NLP: Advanced Architectures

Yuanzhi Zhu
SiDNN

26/Apr/2022

Content

- Transformer
- GPT
- BERT

Content

- Transformer
- GPT
- BERT

NLP: Seq2Seq Problems

Seq2Seq Problems: More than word embeddings

Machine Translation

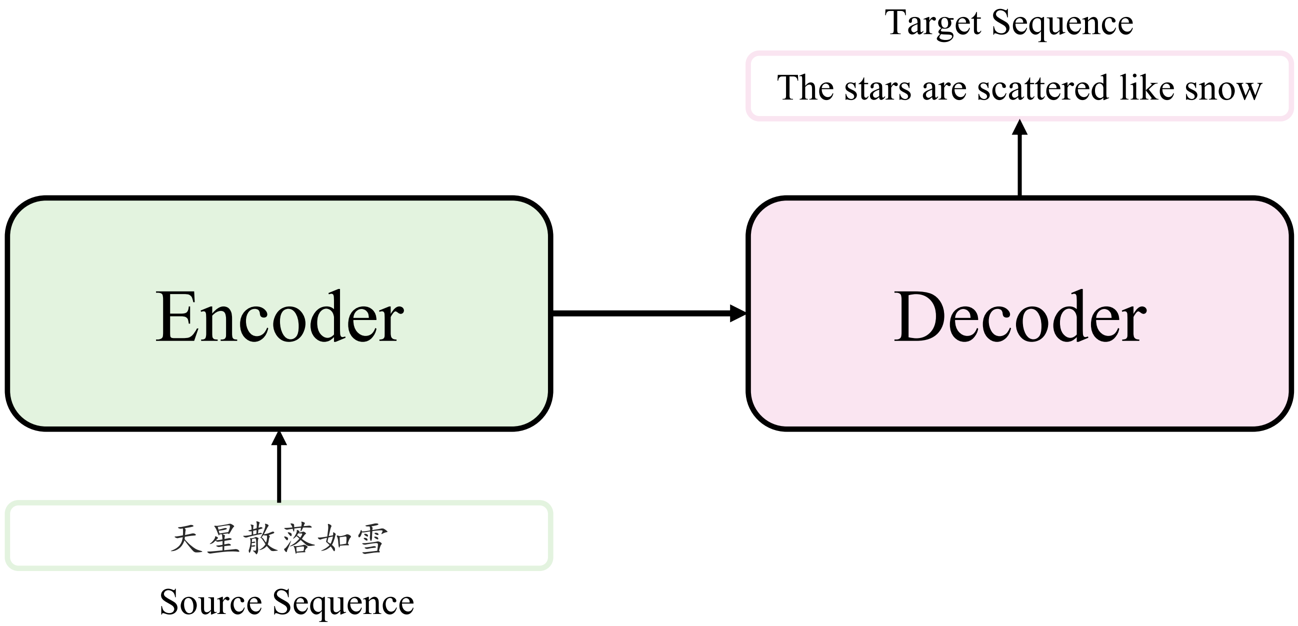
天星散落如雪
The stars in the sky are scattered like snow

Question Answering

How are you?
I am good

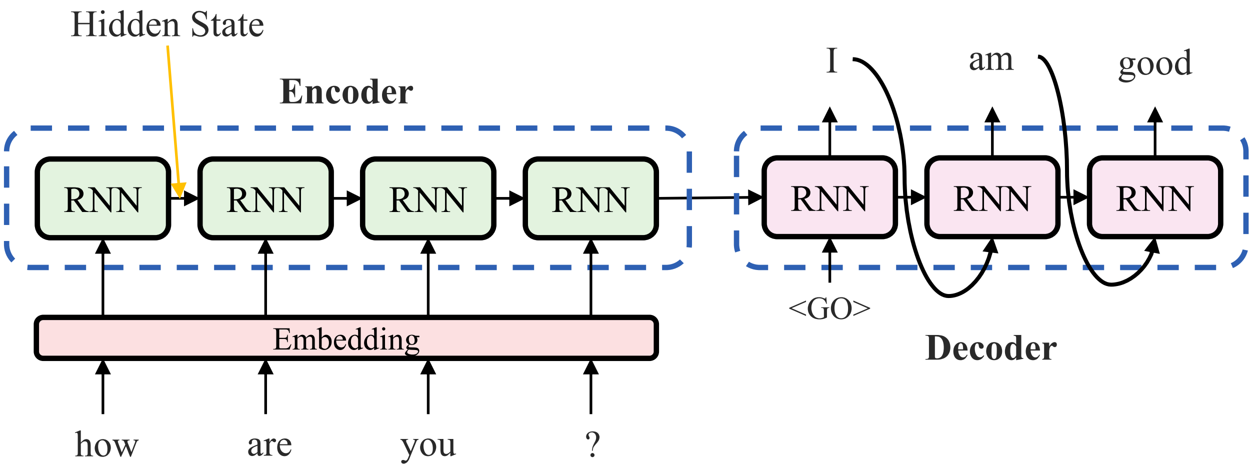
NLP: Seq2Seq Model

General Encoder Decoder Architecture



NLP: Seq2Seq Model

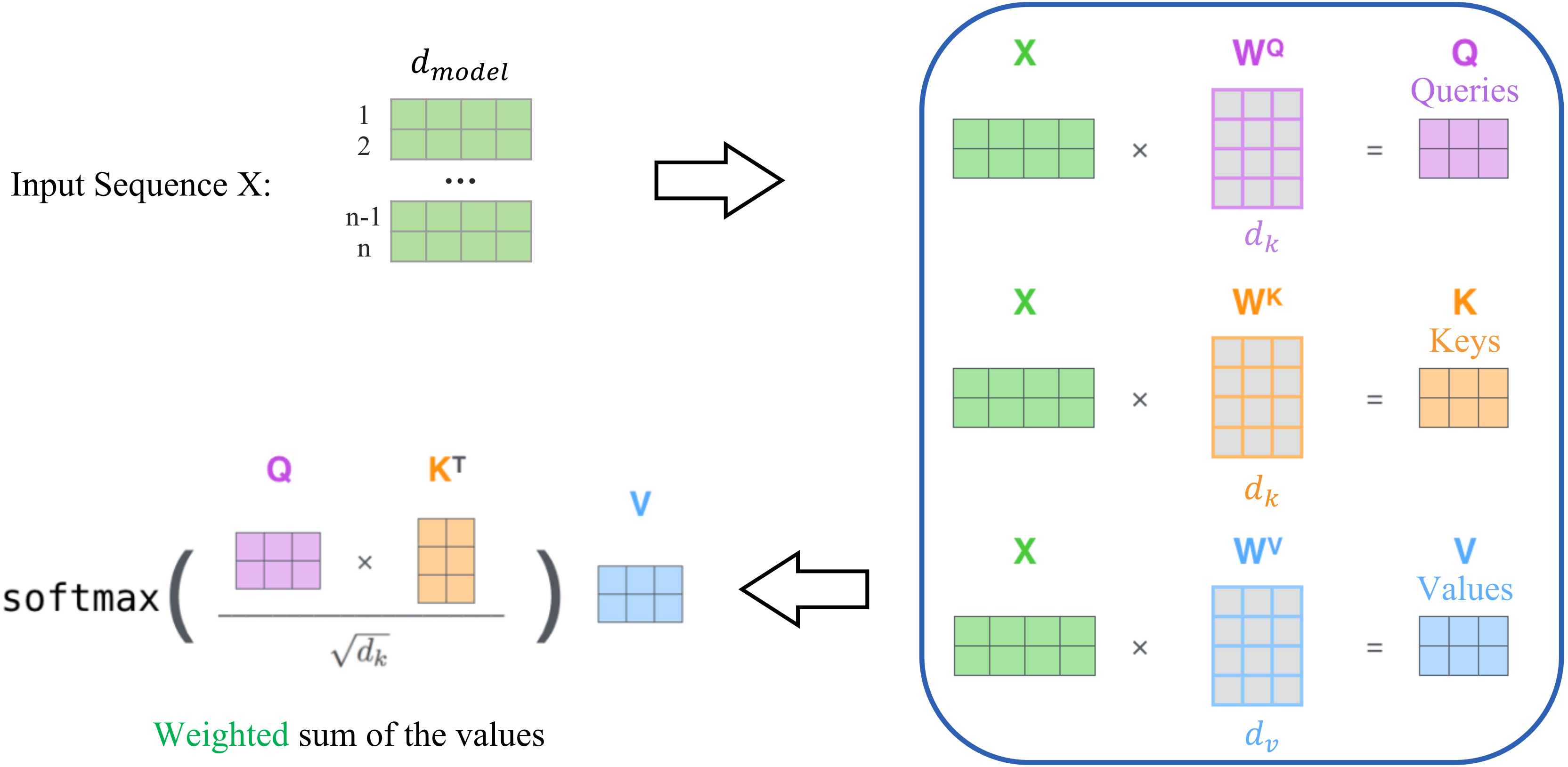
RNN Encoder Decoder Architecture



Limitations:

- Long Sequences
- No Parallelization

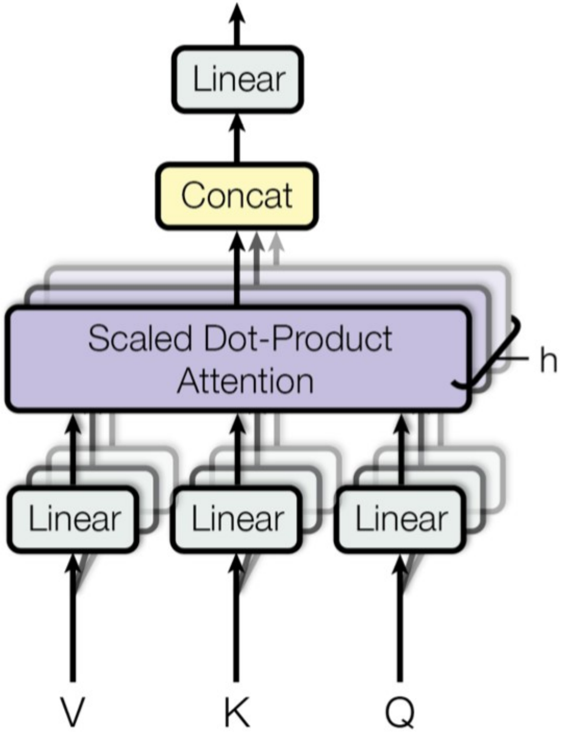
Self-Attention: Better Info Mixer



Weighted sum of the values

Multi-head Attention ← Multi-channel CNN

All heads are initialized randomly
to learn different attentions

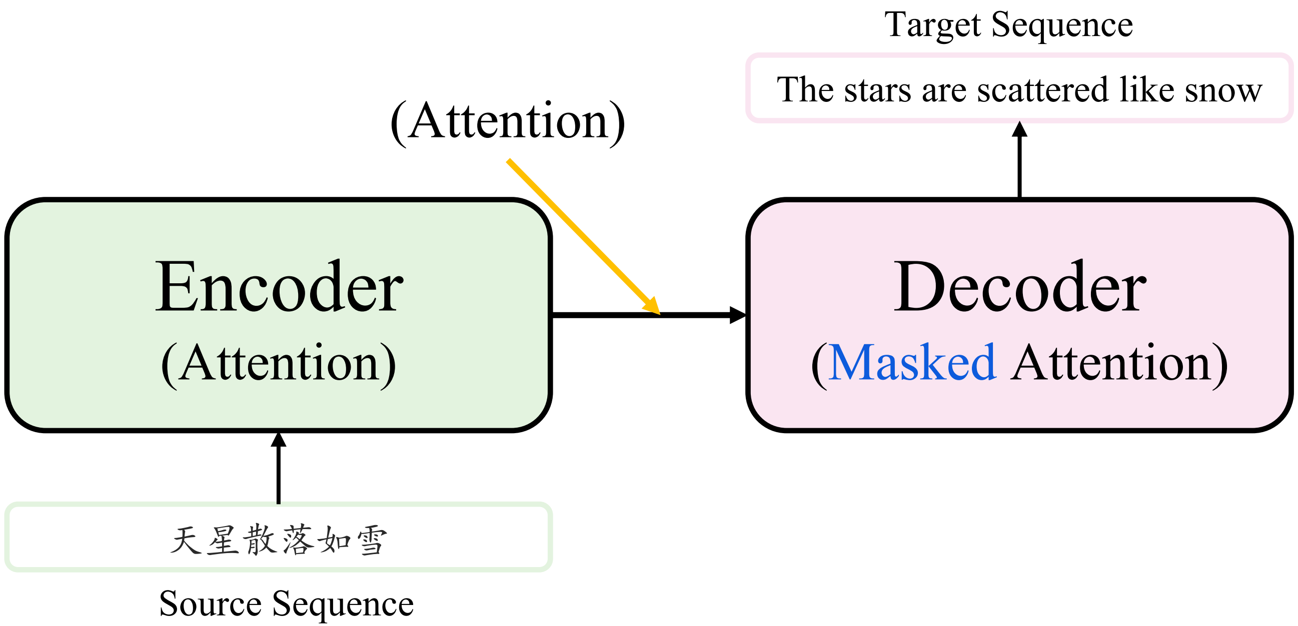


$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, \ldots, \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$\text{output dimension} = \mathcal{R}^{n \times d_{model}}$$

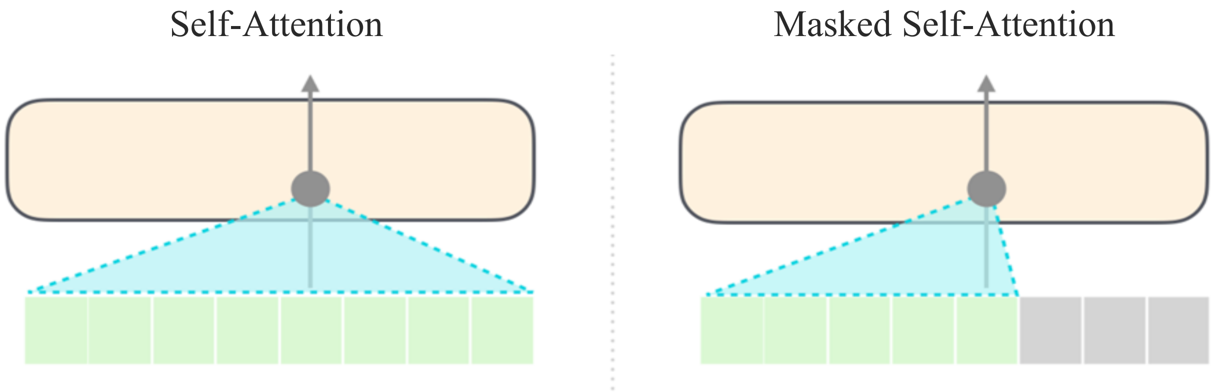
Seq2Seq Model with Attention

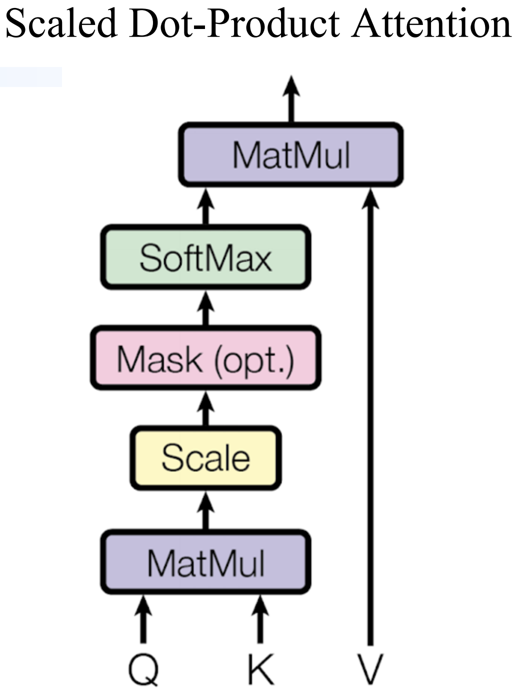
Encoder Decoder Architecture Using Attention



Mask: In Decoder

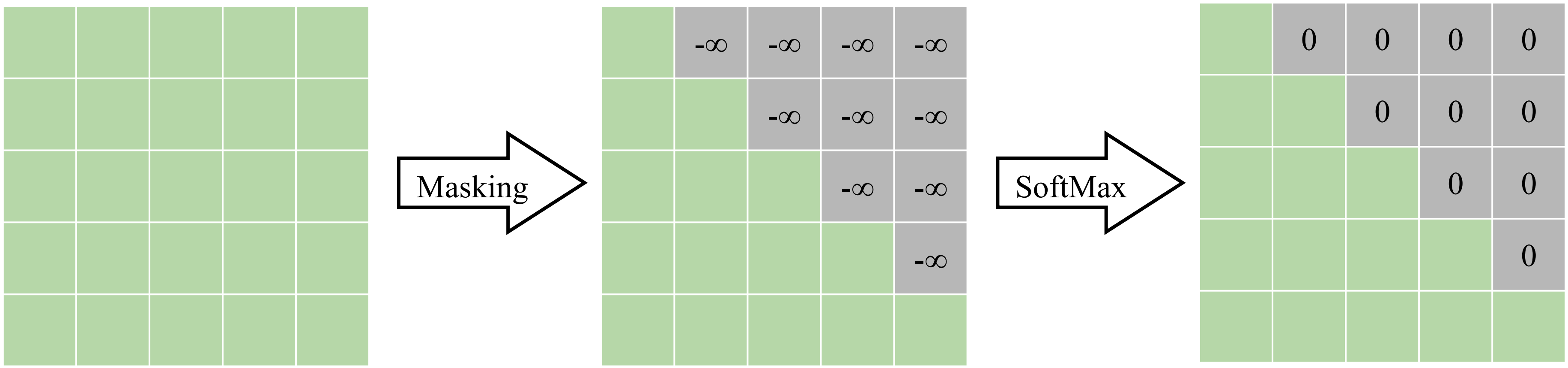
Why? Prediction should only rely on all the previous tokens





Mask: In Decoder

Masked attention

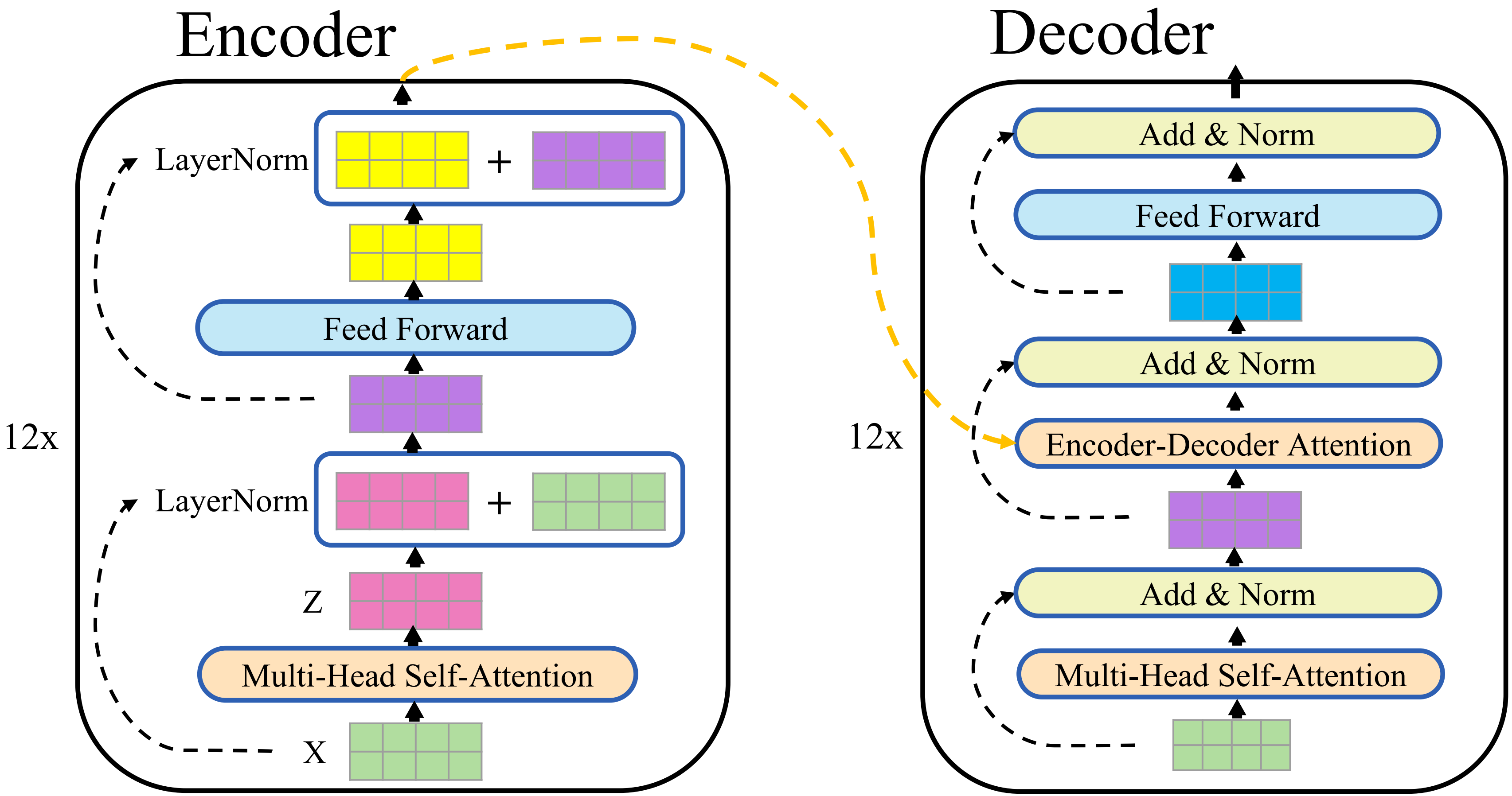


Product of Q*K (scale)

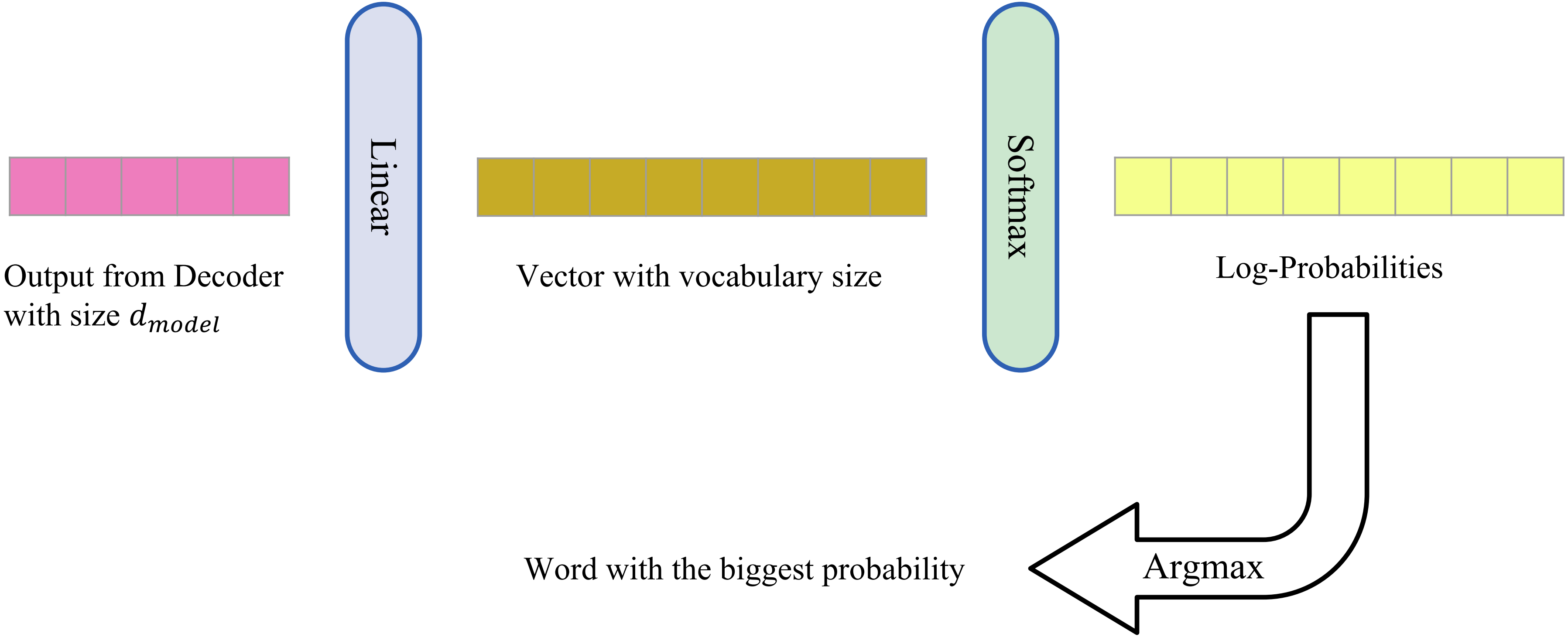
After Masking

Weight of Attention

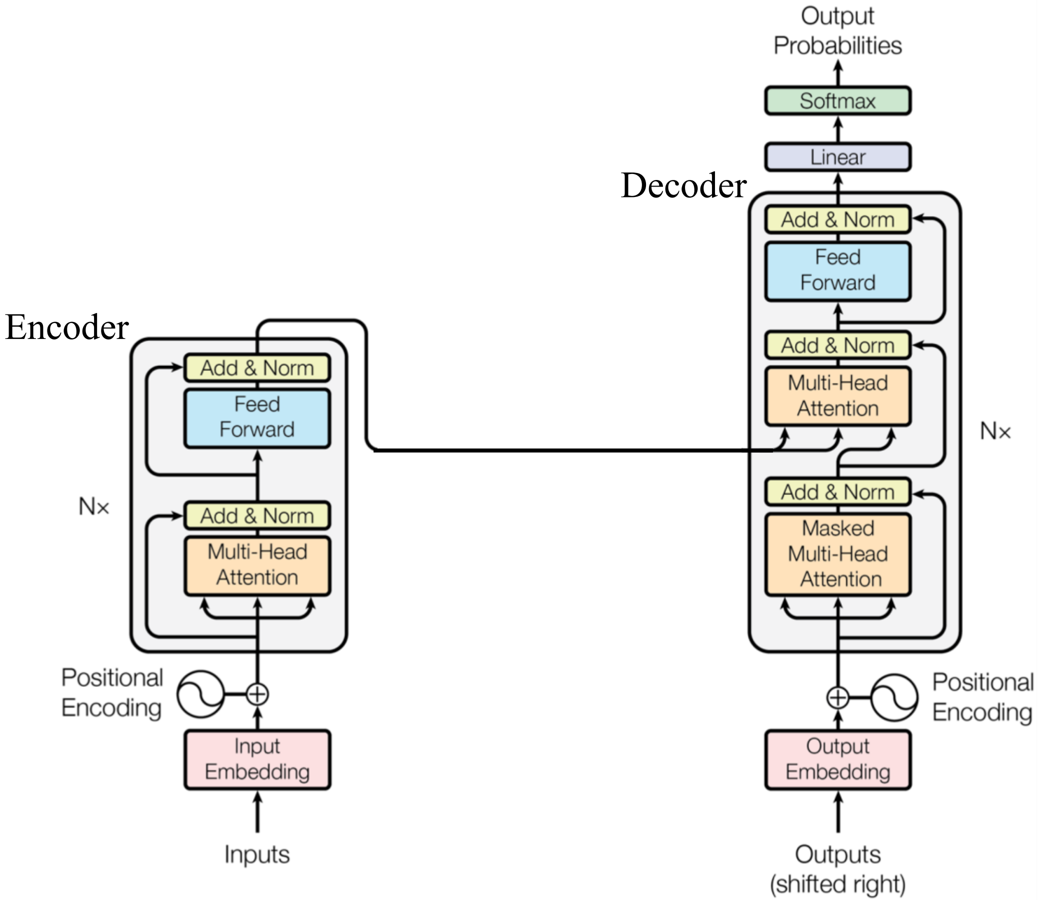
Encoder & Decoder in Detail



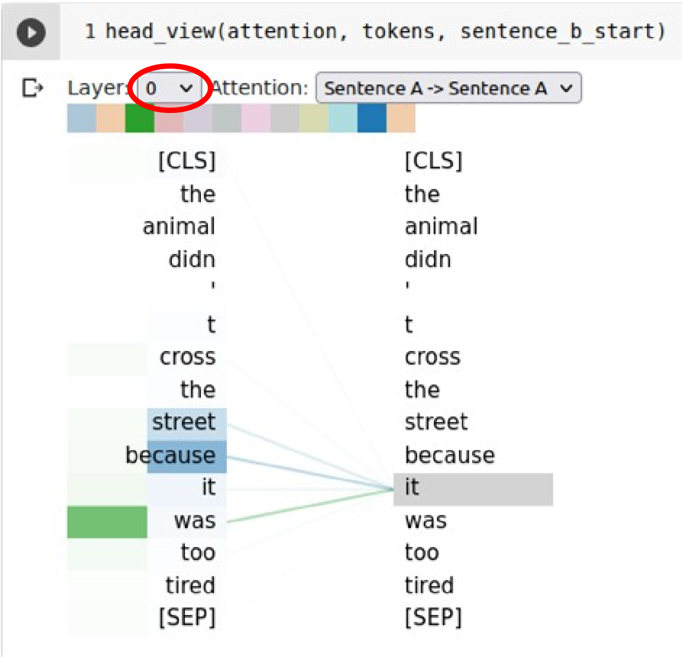
The Output Layer: (Language Generation Model)

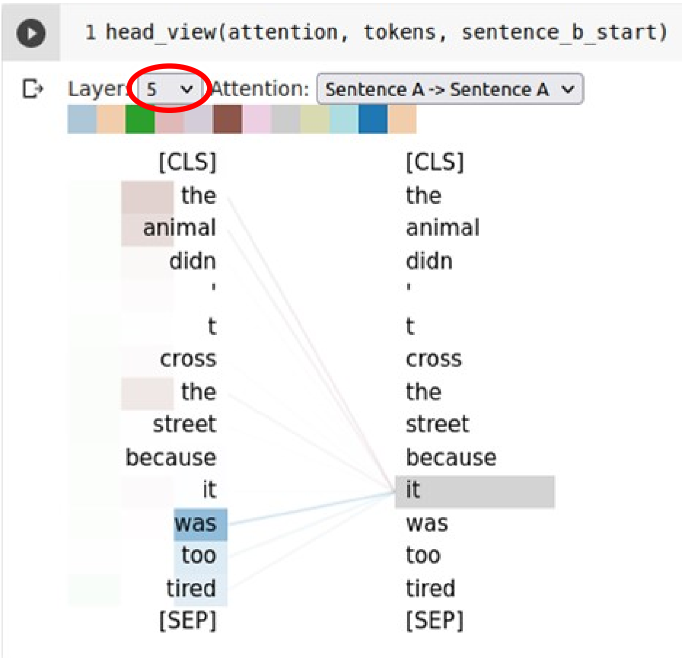


Attention → Transformer



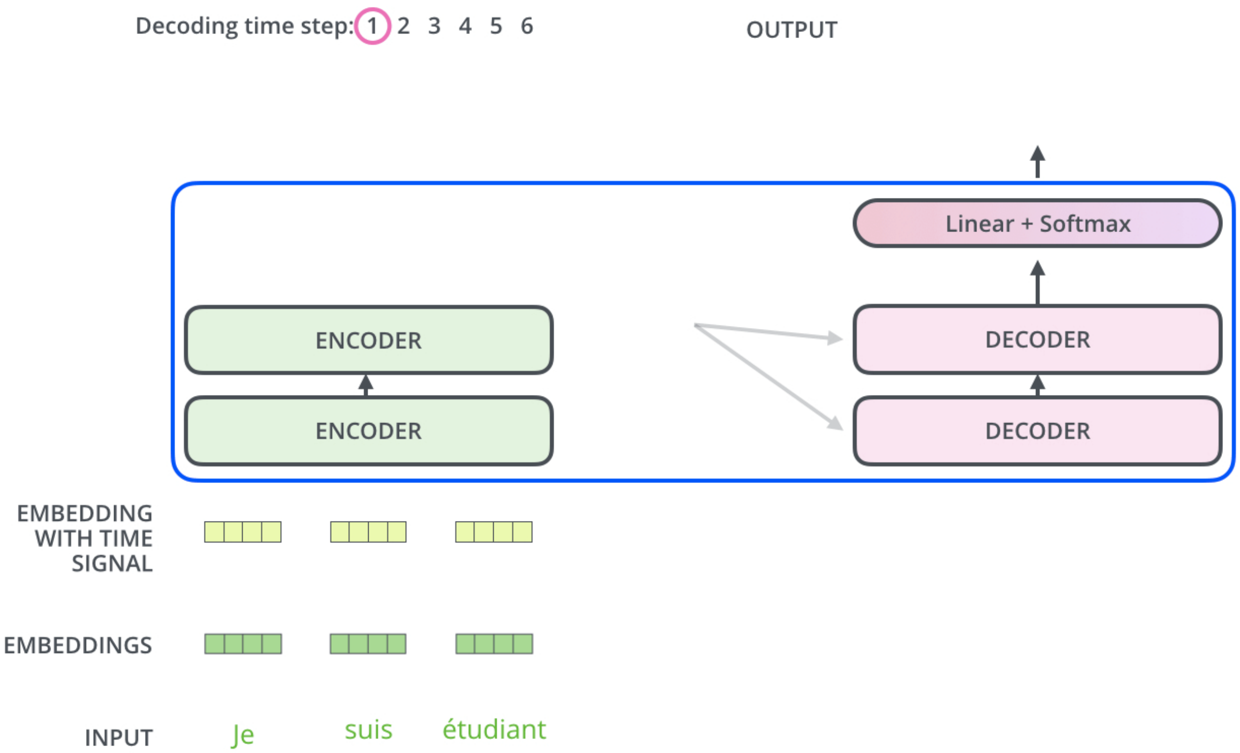
Self-Attention: Example





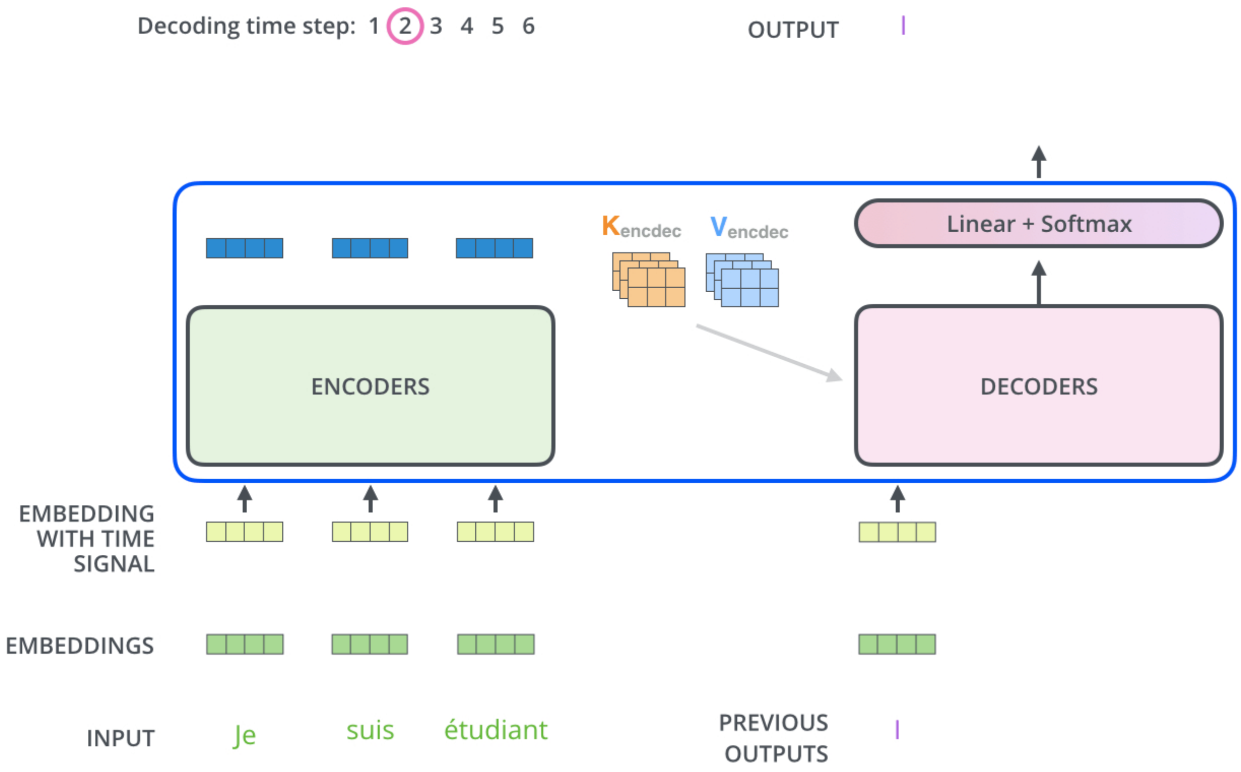
https://github.com/jessevig/bertviz

Encoder: Animation



http://jalammar.github.io/illustrated-transformer

Decoder: Animation



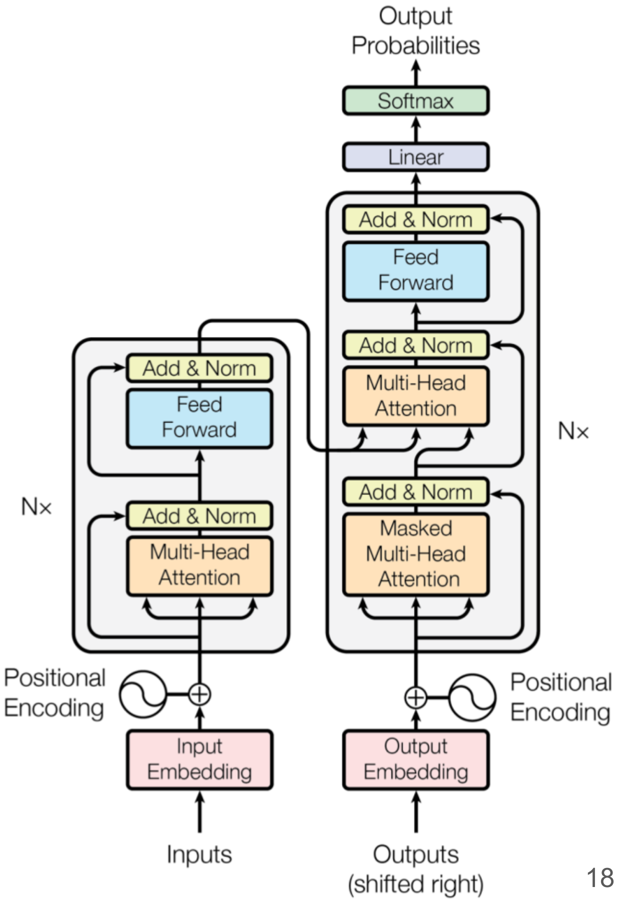
http://jalammar.github.io/illustrated-transformer

Transformer: Review

Attention is all you need (beyond RNN)

n: length of sequence d: dimension of word embedding

Layer Type	Complexity/Layer	Sequential Operation	Maximum Path Length
Self-Attention	$O(n^2d)$	$O(1)$	$O(1)$
Recurrent	$O(nd^2)$	$O(n)$	$O(n)$
Convolutional	$O(knd)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(rnd)$	$O(1)$	$O(n/r)$

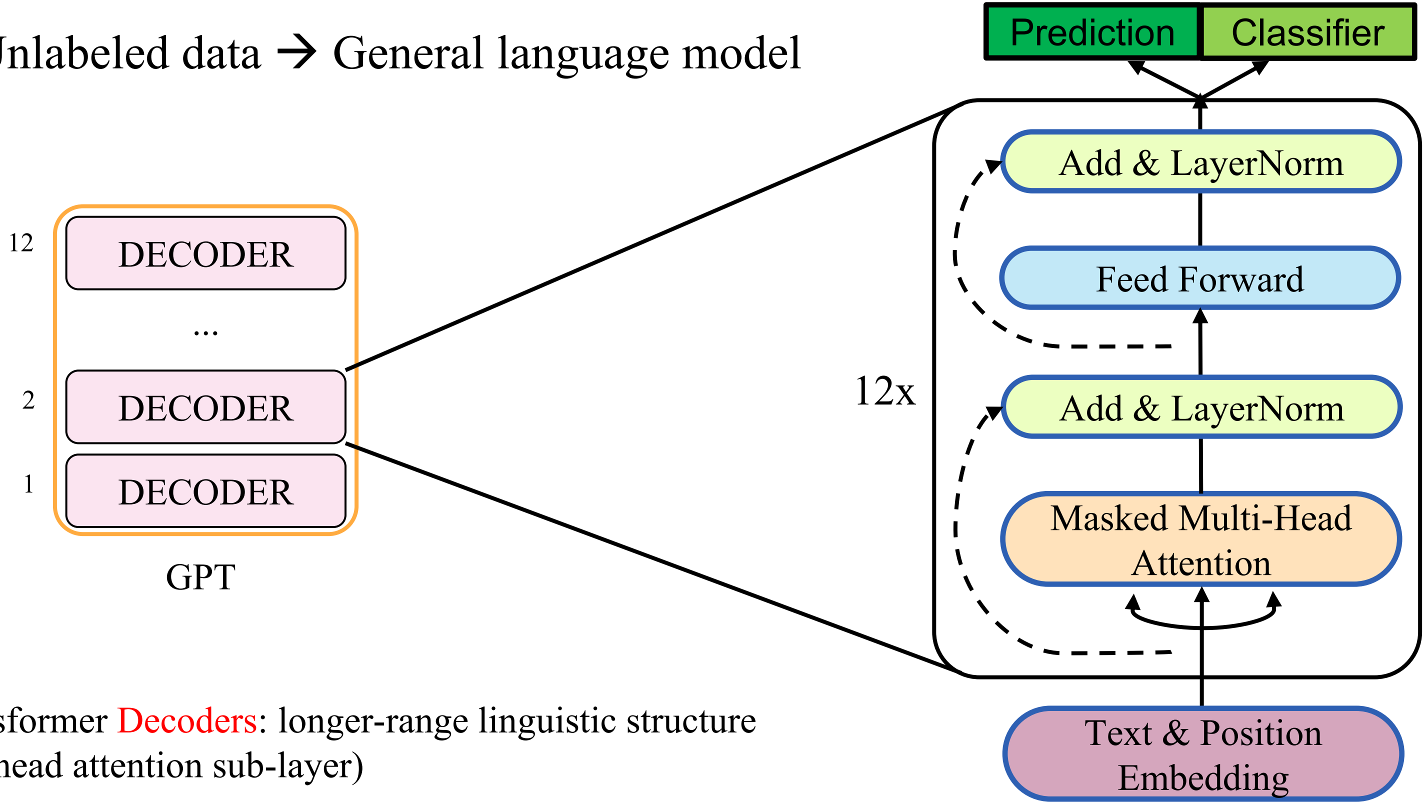


Content

- Transformer
- GPT
- BERT

GPT: Generative Pre-Training*

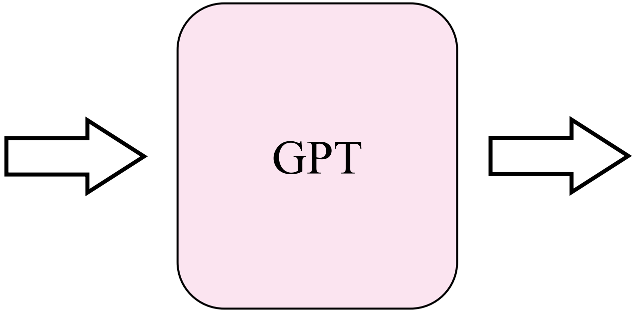
Motivation: Unlabeled data → General language model

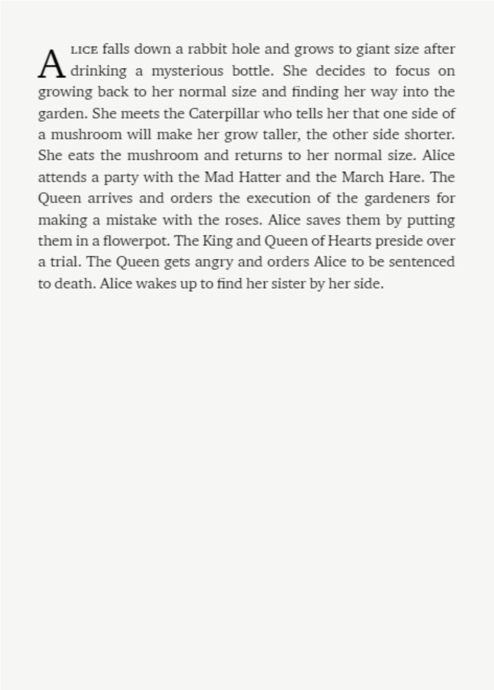


Multi-layer Transformer Decoders: longer-range linguistic structure
(Only one multi-head attention sub-layer)

*Radford, Alec, et al. "Improving language understanding by generative pre-training." (2018).

GPT Example: Summarization




ALICE falls down a rabbit hole and grows to giant size after drinking a mysterious bottle. She decides to focus on growing back to her normal size and finding her way into the garden. She meets the Caterpillar who tells her that one side of a mushroom will make her grow taller, the other side shorter. She eats the mushroom and returns to her normal size. Alice attends a party with the Mad Hatter and the March Hare. The Queen arrives and orders the execution of the gardeners for making a mistake with the roses. Alice saves them by putting them in a flowerpot. The King and Queen of Hearts preside over a trial. The Queen gets angry and orders Alice to be sentenced to death. Alice wakes up to find her sister by her side.

[Summarizing Books with Human Feedback (openai.com)]

GPT: Pre-Training

Two stages semi-supervised training

- Unsupervised Language Modelling (pre-training)

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

I am good
argmax $P(u_i | \text{I, am}; \Theta) = \text{good}$

$$\begin{aligned} h_0 &= UW_e + W_p \\ h_l &= \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n] \\ P(u) &= \texttt{softmax}(h_n W_e^T) \end{aligned}$$

GPT: Fine-Tuning

Two stages semi-supervised training

- Supervised fine-tuning

$$P(y|x^1,\ldots,x^m) = \texttt{softmax}(h_l^m W_y).$$

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1,\ldots,x^m).$$

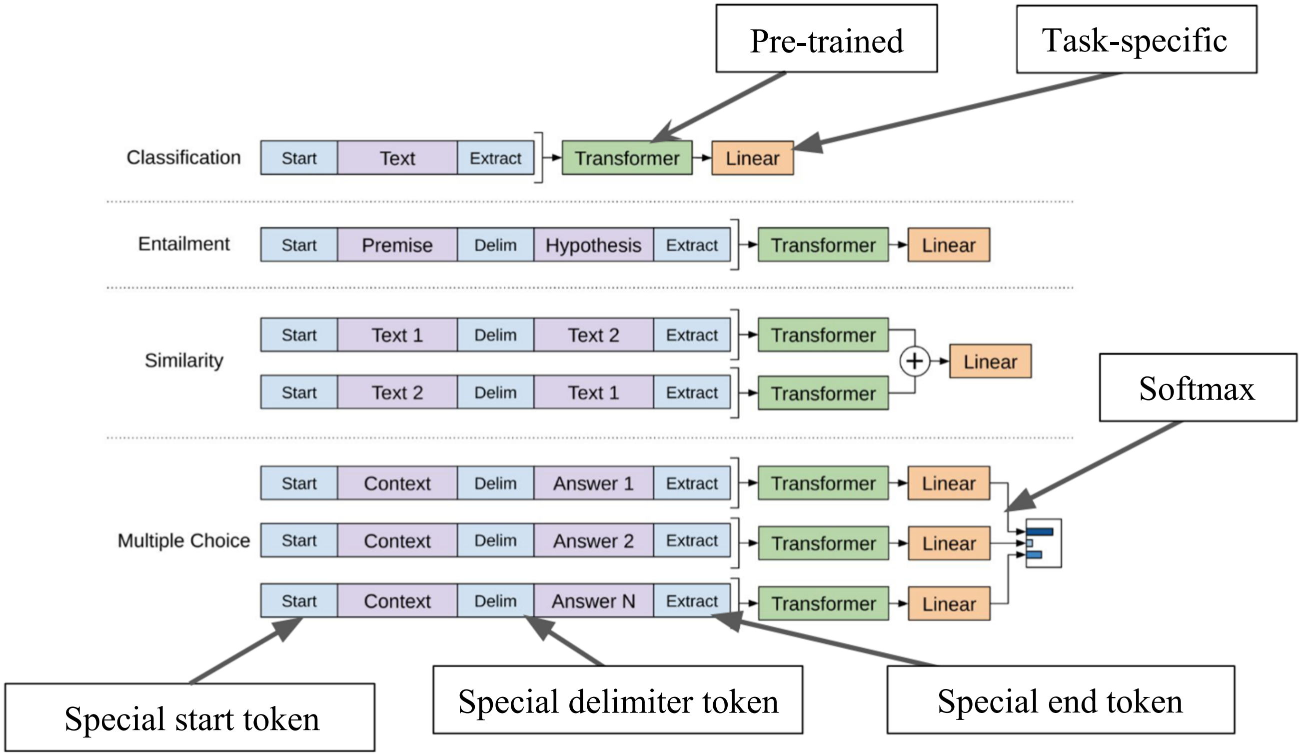
X: I am good
y: positive

Including language modeling as an auxiliary objective

$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

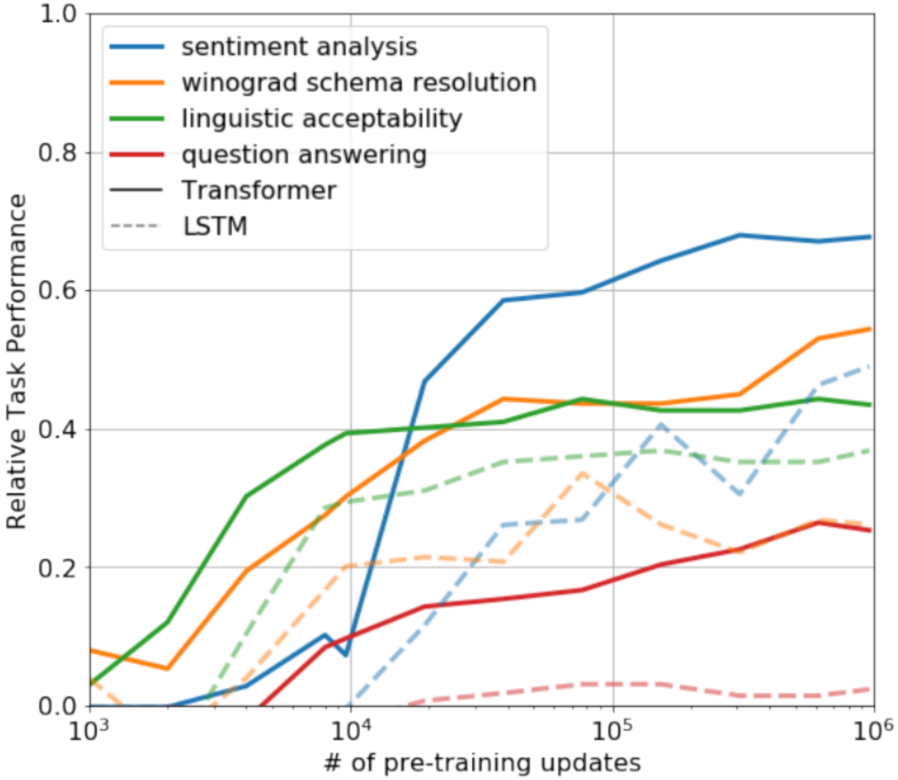
GPT: Task-Specific Input Transformations

GPT is SOTA: achieves new state-of-the-art results in 9 out of the 12 datasets

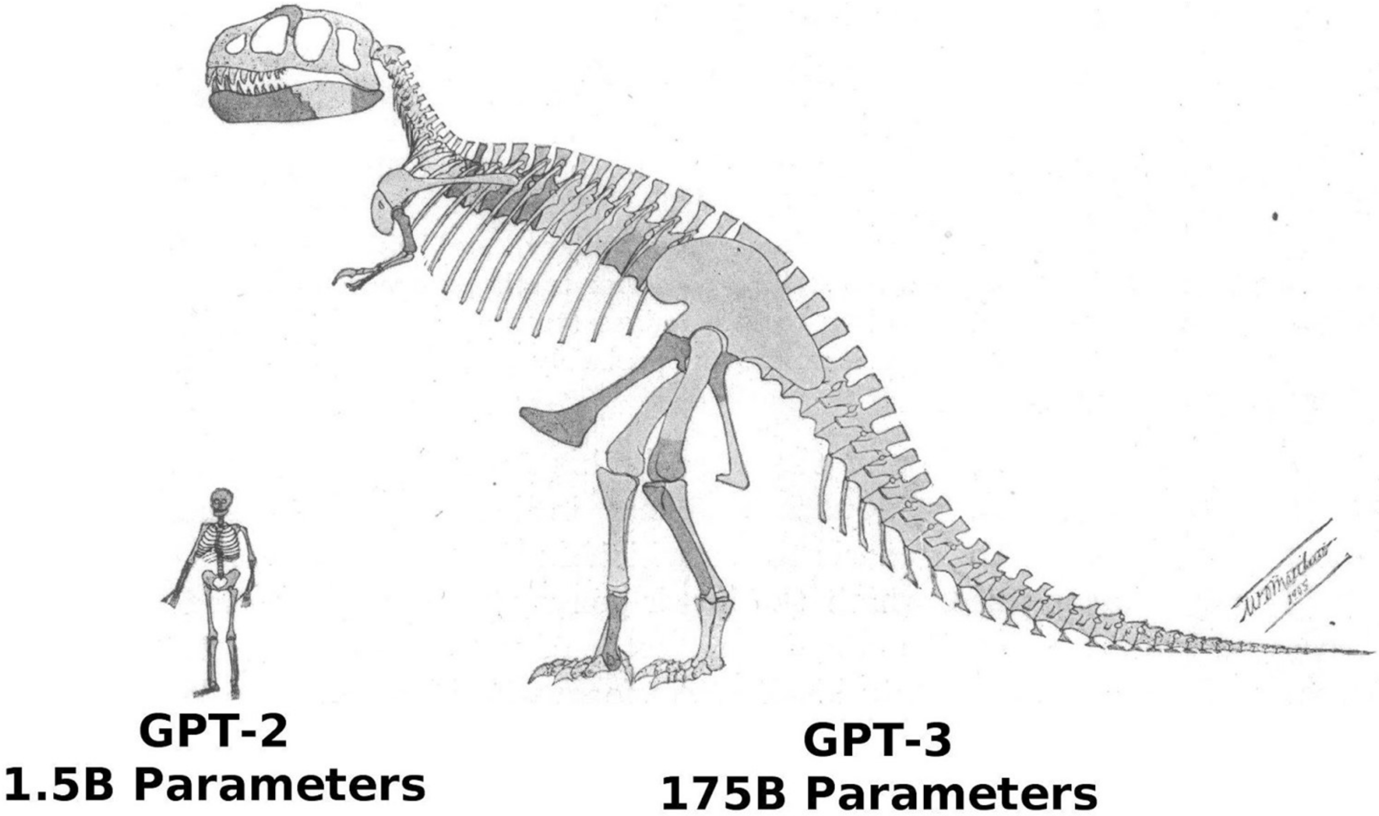


GPT: Experiment & Results

Zero-shot Behaviors



GPT2 and GPT3

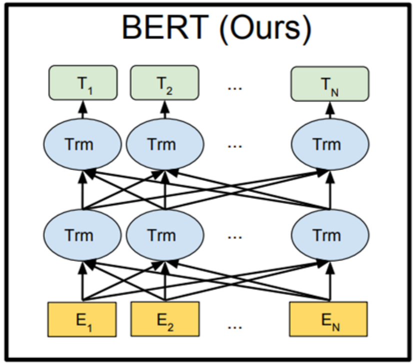


Content

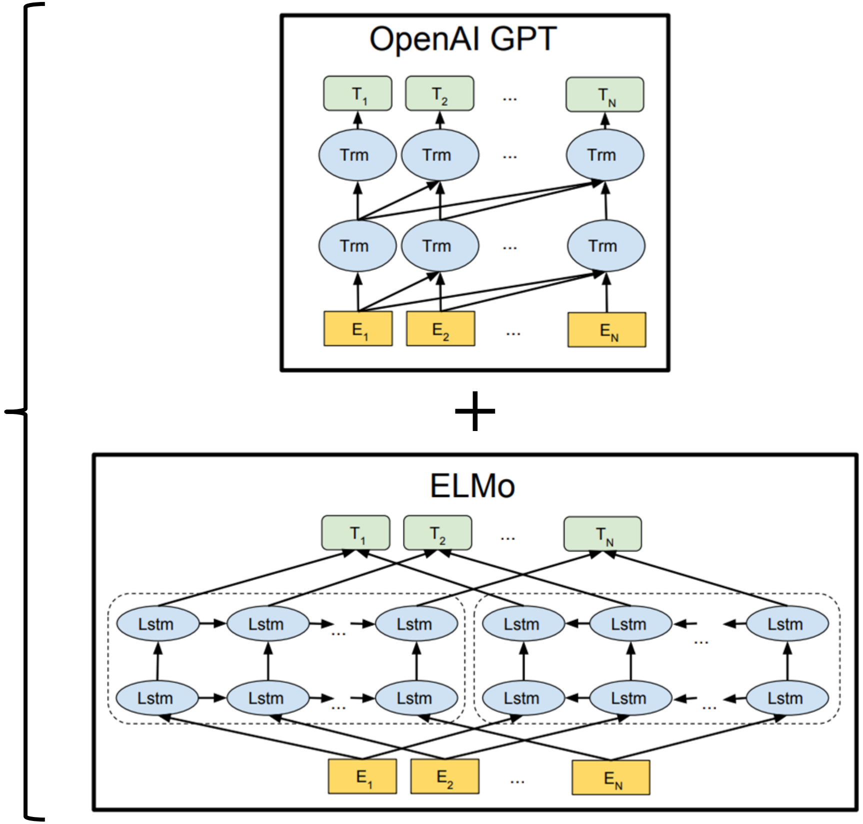
- Transformer
- GPT
- BERT

BERT: Bidirectional Encoder Representations from Transformers*

Motivation: ELMo + GPT

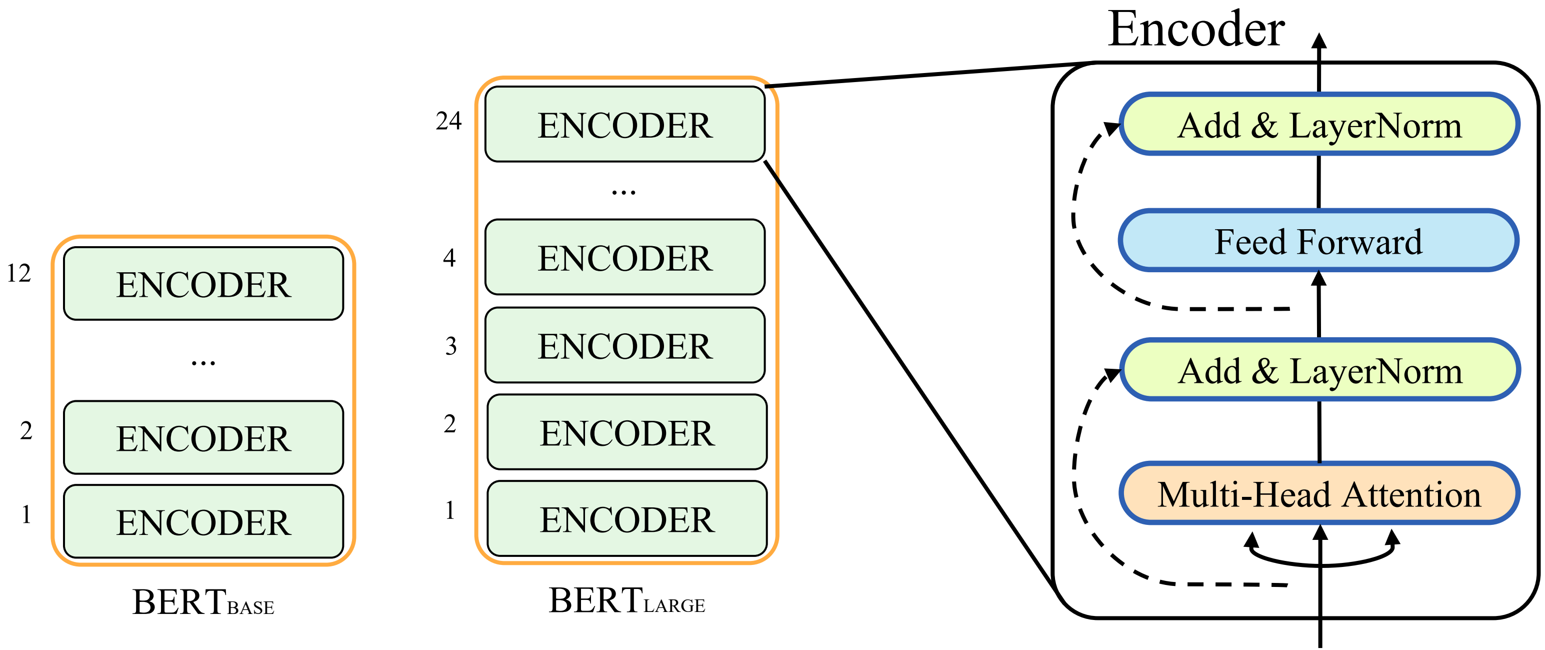


=



*Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).

BERT: Bidirectional Encoder Representations from Transformers

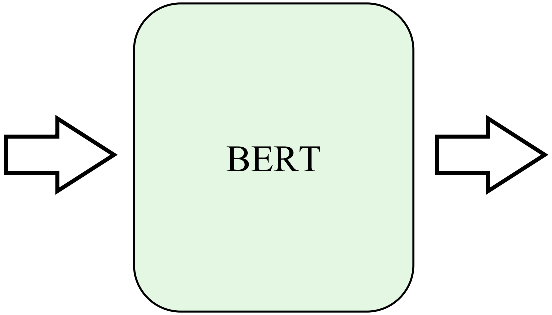


Multi-layer bidirectional Transformer Encoders (not masked)

BERT Example: Question Answering

Question = "What are some example applications of BERT?"

Passage = "…BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, **such as question answering and language inference**, without substantial task-specific architecture modifications."



Answer: *question answering* and *language inference*

Pre-Training Task #1: Masked LM (MLM)

Standard bidirectional conditioning would allow each word to indirectly “see itself” in a multi-layered context

→ Masking 15% of the input tokens at random, then predicting only those masked tokens

$$L(\mathcal{U}) = \sum_{i \in I} \log P(u_i | u_{j \notin I}; \Theta)$$

Downside: Mismatch between pre-training and fine-tuning, since the [MASK] token is never seen during fine-tuning.

My dog is hairy. → choose hairy

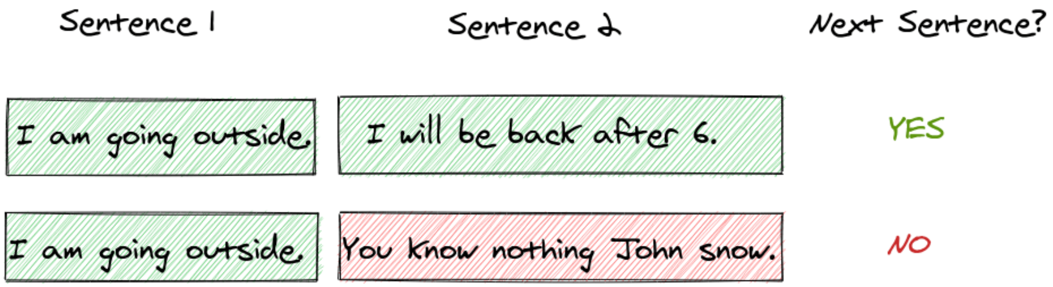
- 80% of time: my dog is [MASK]
- 10% of time: my dog is apple
- 10% of time: my dog is hairy

Pre-Training Task #2: Next Sentence Prediction (NSP)

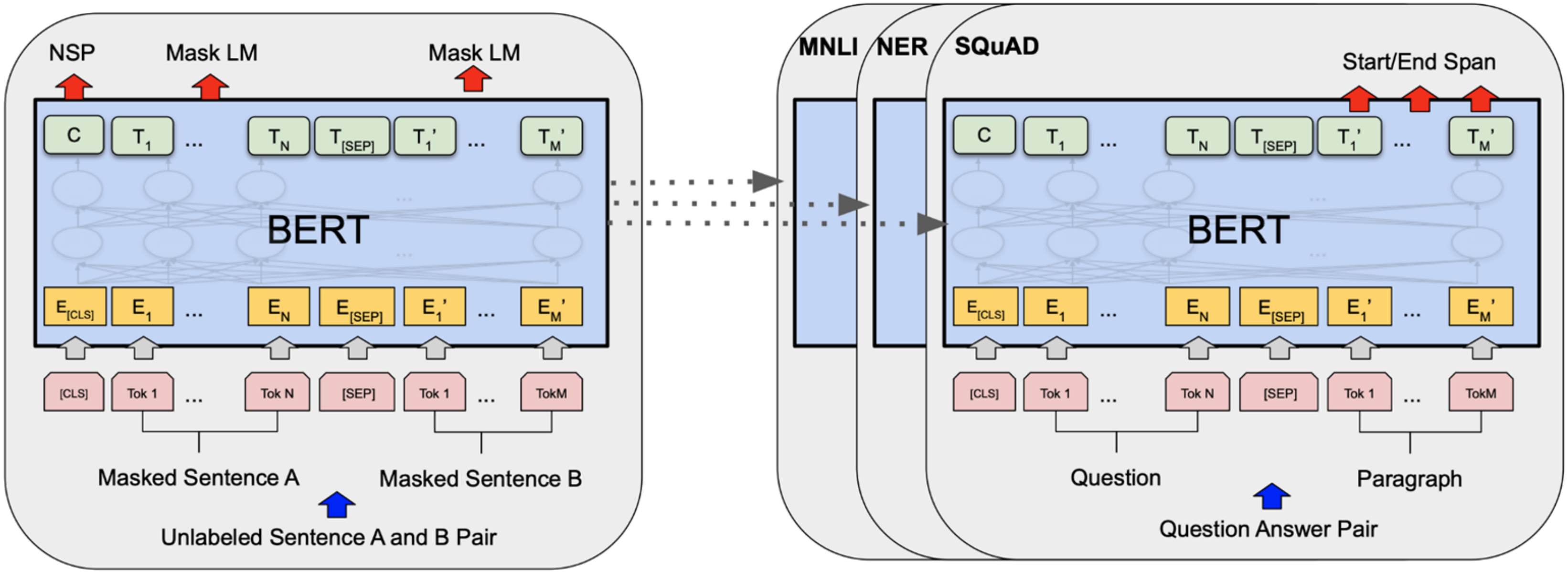
Question Answering / Natural Language Inference
→ Sentence relationships

Specifically, when choosing the sentence 1 and 2 for each pre-training example:

- 50% of time: 2 is the actual next sentence that follows 1
- 50% of time: 2 is a random sentence



BERT: Bidirectional Encoder Representations from Transformers



Pre-training Fine-Tuning

BERT: Experiment & Results

Importance of both tasks

Tasks	MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\mathrm{BERT}_{\mathrm{BASE}}$	**84.4**	**88.4**	**86.7**	**92.7**	**88.5**
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Influence of hyperparameters

Hyper-parameters				Dev Set Accuracy		
#L	d_{model}	#H	LM (ppl)	MNLl-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

BERT: Experiment & Results

BERT is SOTA

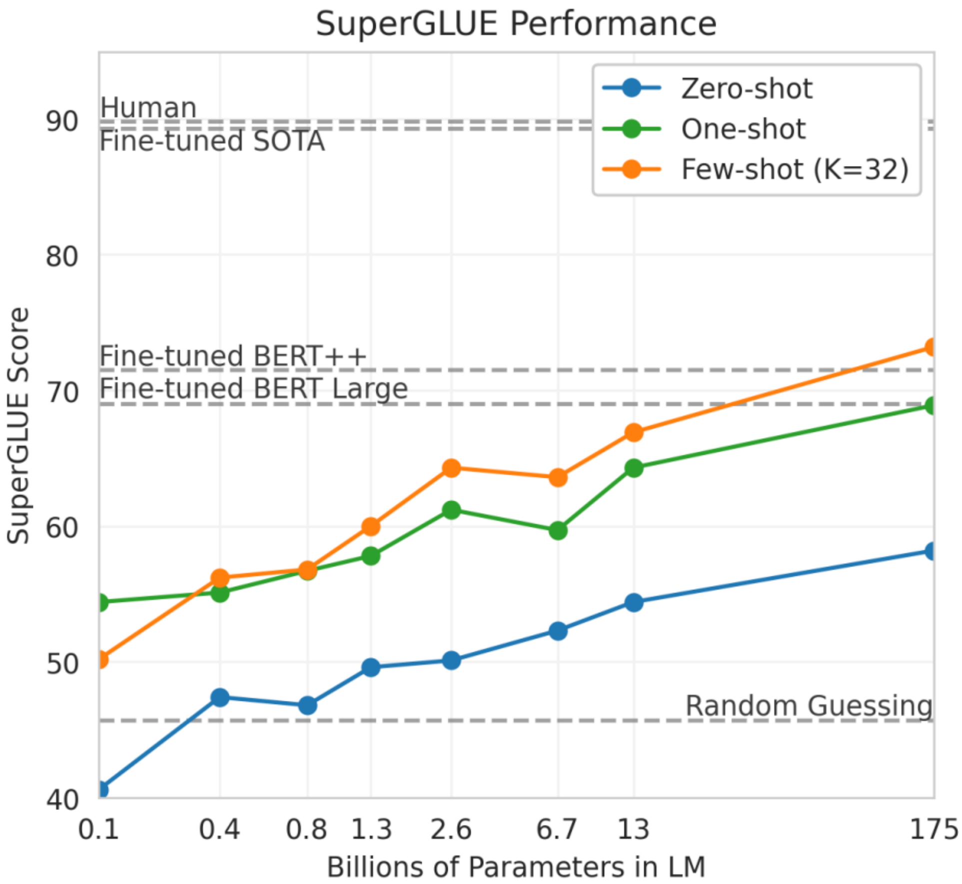
System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Comparison: GPT & BERT

Word Embedding	$N_{vocab} \times d_{model}$
BERT Encoder	$12 N d_{model} \times d_{model}$
GPT Decoder	$12 N d_{model} \times d_{model}$

Model	**Layer**	**d_{model}**	**#parameter**
$\text{BERT}_{\text{BASE}}$	12	768	110M
$\text{BERT}_{\text{LARGE}}$	24	1024	340M
GPT	12	768	110M
GPT2	48	1600	1542M
GPT3	96	12288	175B

Comparison: GPT & BERT



SuperGLUE Benchmark

Thank You!