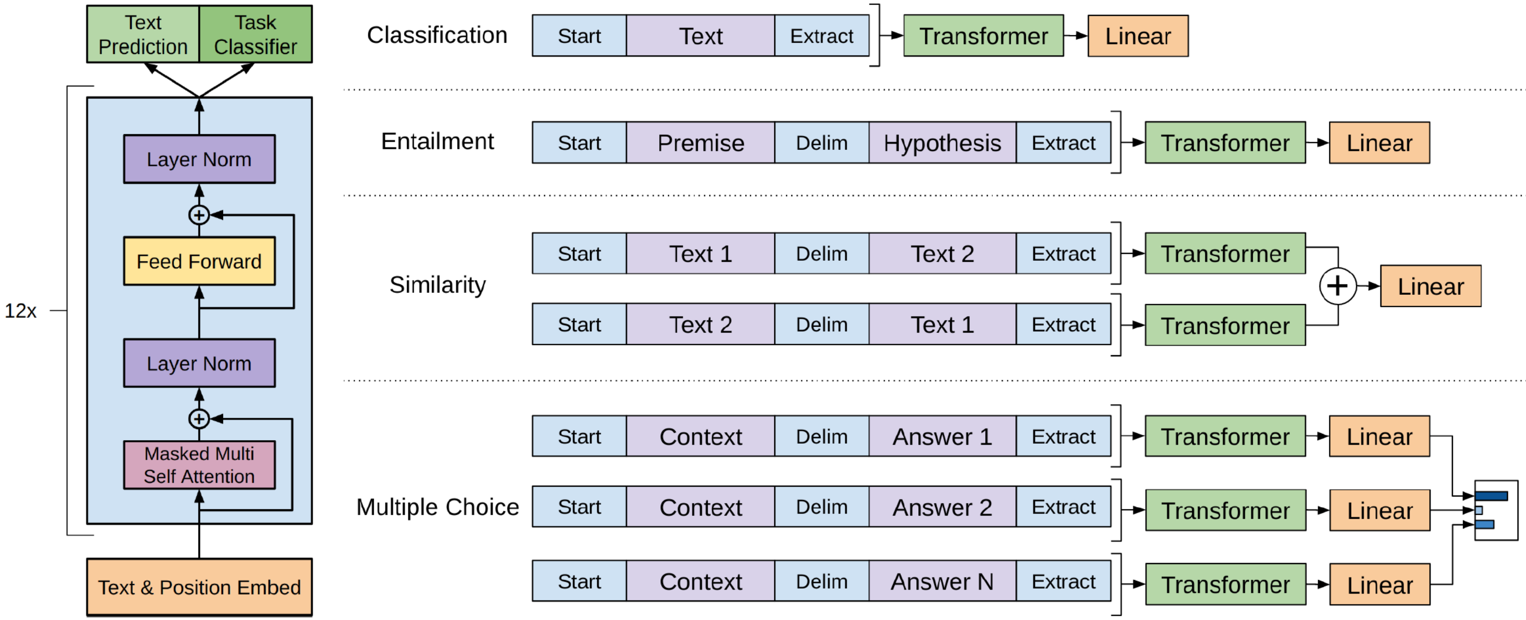
$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

Then for each task, a custom linear layer is added and the entire network retrained (with slower adjustment of the transformer weights).

The language model loss is retained during task-specific retraining of the model.

GPT

Generative Pre-Training can be applied to a variety of (non-generative) tasks:



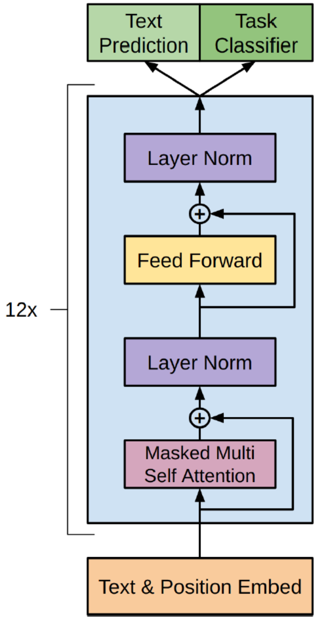
GPT Task performance

Entailment tasks (predict entailment, contradiction, or neutral):
"Anne drove to work" → "Anne has a job"

if a person would say B is probably true given A.

Table 2: Experimental results on natural language inference tasks, comparing our model with current state-of-the-art methods. 5x indicates an ensemble of 5 models. All datasets use accuracy as the evaluation metric.

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

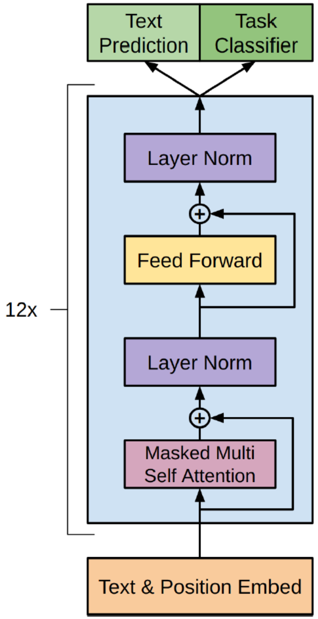


GPT Task performance

Question answering and commonsense reasoning:
RACE contains questions from high-school and middle-school exams.
Story Cloze is story completion

Table 3: Results on question answering and commonsense reasoning, comparing our model with current state-of-the-art methods.. 9x means an ensemble of 9 models.

Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	77.6	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	60.2	50.3	53.3
Finetuned Transformer LM (ours)	**86.5**	**62.9**	**57.4**	**59.0**

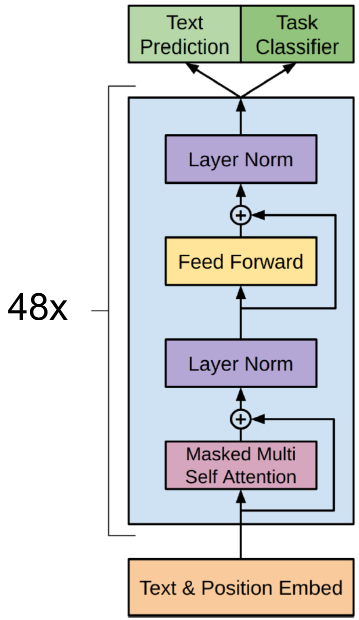


GPT2

A depth-48 version of GPT trained on a massive dataset (WebText):

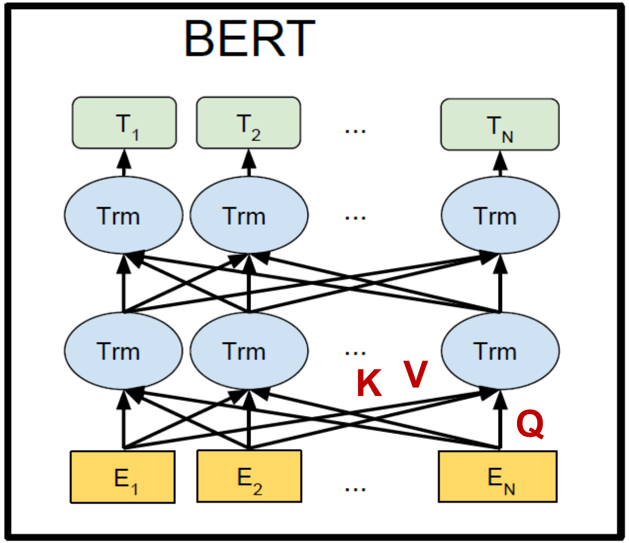
A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.

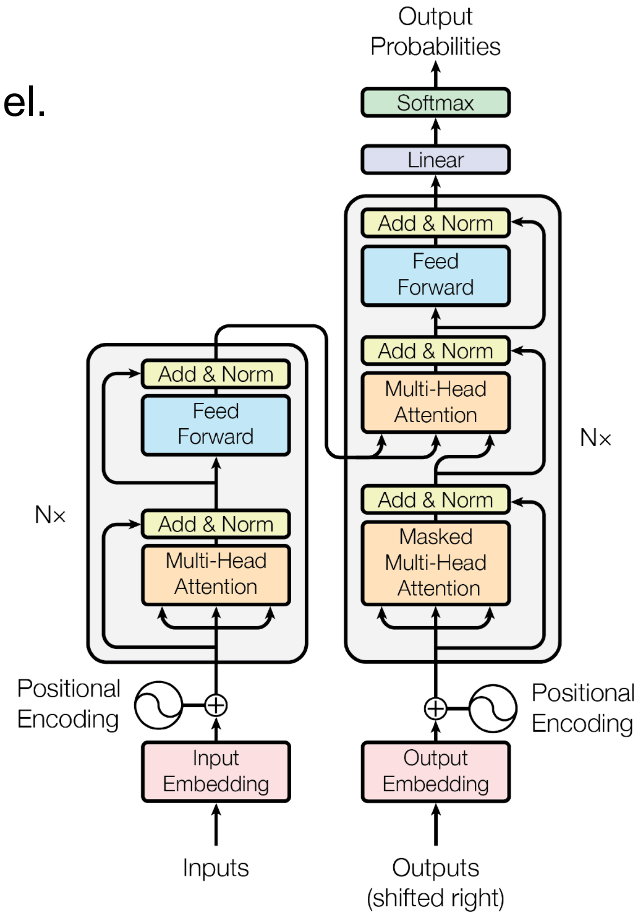
In an email to Ohio news outlets, the U.S. Department of Energy said it is working with the Federal Railroad Administration to find the thief.
“The theft of this nuclear material will have significant negative consequences on public and environmental health, our workforce and the economy of our nation,” said Tom Hicks, the U.S. Energy Secretary, in a statement. “Our top priority is to secure the theft and ensure it doesn’t happen again.”
The stolen material was taken from the University of Cincinnati’s Research Triangle Park nuclear research site, according to a news release from Department officials.
The Nuclear Regulatory Commission did not immediately release any information.
According to the release, the U.S. Department of Energy’s Office of Nuclear Material Safety and Security is leading that team’s investigation.
“The safety of people, the environment and the nation’s nuclear stockpile is our highest priority,” Hicks said. “We will get to the bottom of this and make no excuses.”



BERT

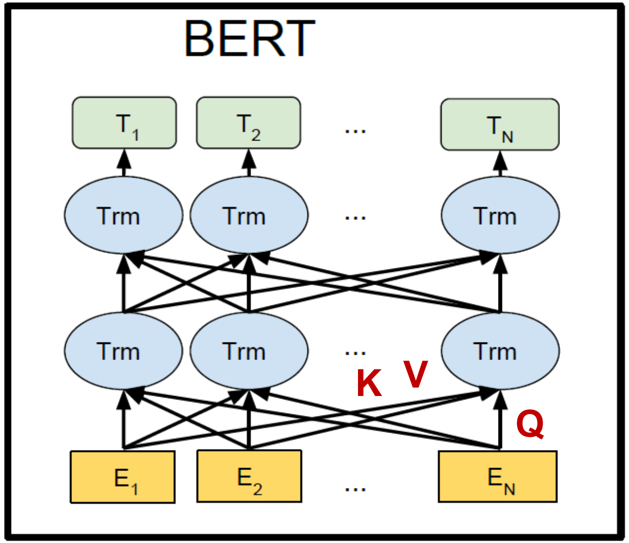
BERT is a bidirectional (Transformer encoder) model.

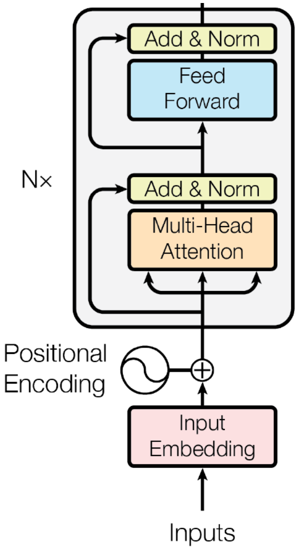




BERT

BERT is a bidirectional (Transformer encoder) model.

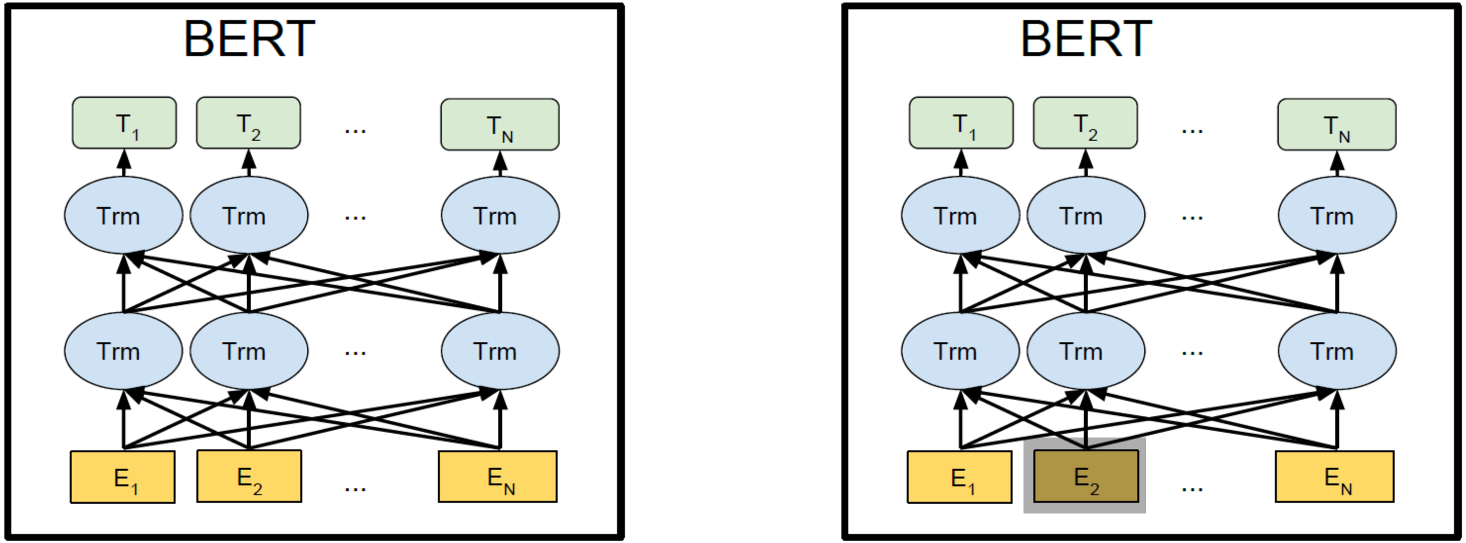




BERT

BERT is trained with two types of loss:

- Word prediction: 15% of input words are removed and then re-predicted



BERT

BERT is trained with two types of loss:

- Next sentence prediction: from an actual corpus of consecutive sentence pairs, create a dataset with 50% real pairs, and 50% “fake” pairs (where the second sentence is a random one).

$$\text{Input} = \texttt{[CLS] the man went to [MASK] store [SEP]}$$
$$\texttt{he bought a gallon [MASK] milk [SEP]}$$
$$\text{Label} = \texttt{IsNext}$$

$$\text{Input} = \texttt{[CLS] the man [MASK] to the store [SEP]}$$
$$\texttt{penguin [MASK] are flight ##less birds [SEP]}$$
$$\text{Label} = \texttt{NotNext}$$

BERT

BERT is trained with two types of loss:

- Next sentence prediction: from an actual corpus of consecutive sentence pairs, create a dataset with 50% real pairs, and 50% “fake” pairs (where the second sentence is a random one).

- This is an example of a very general loss for unsupervised learning called “contrastive loss”. The loss contrasts true positives with “near miss” negatives.

BERT Task specialization

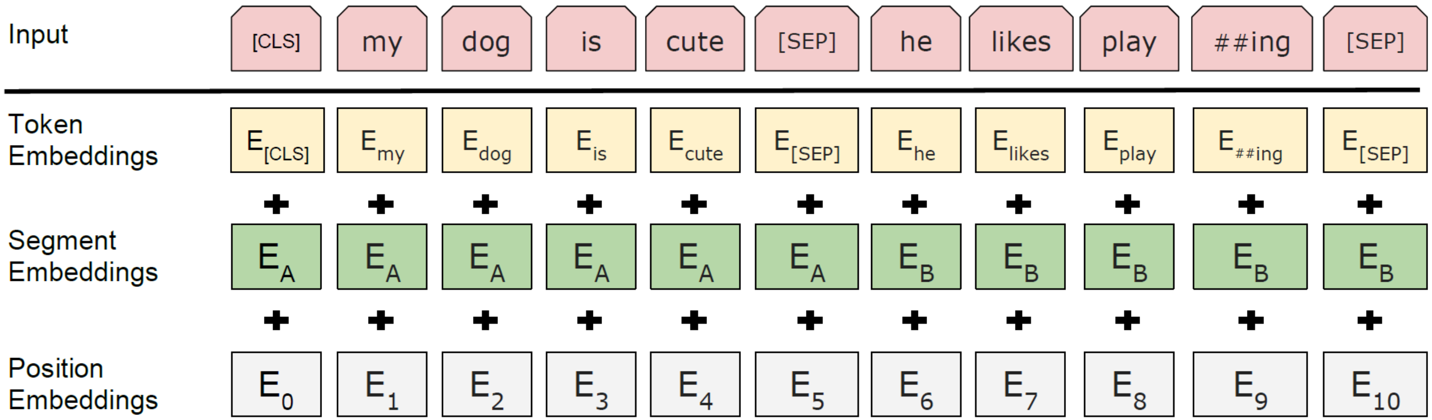
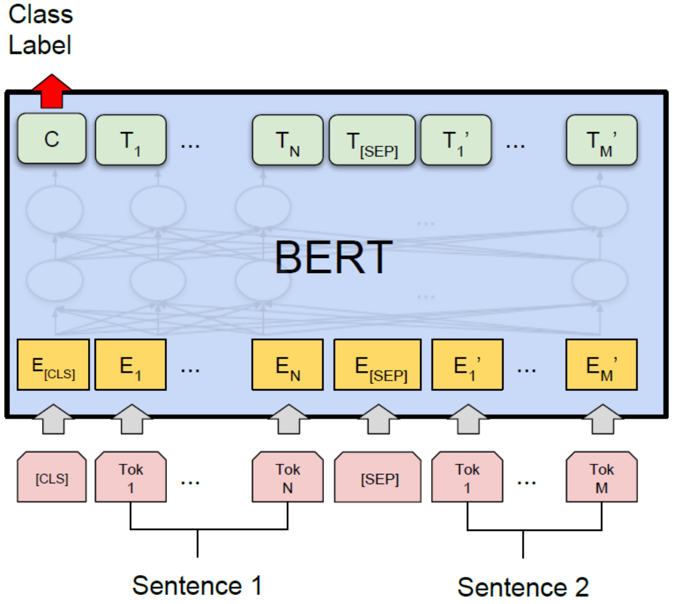
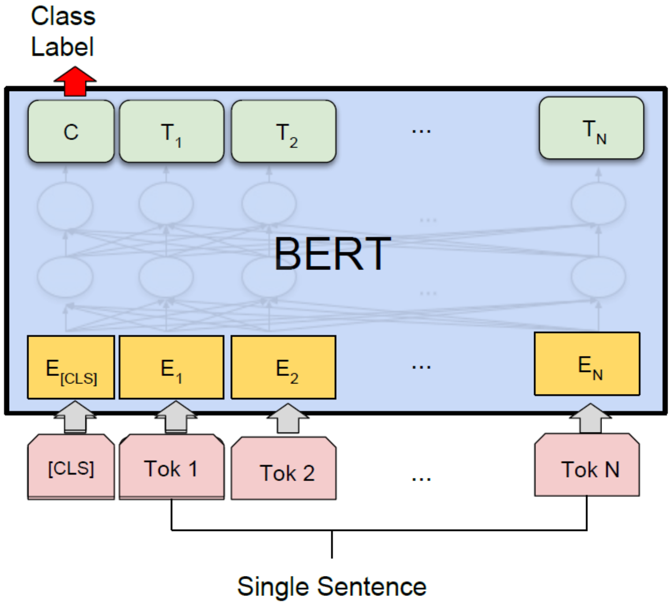


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

BERT Task specialization

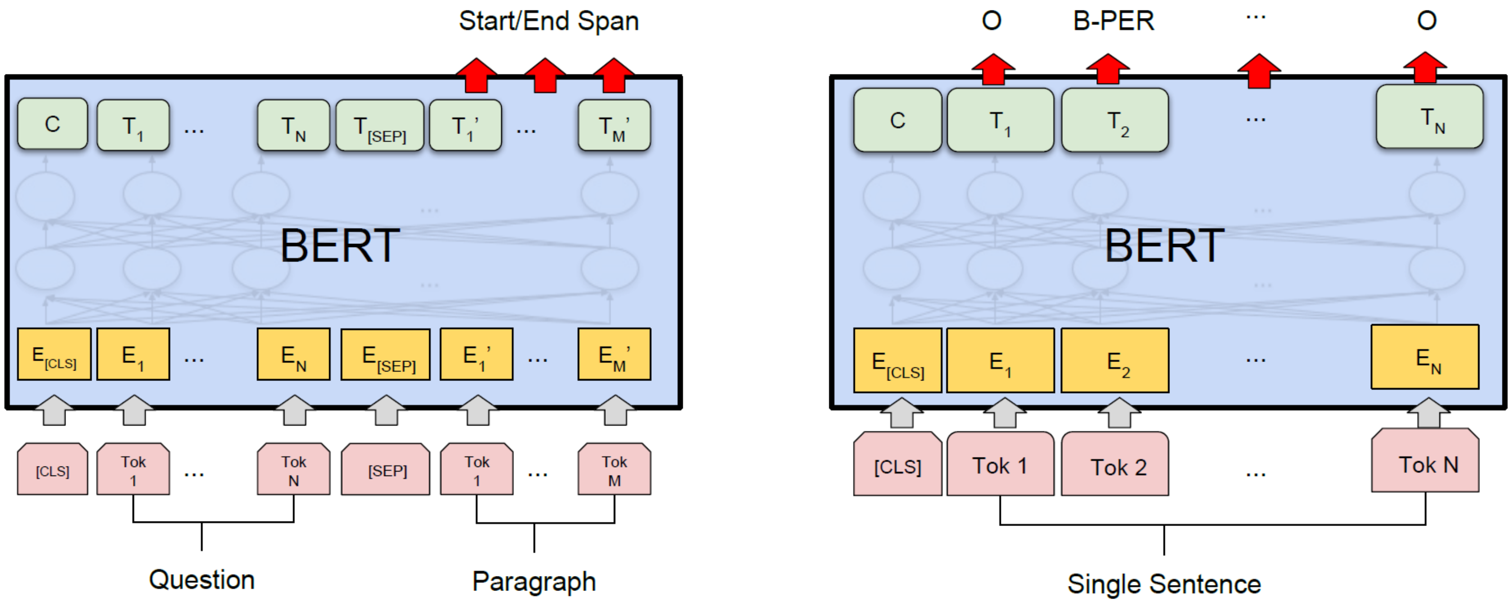


(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



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

BERT Task specialization



(c) Question Answering Tasks:
SQuAD v1.1

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

BERT Performance

BERT Base: L=12, H=768, A=12, total param =110M
BERT Large: L=24, H=1024, A=16, total param=340M

L=number of layers, H=model dimension, A=number of multi-attention heads

Glue tasks

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
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**

BERT Performance

SQuAD

System	Dev EM	Dev F1	Test EM	Test F1
Leaderboard (Oct 8th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
#1 Single - nlnet	-	-	83.5	90.1
#2 Single - QANet	-	-	82.5	89.3
Published				
BiDAF+ELMo (Single)	-	85.8	-	-
R.M. Reader (Single)	78.9	86.3	79.5	86.6
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Dialog

DSTC = Dialog System Technology Challenge:

- Sentence selection
- Sentence generation
- Audio-visual scene-aware dialog

Sample tasks for DSTC

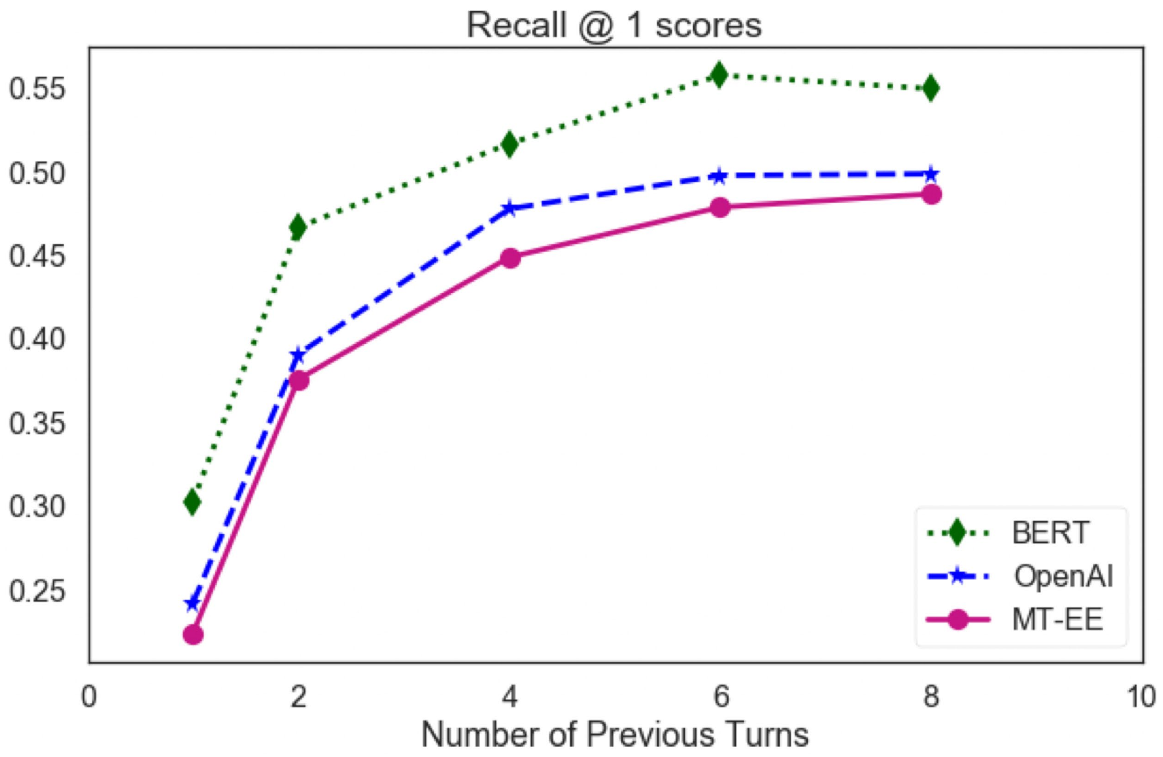
ADVISOR | Hi! What can I help you with?
STUDENT | Hello! I'm trying to schedule classes for next semester. Can you help me?
STUDENT | Hardware has been an interest of mine.
STUDENT | But I don't want too hard of classes
ADVISOR | So are you interested in pursuing Electrical or Computer Engineering?
STUDENT | I'm undecided
STUDENT | I enjoy programming but enjoy hardware a little more.
ADVISOR | Computer Engineering consists of both programming and hardware.
ADVISOR | I think it will be a great fit for you.
STUDENT | Awesome, I think that's some good advice.
STUDENT | What classes should I take to become a Computer Engineer?
ADVISOR | You haven't taken EECS 203, 280, and 270, so it may be in your best interest to take one or two of those classes next semester
STUDENT | Ok. Which of those is in the morning. I like morning classes

Sample tasks for DSTC

[13:11] <user_1> anyone here know memcached?
[13:12] <user_1> trying to change the port it runs on
[13:12] <user_2> user_1: and ?
[13:13] <user_1> user_2: I'm not sure where to look
[13:13] <user_1> !
[13:13] <user_2> user_1: /etc/memcached.conf ?
[13:13] <user_1> haha
[13:13] <user_1> user_2: oh yes, it's much simpler than I thought
[13:13] <user_1> not sure why, I was trying to work through the init.d stuff

Should also use external reference information from unix man pages.

DSTC7 results



CoQA

https://stanfordnlp.github.io/coqa/

CoQA contains 127,000+ questions with answers collected from 8000+ conversations. Each conversation is collected by pairing two crowdworkers to chat about a passage in the form of questions and answers.

CoQA Leaderboard 3/20/19

Leaderboard

Rank	Model	In-domain	Out-of-domain	Overall
	Human Performance *Stanford University* (Reddy et al. '18)	89.4	87.4	88.8
1 Jan 25, 2019	BERT + MMFT + ADA (ensemble) *Microsoft Research Asia*	**87.5**	**85.3**	**86.8**
2 Jan 21, 2019	BERT + MMFT + ADA (single model) *Microsoft Research Asia*	86.4	81.9	85.0
3 Jan 03, 2019	BERT + Answer Verification (single model) *Sogou Search AI Group*	83.8	80.2	82.8
4 Jan 06, 2019	BERT with History Augmented Query (single model) *Fudan University NLP Lab*	82.7	78.6	81.5
5 Jan 31, 2019	BERT Large Finetuned Baseline (single model) *Anonymous*	82.6	78.4	81.4
6 Jan 21, 2019	BERT Large Augmented (single model) *Microsoft Dynamics 365 AI Research*	82.5	77.6	81.1
7 Dec 12, 2018	D-AoA + BERT (single model) *Joint Laboratory of HIT and iFLYTEK Research*	81.4	77.3	80.2

Takeaways

- Pre-training language models on very large corpora improves the state-of-the-art for multiple NLP tasks (ELMo, GPT, BERT).
- Transformer designs (GPT and BERT) have superseded RNN designs (ELMo).
- Single-task execution involves input encoding, a small amount of output hardware, and fine-tuning.
- BiDirectional encoder models (BERT) do better than generative models (GPT) at non-generation tasks, for comparable training data/model complexity.
- Generative models have training efficiency and scalability advantages that may make them ultimately more accurate. They can also solve entirely new kinds of task that involve text generation.