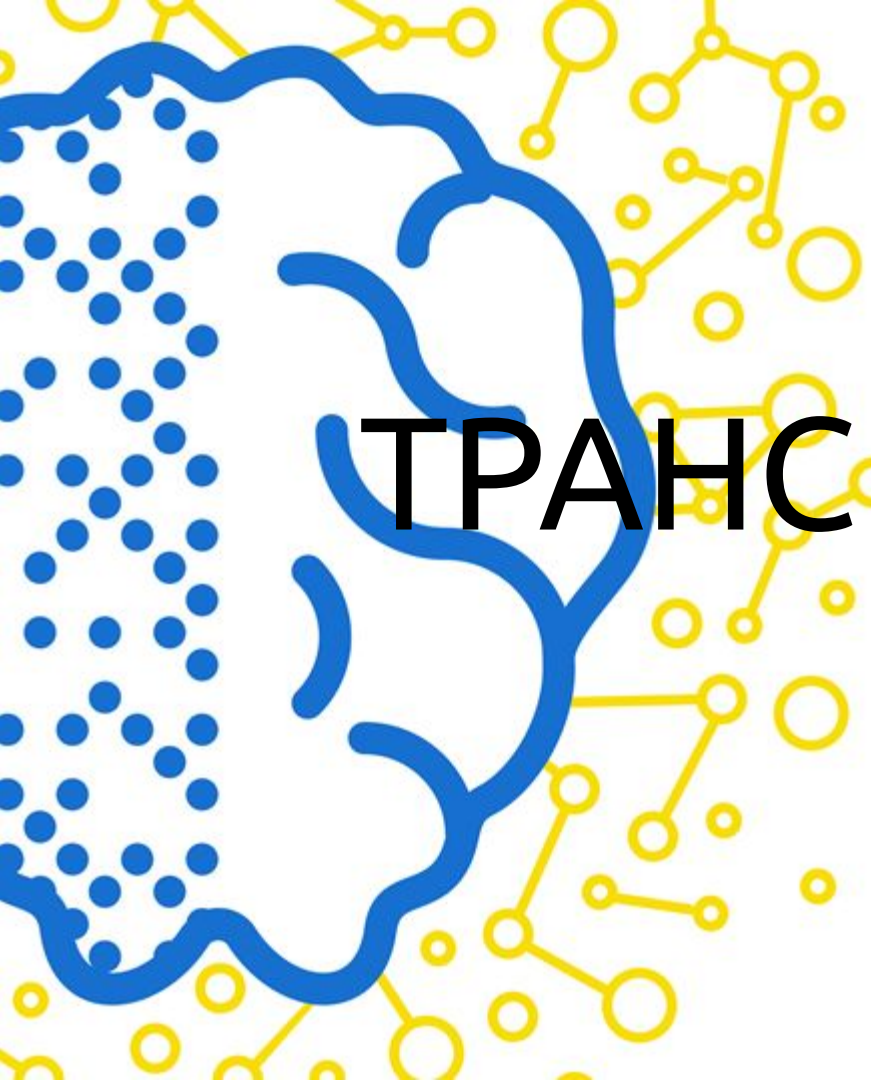
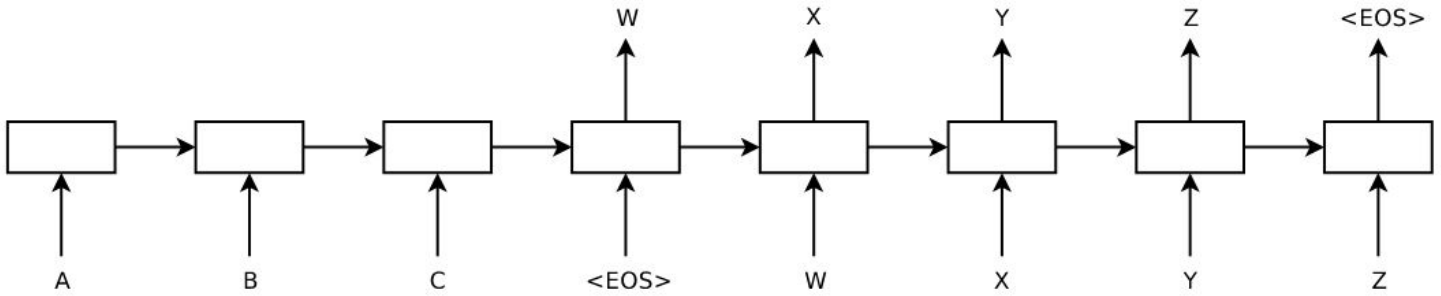


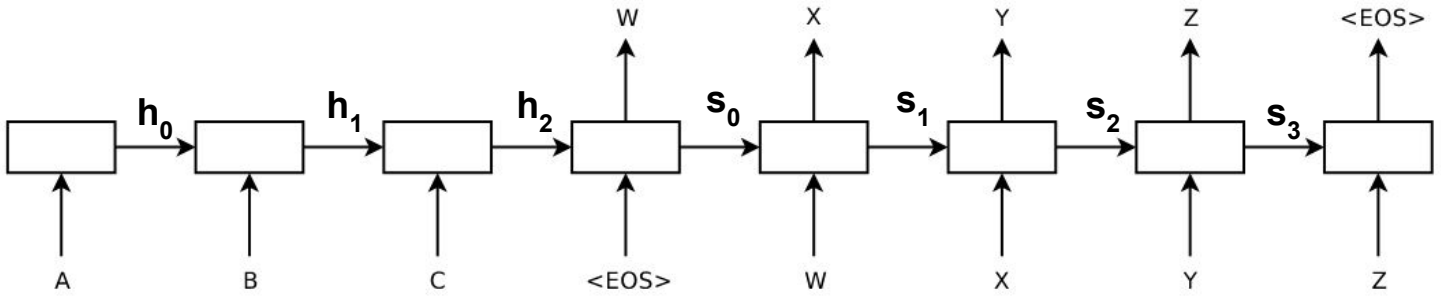
ТРАНСФОРМЕР

Vanilla seq2seq



[Sutskever et al. 14] Sequence to Sequence Learning with Neural Networks

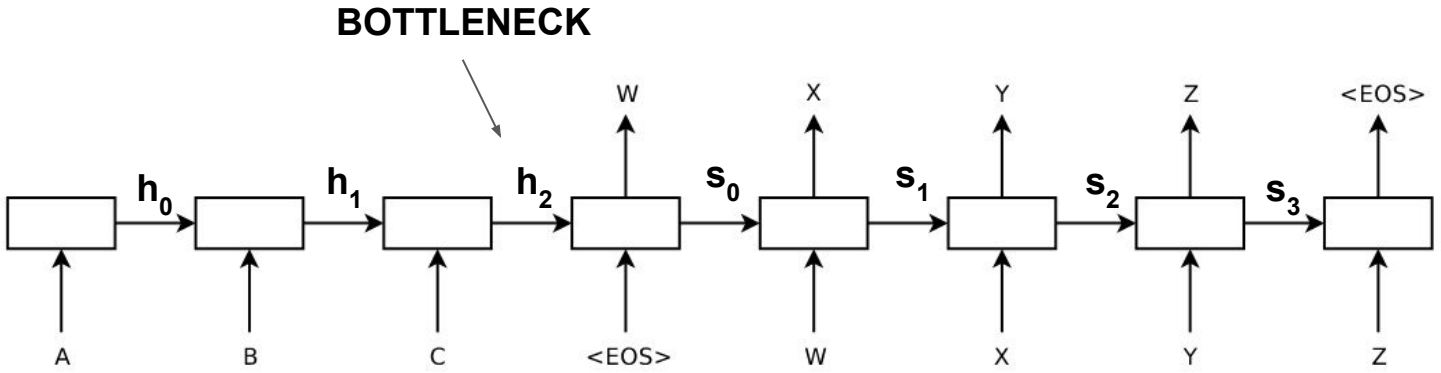
Vanilla seq2seq



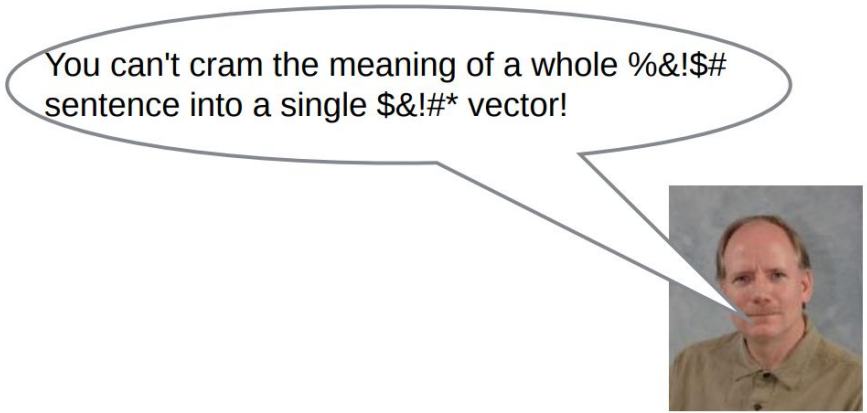
$\mathbf{h}_i$ - encoder hidden state
$\mathbf{s}_j$ - decoder hidden state

[Sutskever et al. 14] Sequence to Sequence Learning with Neural Networks

Vanilla seq2seq

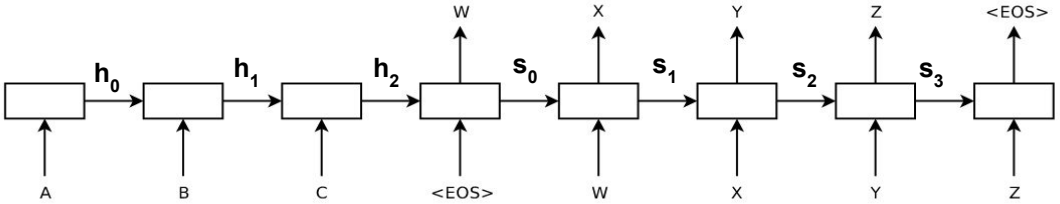


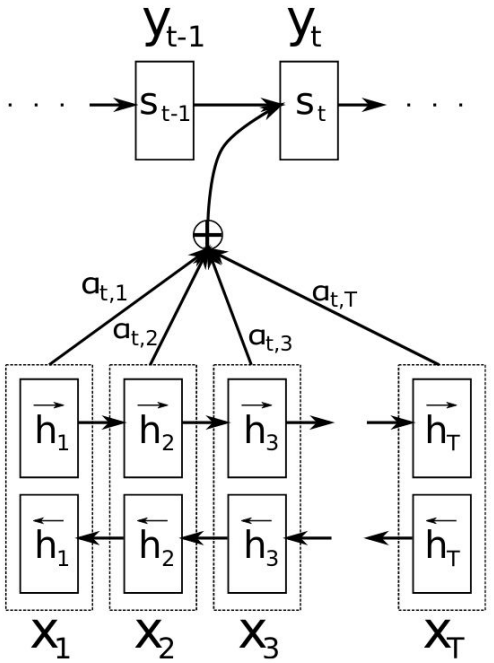
h_i - encoder hidden state
s_j - decoder hidden state



[Sutskever et al. 14] Sequence to Sequence Learning with Neural Networks

Attention Beginnings





$\mathbf{h}_i$ - encoder hidden state

$\mathbf{s}_j$ - decoder hidden state

$\alpha_{j,i}$ - attention weight from **j**-th decoder state to **i**-th encoder state

[Bahdanau et al. 15] Neural Machine Translation by Jointly Learning to Align and Translate

[Luong et al. 15] Effective Approaches to Attention-based Neural Machine Translation

Attention Beginnings

Additive Attention

[Bahdanau 15]

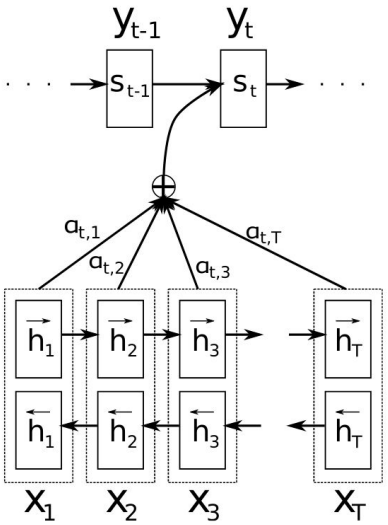
$$\alpha_{t,i}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{v}_a^T tanh(\mathbf{W}_a[\mathbf{h}_i; \mathbf{s}_t])$$

$$\alpha_{t,i}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{v}_a^T tanh(\mathbf{W}_1\mathbf{h}_i + \mathbf{W}_2\mathbf{s}_t)$$

Multiplicative Attention

[Luong 15]

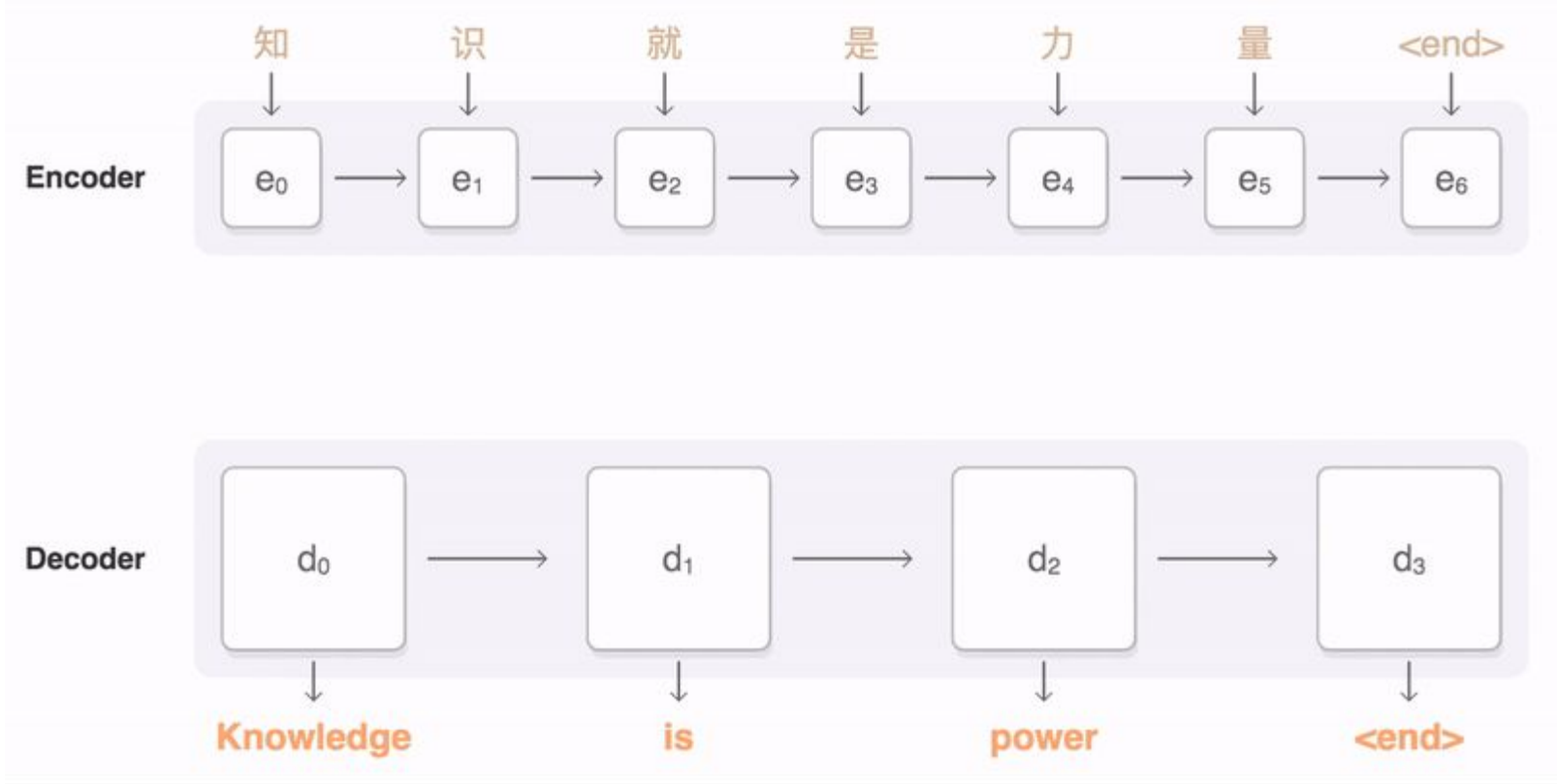
$$\alpha_{t,i}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{h}_i^T \mathbf{W}_a \mathbf{s}_t$$



[Bahdanau et al. 15] Neural Machine Translation by Jointly Learning to Align and Translate

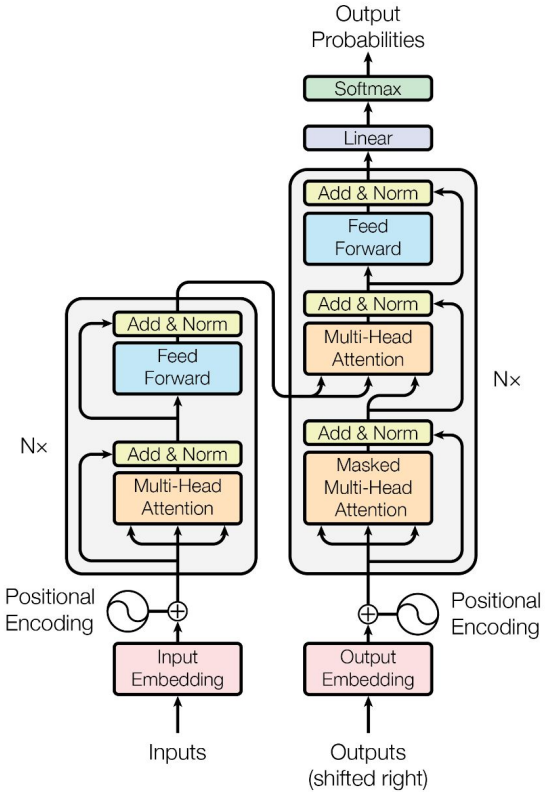
[Luong et al. 15] Effective Approaches to Attention-based Neural Machine Translation

Google seq2seq



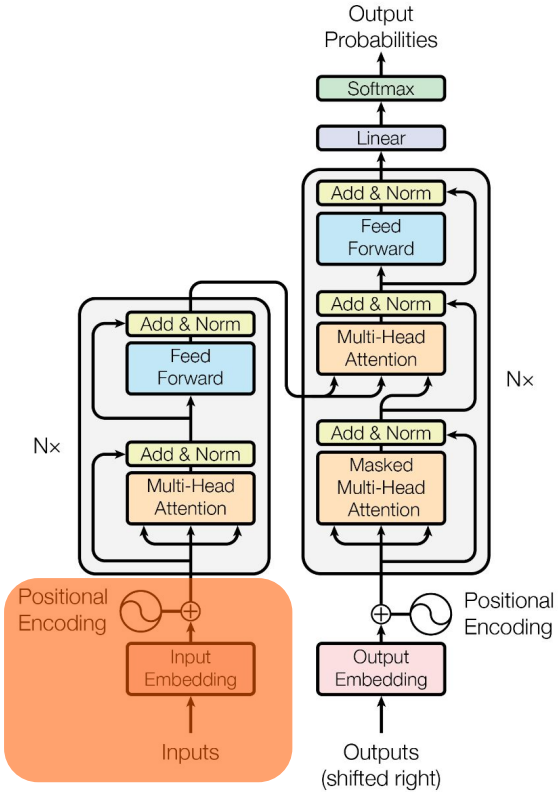
[Britz et al. 17] Massive Exploration of Neural Machine Translation Architectures

Attention Is All You Need (Transformer)



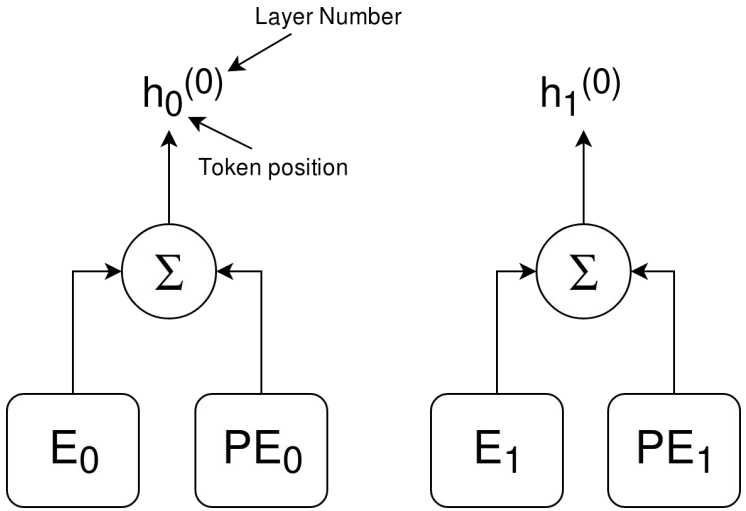
[Vaswani et al. 17] Attention Is All You Need

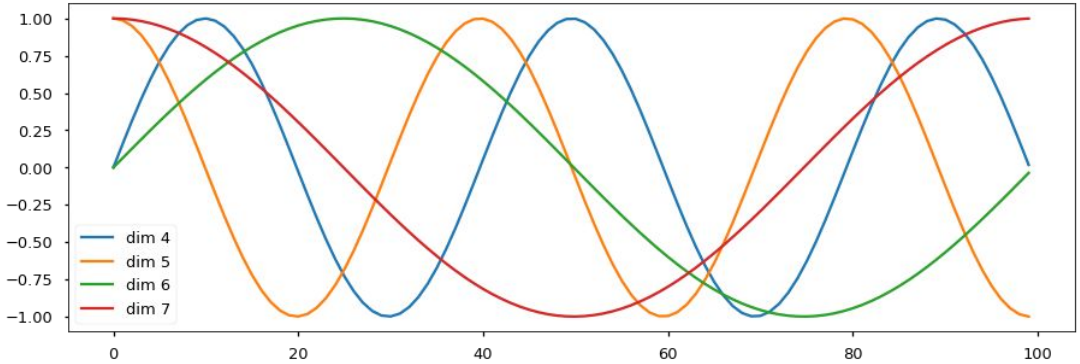
Attention Is All You Need (Transformer)



[Vaswani et al. 17] Attention Is All You Need

Embeddings. Positional and Not



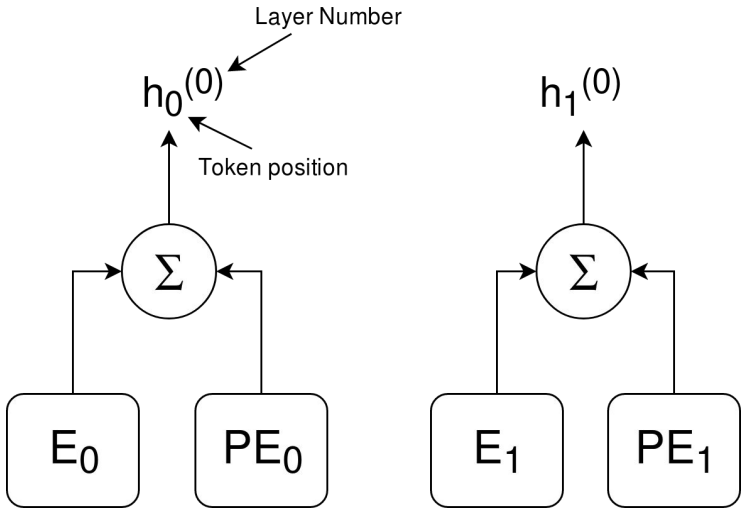


$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

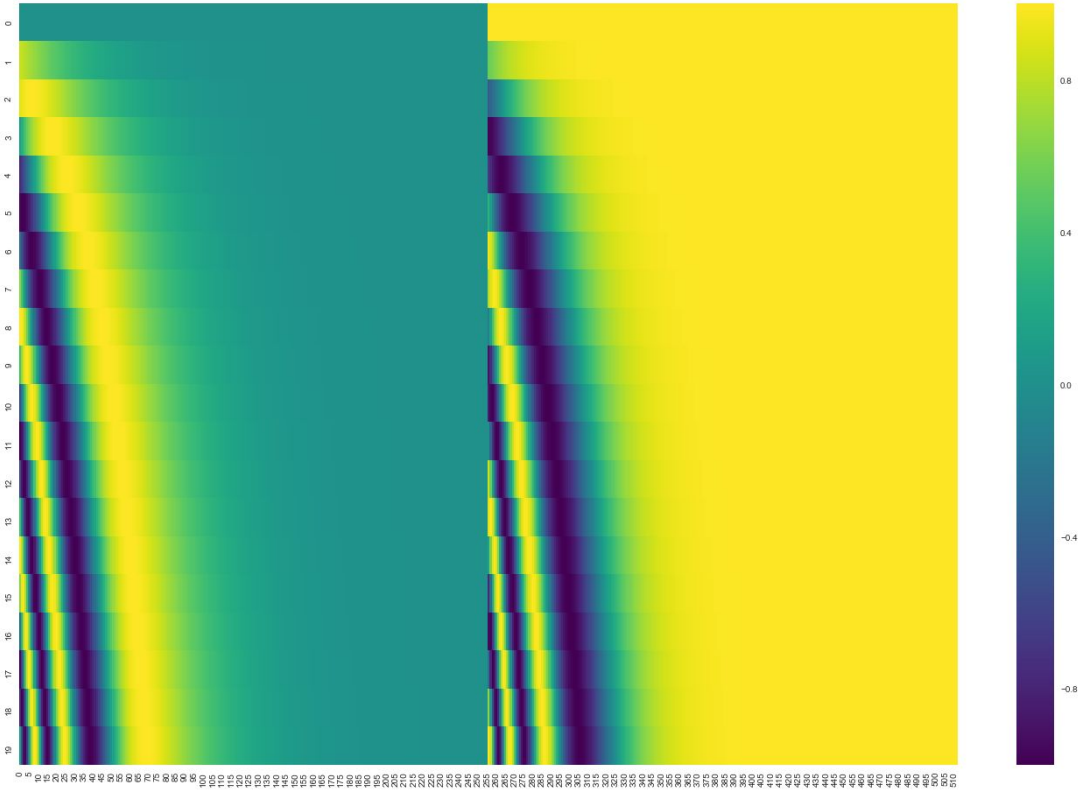
E_i is one of the 30k tokens vocabulary embeddings

[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

Embeddings. Positional and Not



$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

Byte pair encoding

Start from characters:

- a, a, a, b, d, a, a, a, b, a, c
- Vocabulary: (‘a’, ‘b’, ‘c’, ‘d’)

Merge most common bigram “**aa**”:

- **aa,** a, b, d, **aa,** a, b, a, c
- Vocabulary: (‘a’, ‘b’, ‘c’, ‘d’, ‘aa’)

Then merge the following most common bigram “**ab**”:

- aa, **ab,** d, aa, **ab,** a, c
- Vocabulary: (‘a’, ‘b’, ‘c’, ‘d’, ‘aa’, ‘ab’)

And again:

- **aabd,** d, **aabd,** a, c
- Vocabulary: (‘a’, ‘b’, ‘c’, ‘d’, ‘aa’, ‘ab’, ‘abbd’)

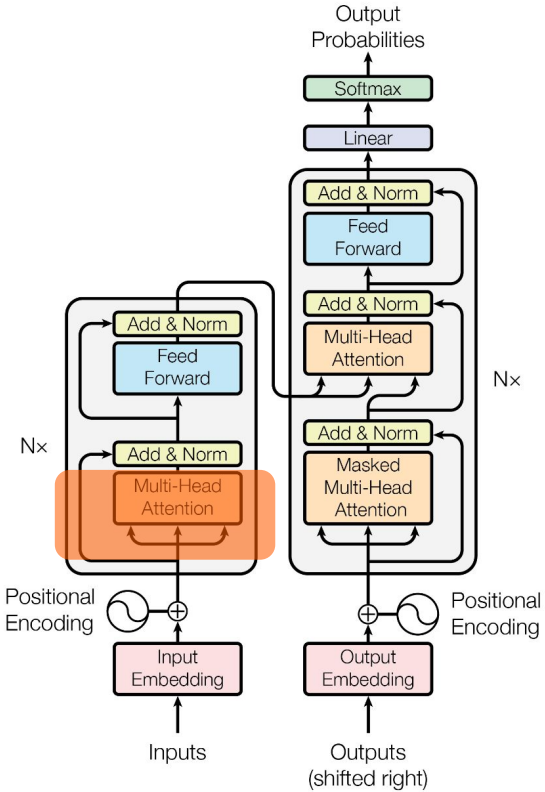
An example of tokenization:

tokenize(‘he likes playing’) →

['he', 'likes', 'play', '##ing']

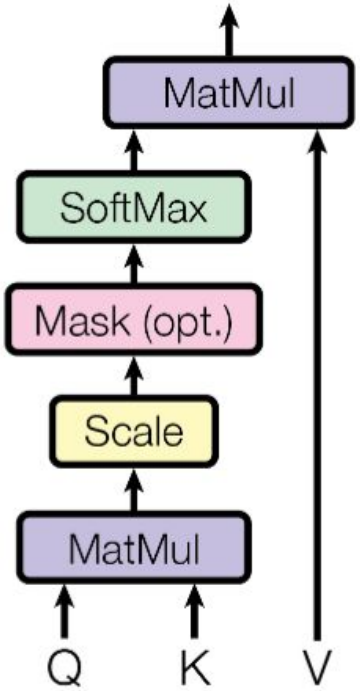
[Sennrich et al. 16] Neural Machine Translation of Rare Words with Subword Units

Attention Is All You Need (Transformer)



[Vaswani et al. 17] Attention Is All You Need

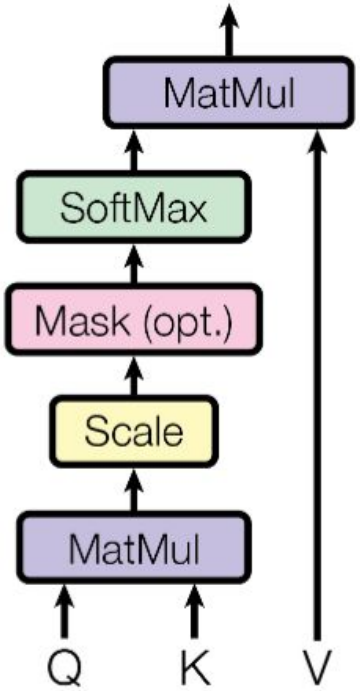
Key-Value Attention



[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

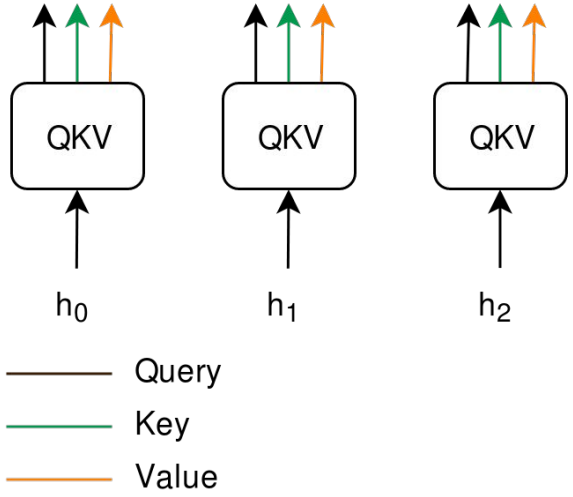
Key-Value Attention

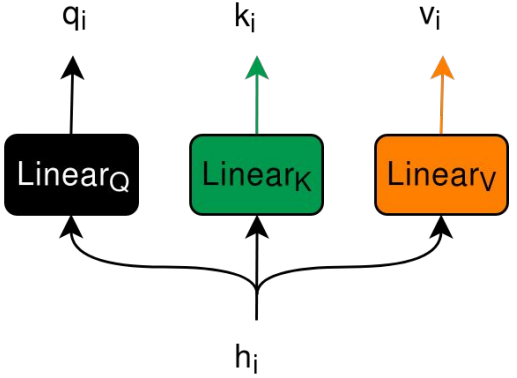
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

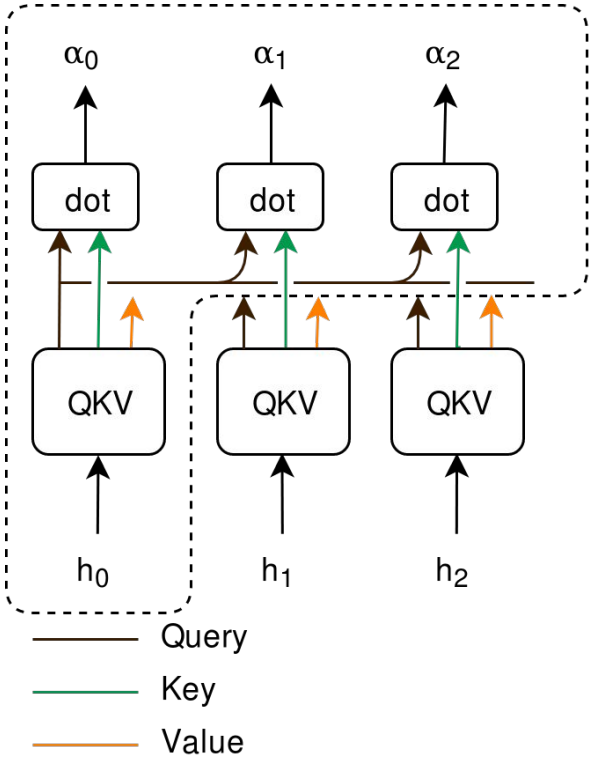
Key-Value Attention





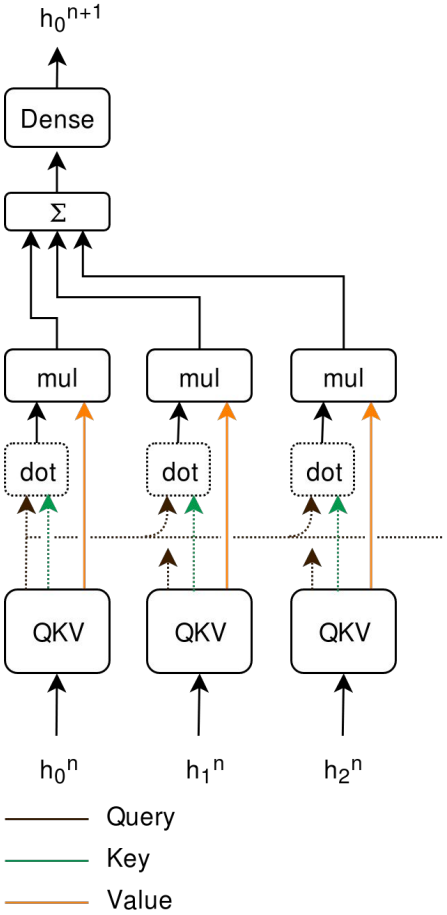
[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

Key-Value Attention



[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

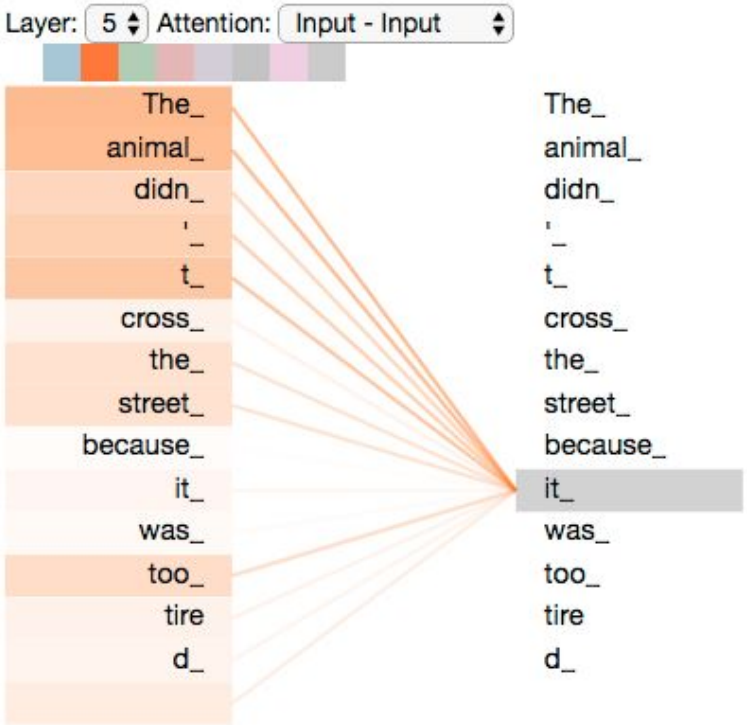
Key-Value Attention



iPavlov.ai

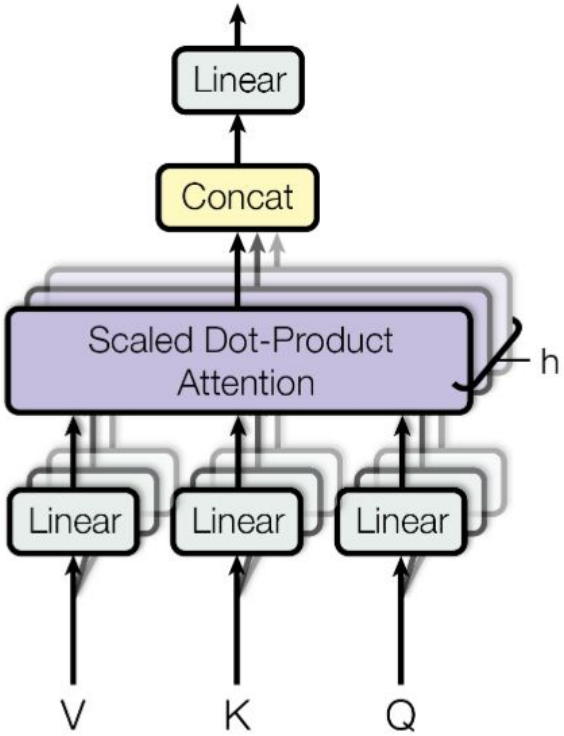
[Vaswani et al. 17] Attention Is All You Need

Key-Value Attention Visualization



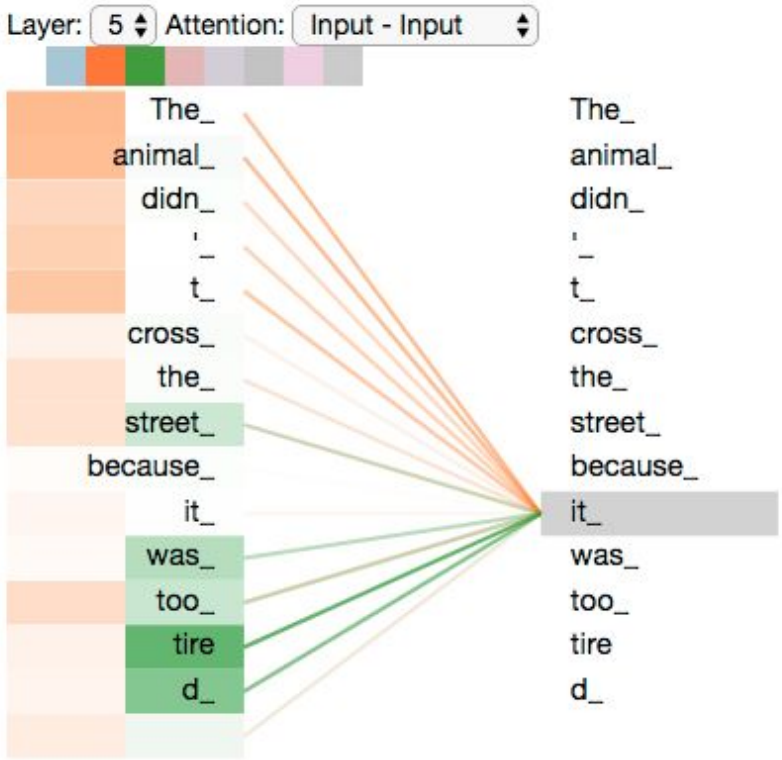
[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

Multihead Attention



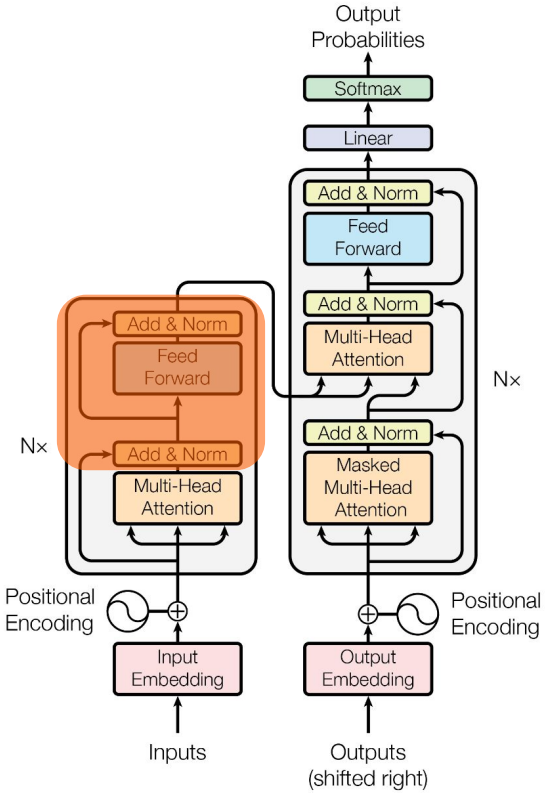
[Vaswani et al. 17] Attention Is All You Need

Multihead Attention Visualization



[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

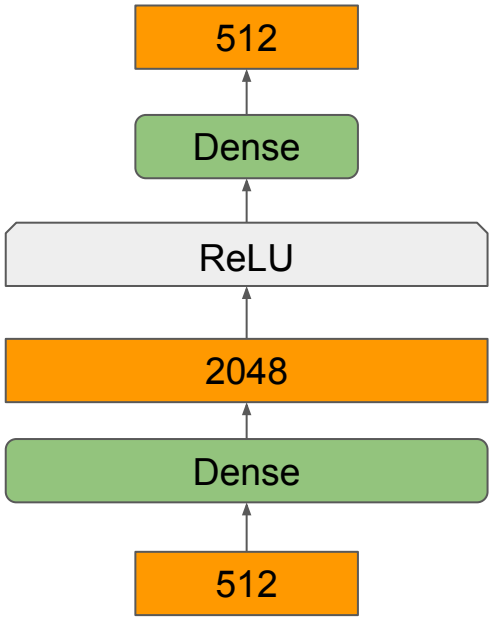
Feed Forward + Normalization

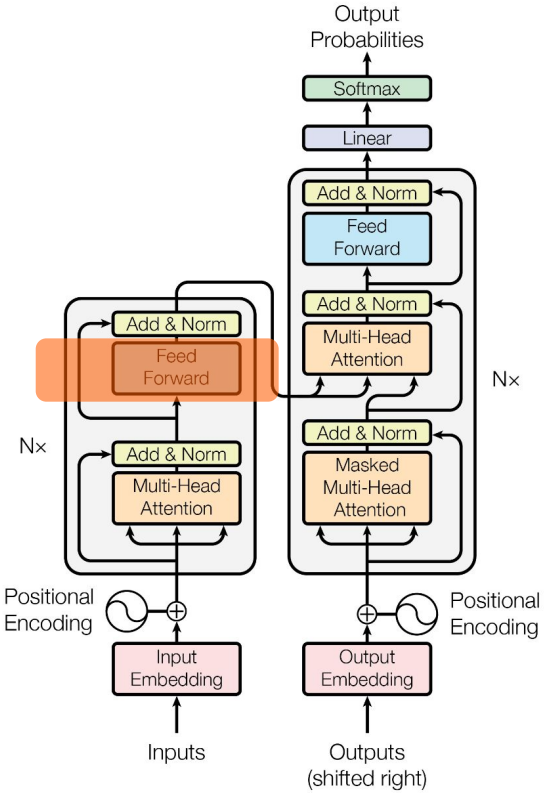


[Vaswani et al. 17] Attention Is All You Need

Feed Forward + Normalization

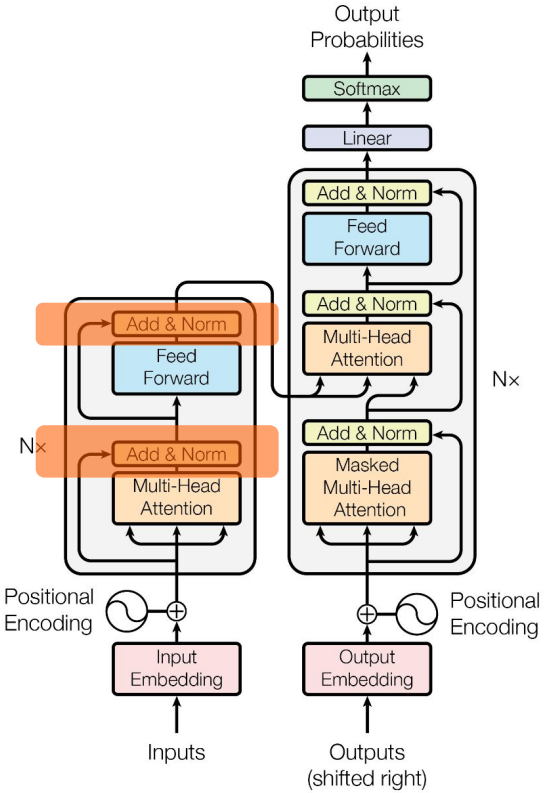
$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$





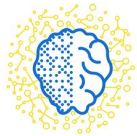
[Vaswani et al. 17] Attention Is All You Need

Feed Forward + Normalization



[Vaswani et al. 17] Attention Is All You Need

Batch Norm



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

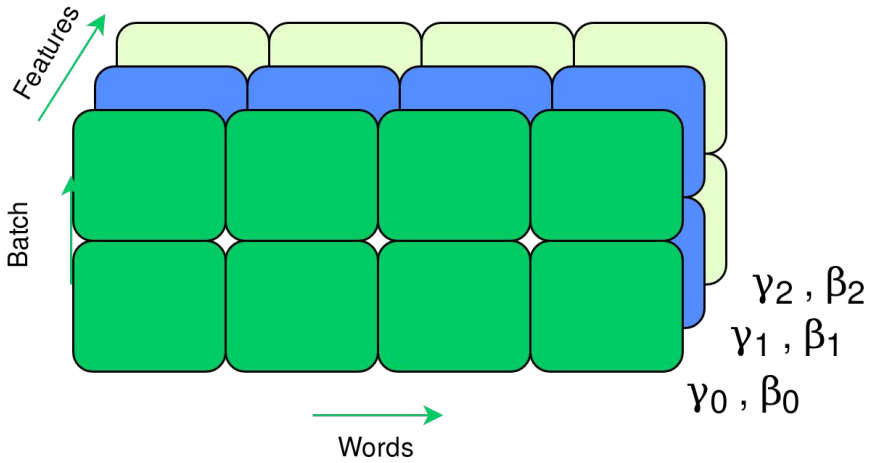
$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

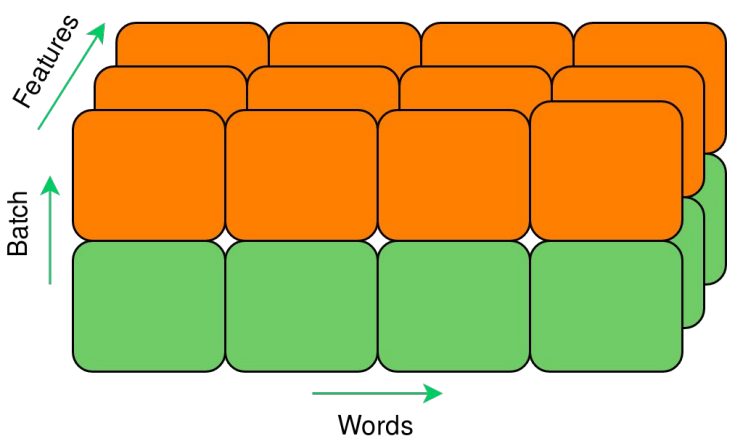
[Ioffe and Szegedy 15] Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

Layer Norm

Batch Norm



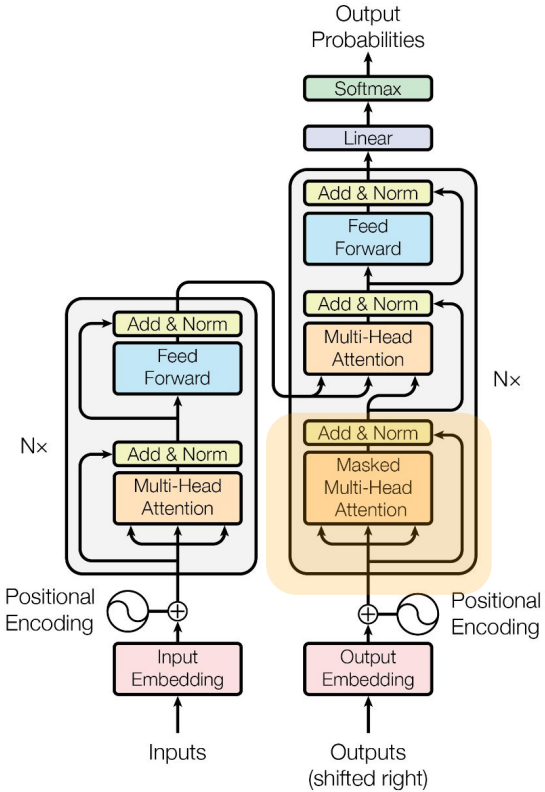
Layer Norm



$$\mathbf{h}^t = f\left[\frac{\mathbf{g}}{\sigma^t} \odot (\mathbf{a}^t - \mu^t) + \mathbf{b}\right] \qquad \mu^t = \frac{1}{H}\sum_{i=1}^{H} a_i^t \qquad \sigma^t = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(a_i^t - \mu^t)^2}$$

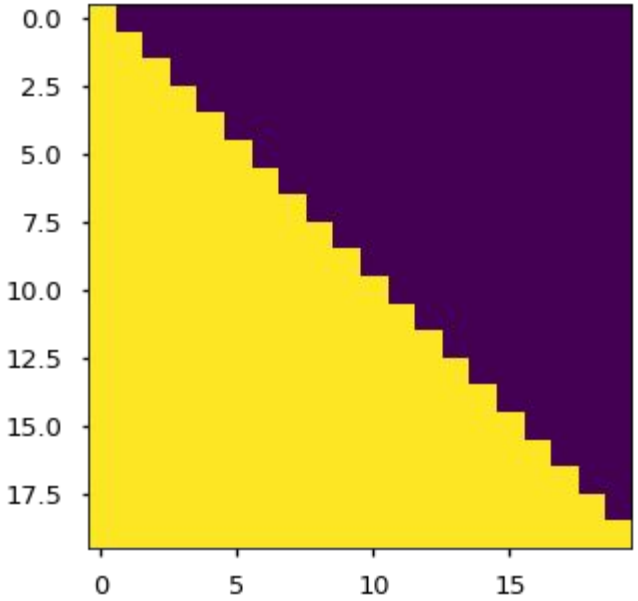
[Ba et al. 16] Layer Normalization

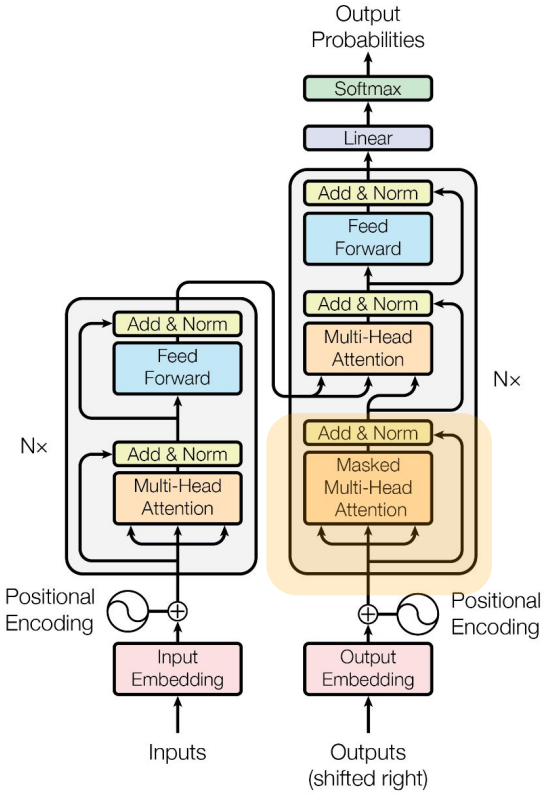
Decoder



[Vaswani et al. 17] Attention Is All You Need

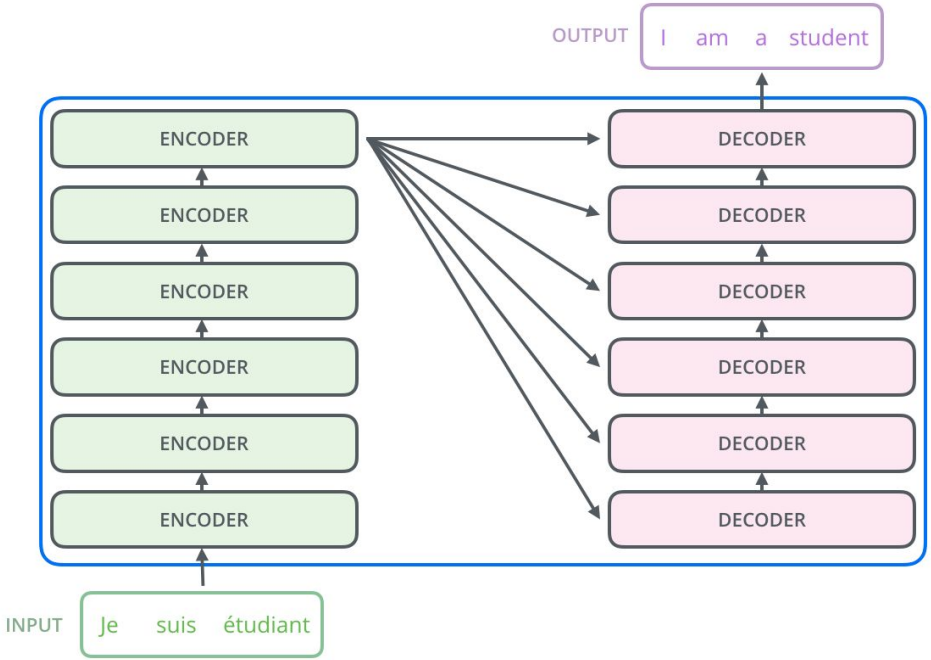
Decoder Masked Self-Attention

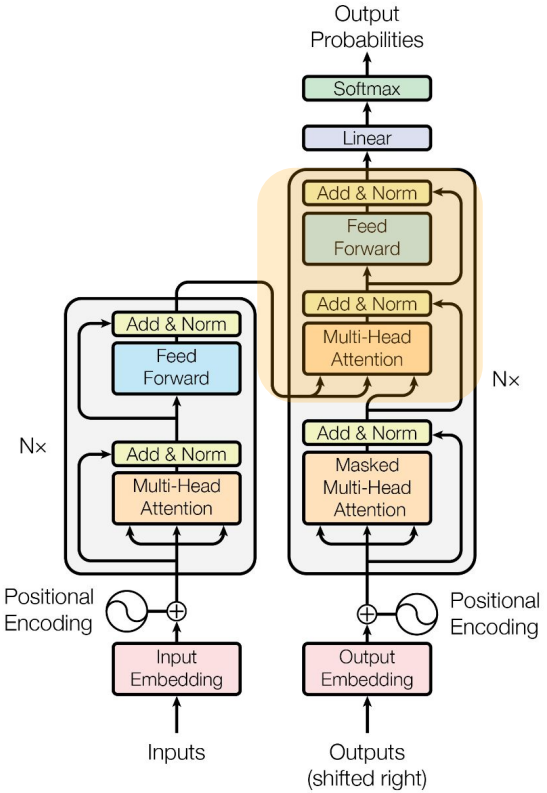




[Vaswani et al. 17] Attention Is All You Need

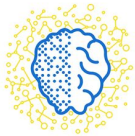
Decoder Attention to Encoder States





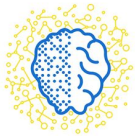
[Vaswani et al. 17] Attention Is All You Need

Visualization



Transformer: A Novel Neural Network Architecture for Language Understanding

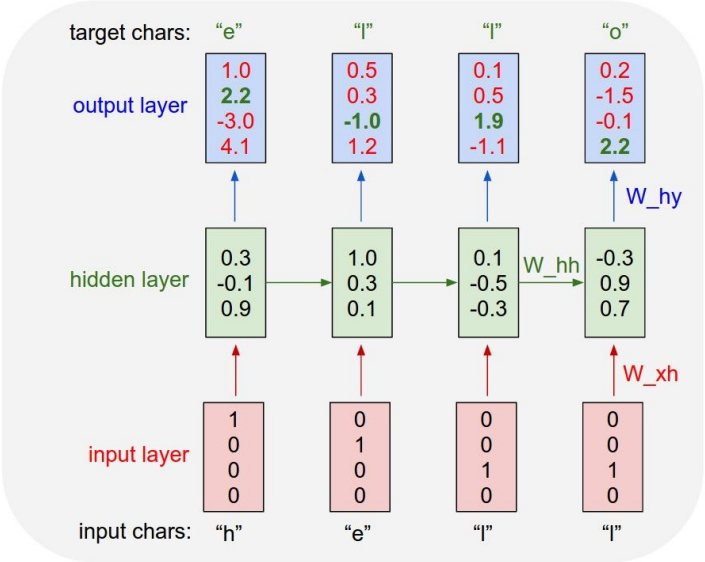
Tricks



- BPE ~30k vocab
- Label smoothing: 0.1
- Averaging the last 5 (20 for big model) checkpoints, which were written at 10-minute intervals
- Layer norm
- Learning rate is proportional to the square root of the dimensionality of the hidden states
- 4000 warm up steps, then decreasing ~ 1 / sqrt(step). Warmup from 1e-7 to 5e-4.

[Vaswani et al. 17] Attention Is All You Need
[Alammar 18] Illustrated Transformer

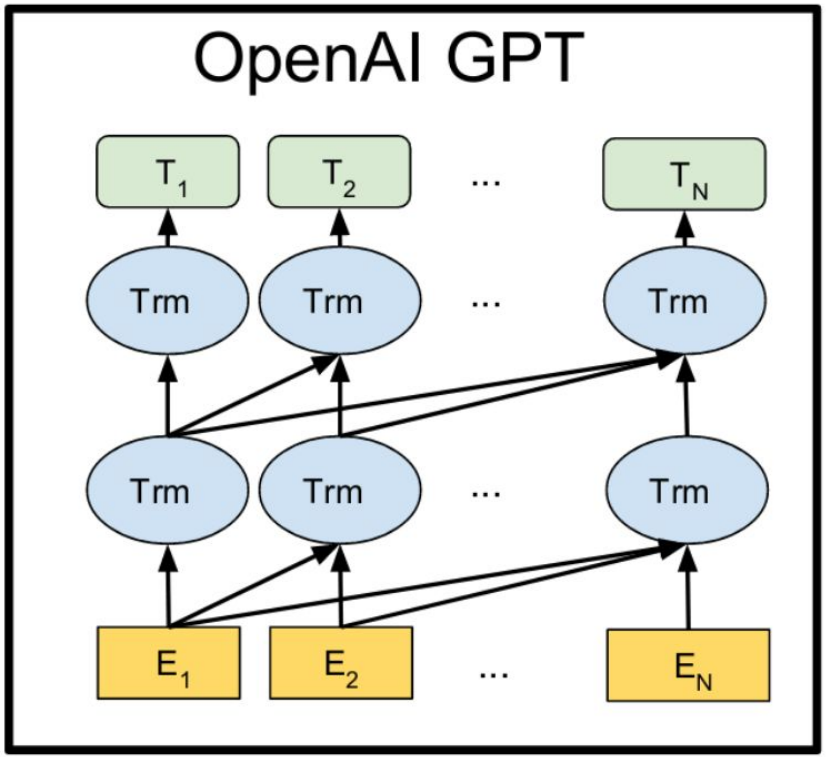
Language Modelling

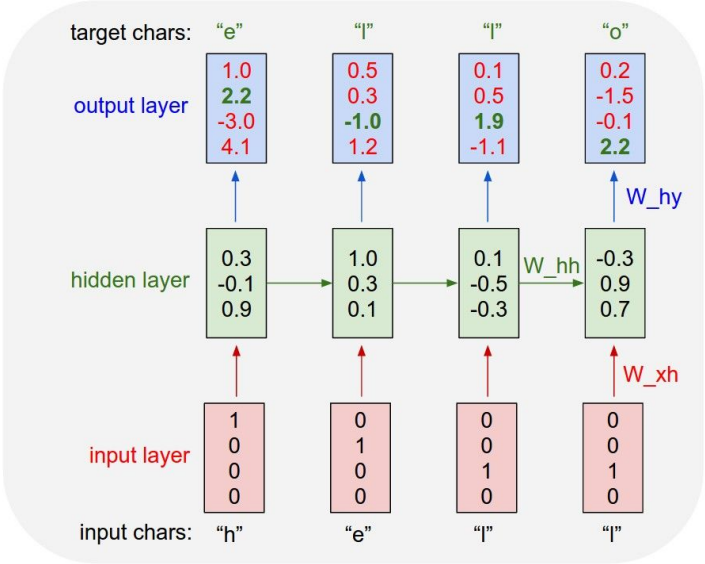


$$P(x_1,x_2,x_3,\ldots,x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1,x_2)\ldots P(x_n|x_1,\ldots,x_{n-1})$$

[Karphathy 15] The Unreasonable Effectiveness of Recurrent Neural Networks

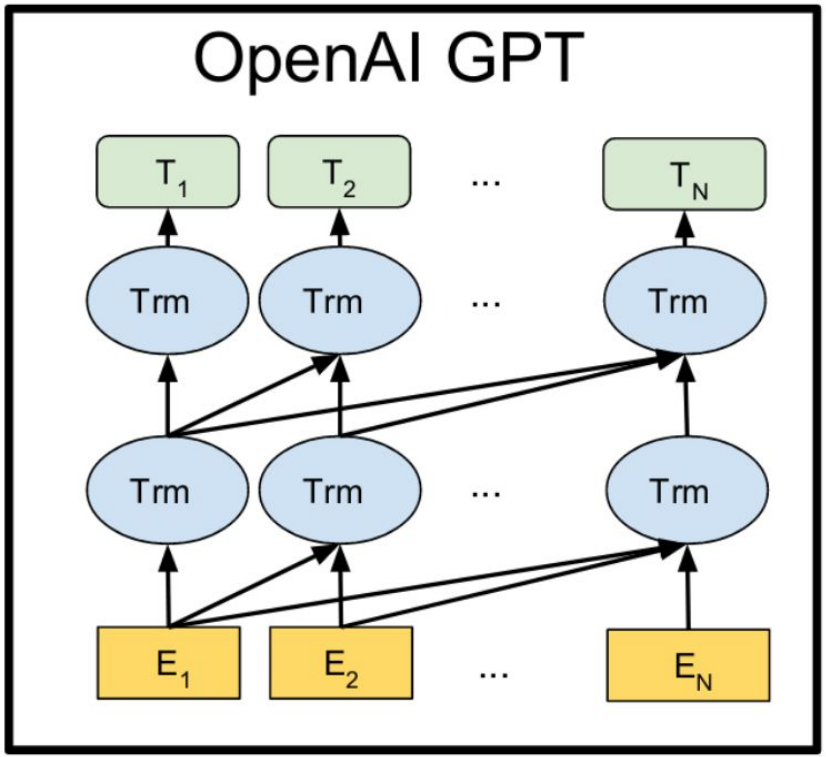
Open AI Generative Pre-Trained Transformer

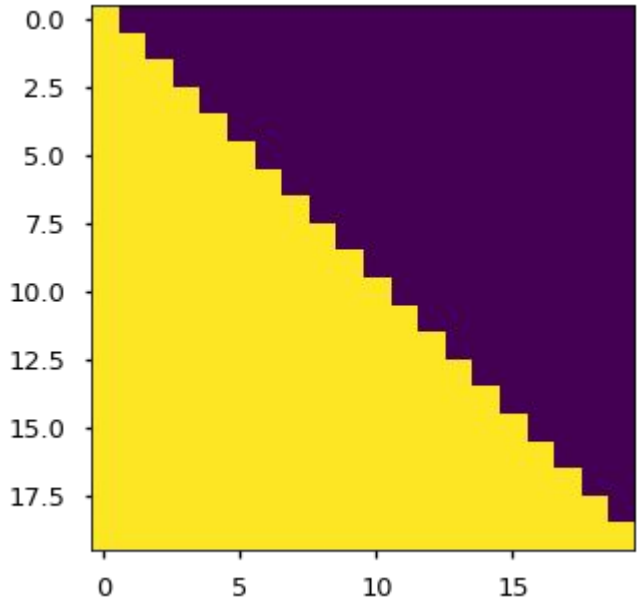




[Radford 18] Improving Language Understanding with Unsupervised Learning

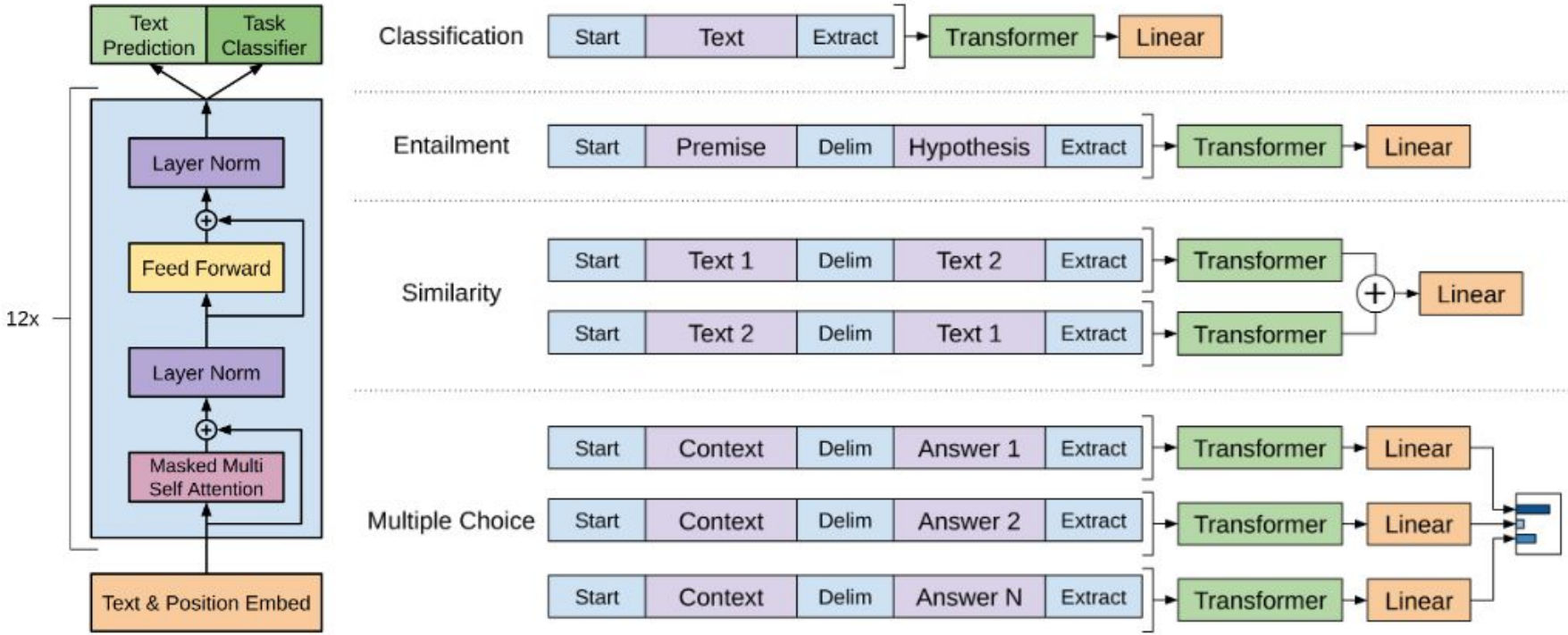
Open AI Generative Pre-Trained Transformer





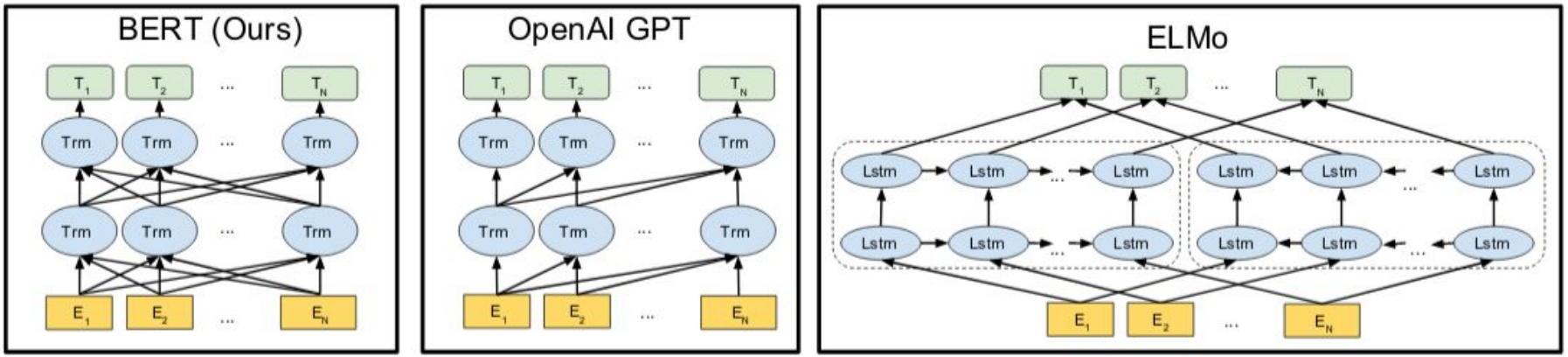
[Radford 18] Improving Language Understanding with Unsupervised Learning

Open AI Generative Pre-Trained Transformer



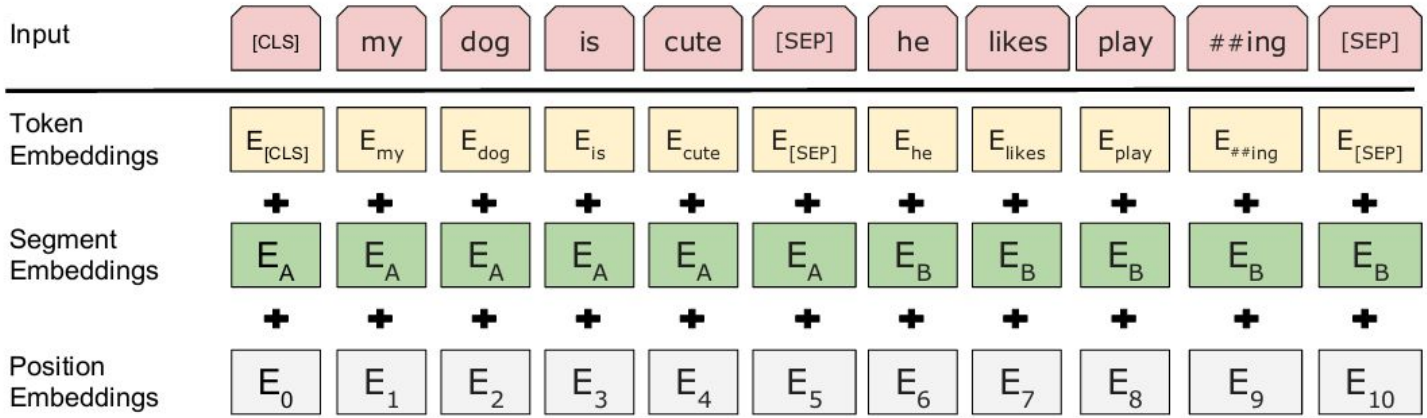
[Radford 18] Improving Language Understanding with Unsupervised Learning

BERT



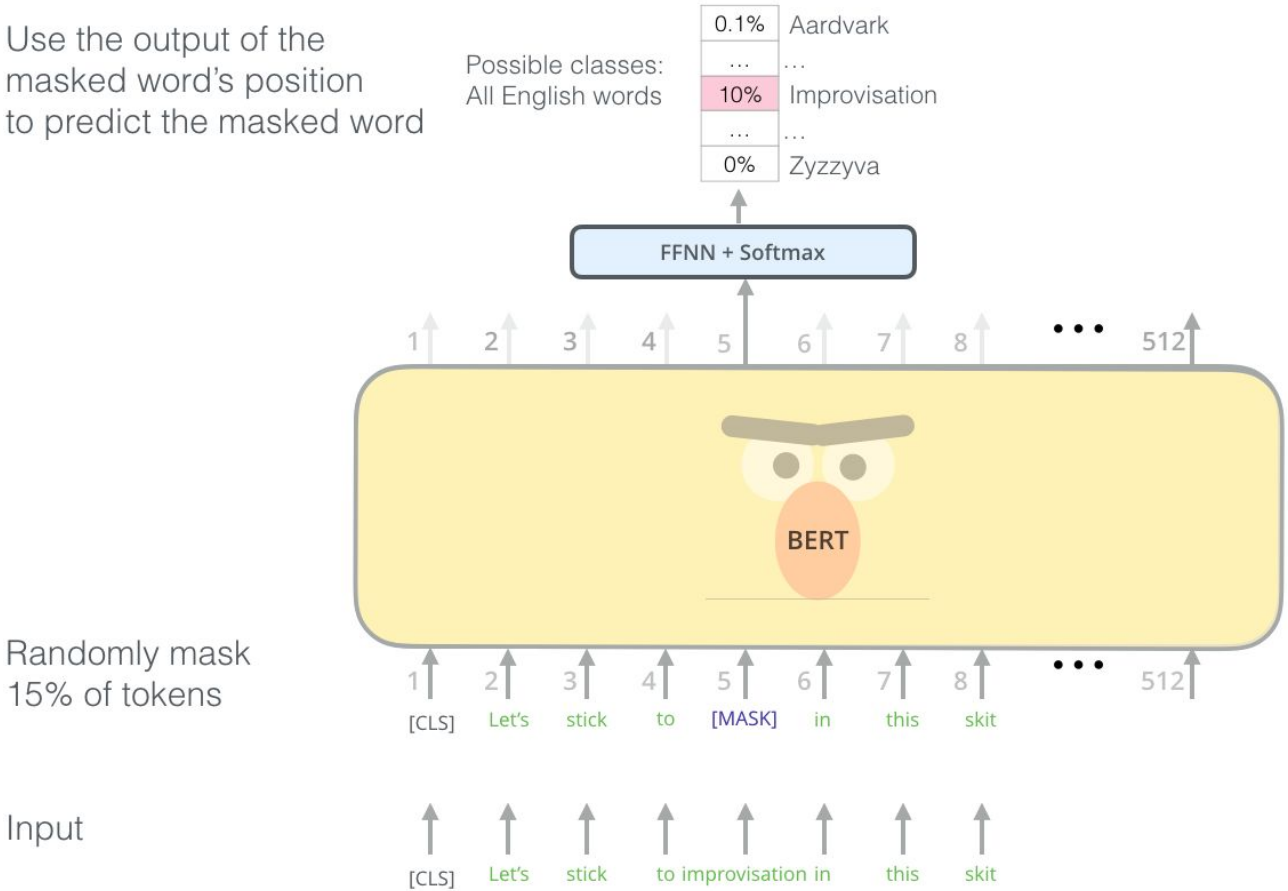
[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT Input Embeddings



[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

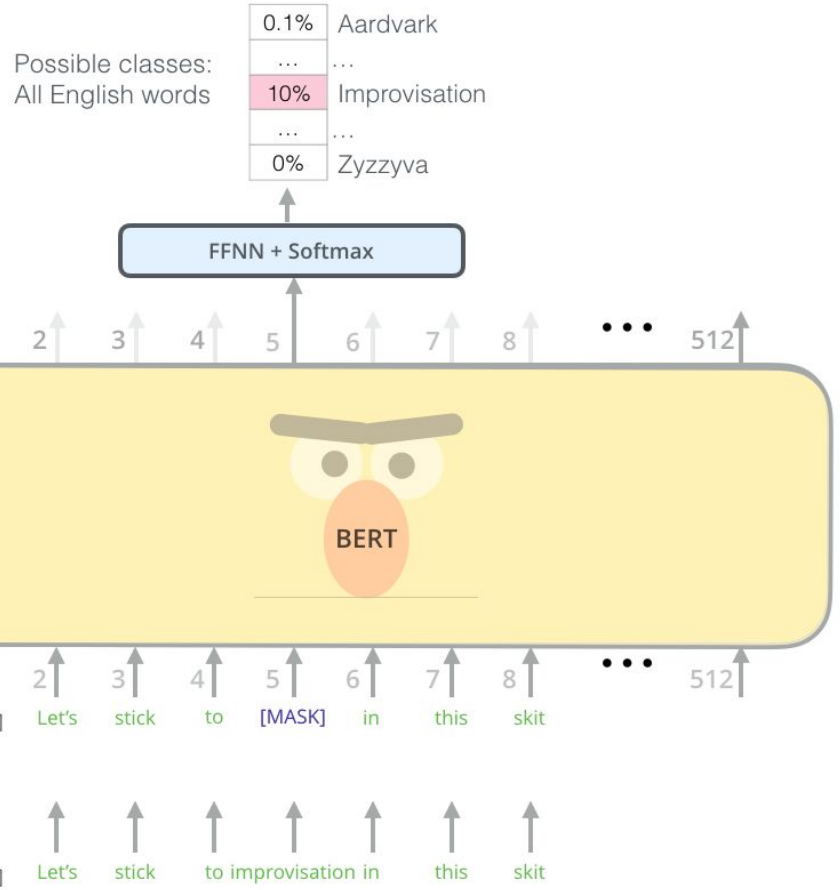
BERT



[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT

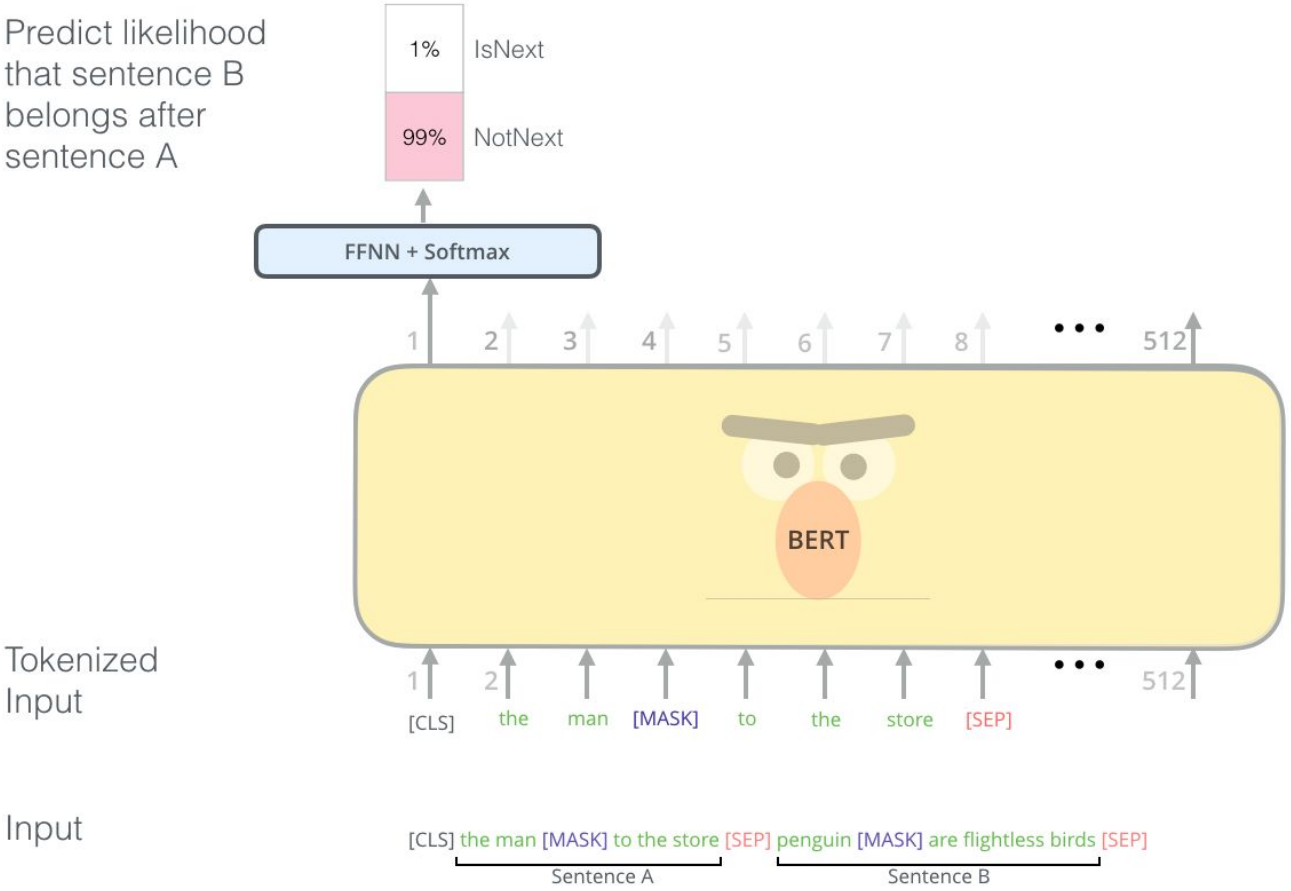
Use the output of the masked word's position to predict the masked word



- Rather than *always* replacing the chosen words with `[MASK]`, the data generator will do the following:
- 80% of the time: Replace the word with the `[MASK]` token, e.g., `my dog is hairy` → `my dog is [MASK]`
- 10% of the time: Replace the word with a random word, e.g., `my dog is hairy` → `my dog is apple`
- 10% of the time: Keep the word unchanged, e.g., `my dog is hairy` → `my dog is hairy`. The purpose of this is to bias the representation towards the actual observed word.

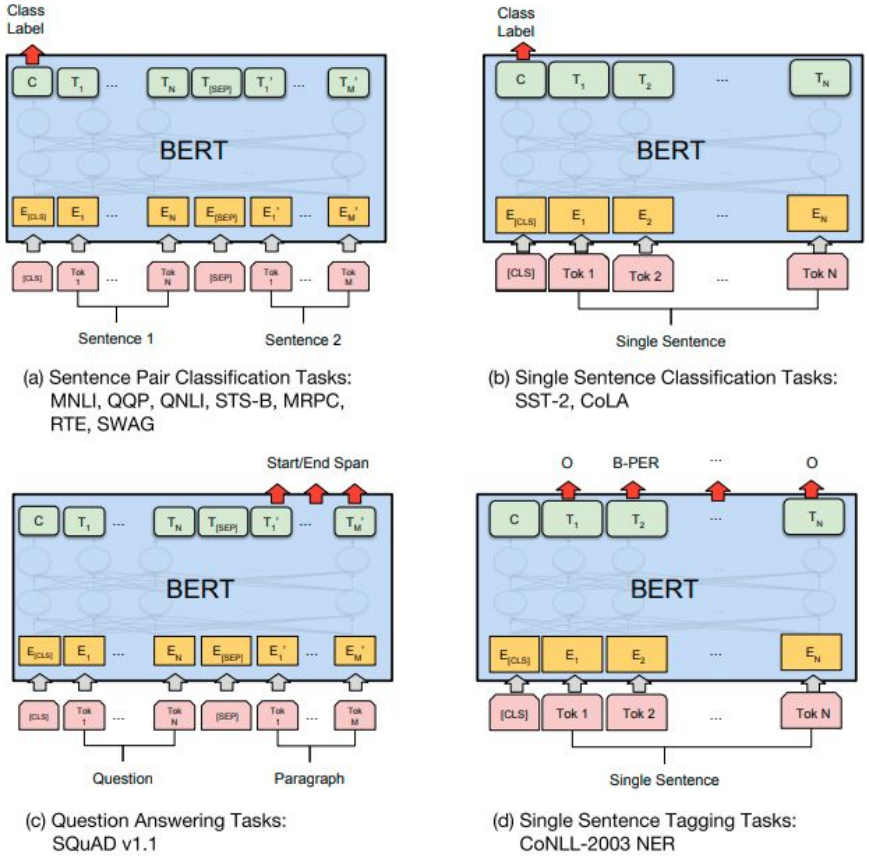
[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT



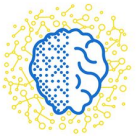
[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT



[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

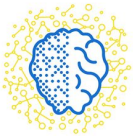
BERT



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**

[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT



System	Dev		Test	
	EM	F1	EM	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**

Table 2: SQuAD results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT

System	Dev F1	Test F1
ELMo+BiLSTM+CRF	95.7	92.2
CVT+Multi (Clark et al., 2018)	-	92.6
$\text{BERT}_{\text{BASE}}$	96.4	92.4
$\text{BERT}_{\text{LARGE}}$	**96.6**	**92.8**

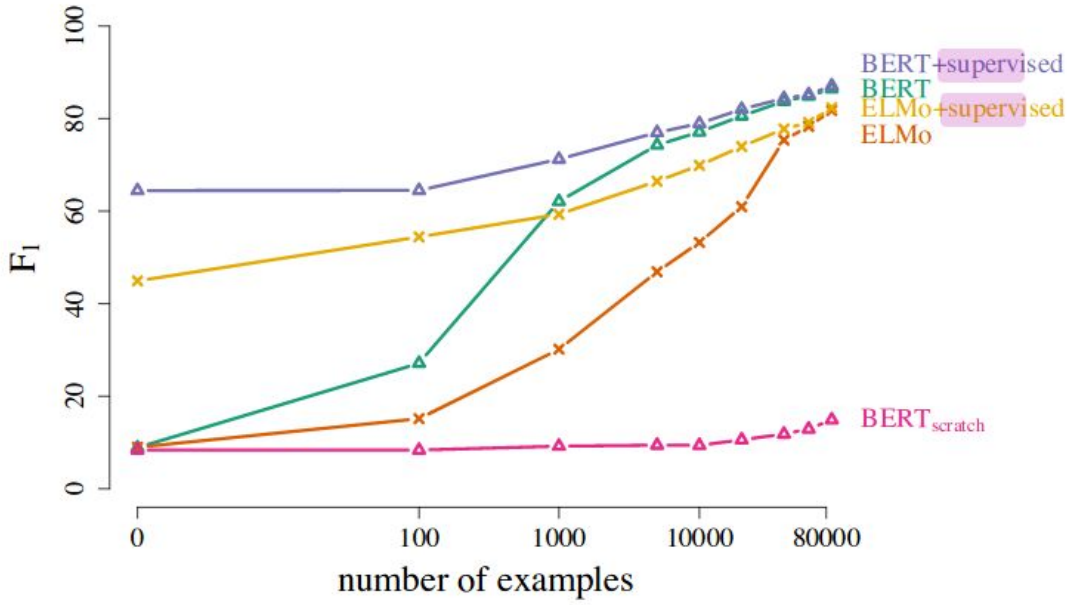
Table 3: CoNLL-2003 Named Entity Recognition results. The hyperparameters were selected using the Dev set, and the reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

[Devlin et al. 18] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Learning and Evaluating General Linguistic Intelligence

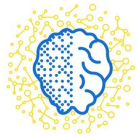
Supervised pre-training tasks:

- Semantic Role Labeling
- Relation Extraction
- Natural Language Inference
- Reading Comprehension (TriviaQA and QuAC)



[Yogatama et al. 19] Learning and Evaluating General Linguistic Intelligence

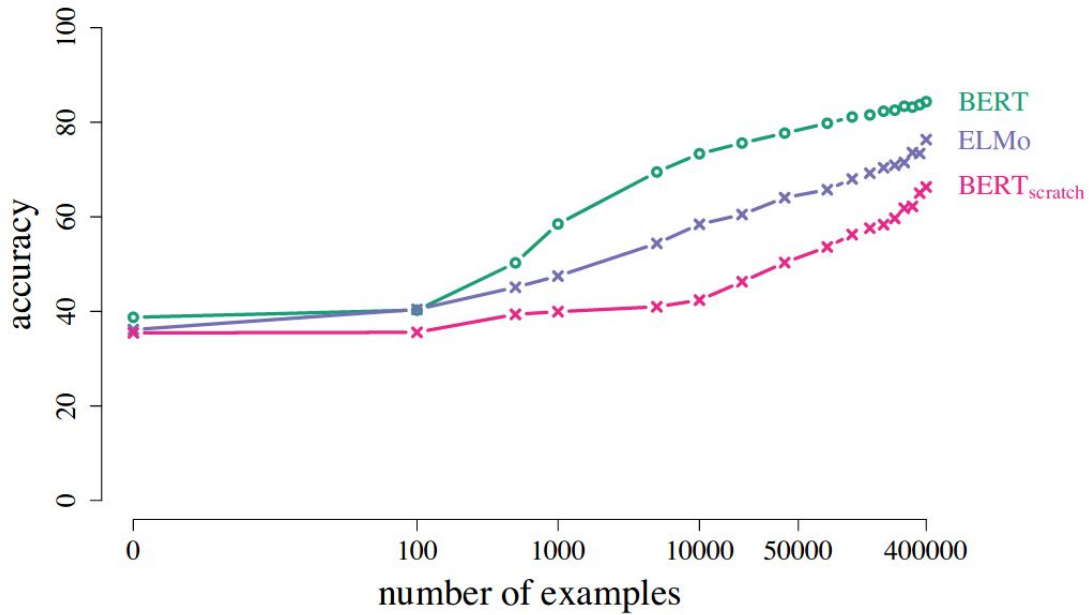
Learning and Evaluating General Linguistic Intelligence



Model	EM (↑)	F_1 (↑)
BERT	78.5	86.5
BERT + supervised	79.4	87.1
ELMo	72.1	81.8
ELMo + supervised	72.8	82.3

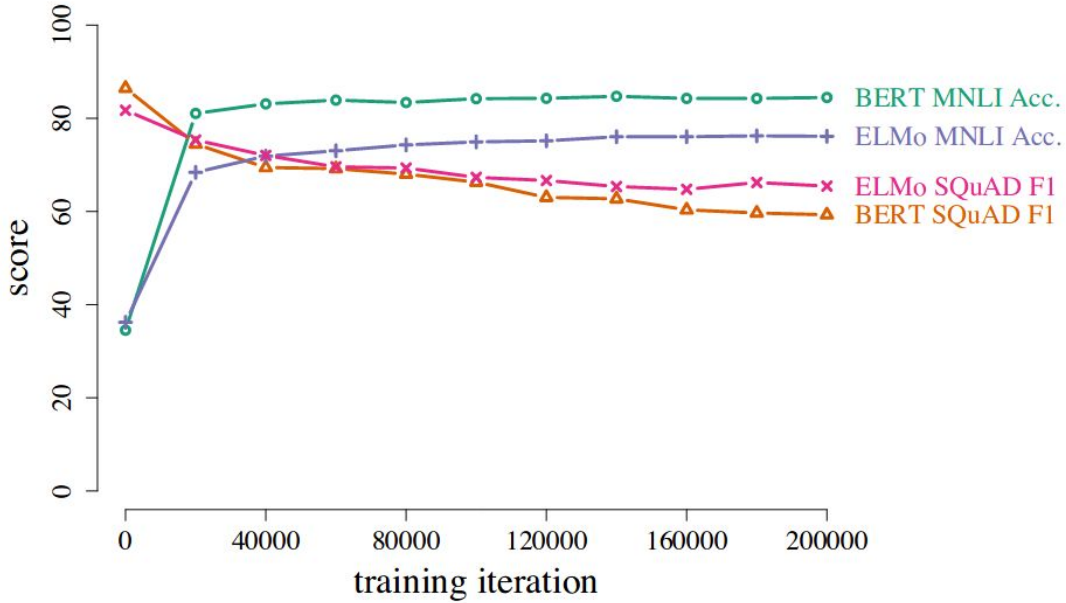
[Yogatama et al. 19] Learning and Evaluating General Linguistic Intelligence

BERT and ELMo for MNLI



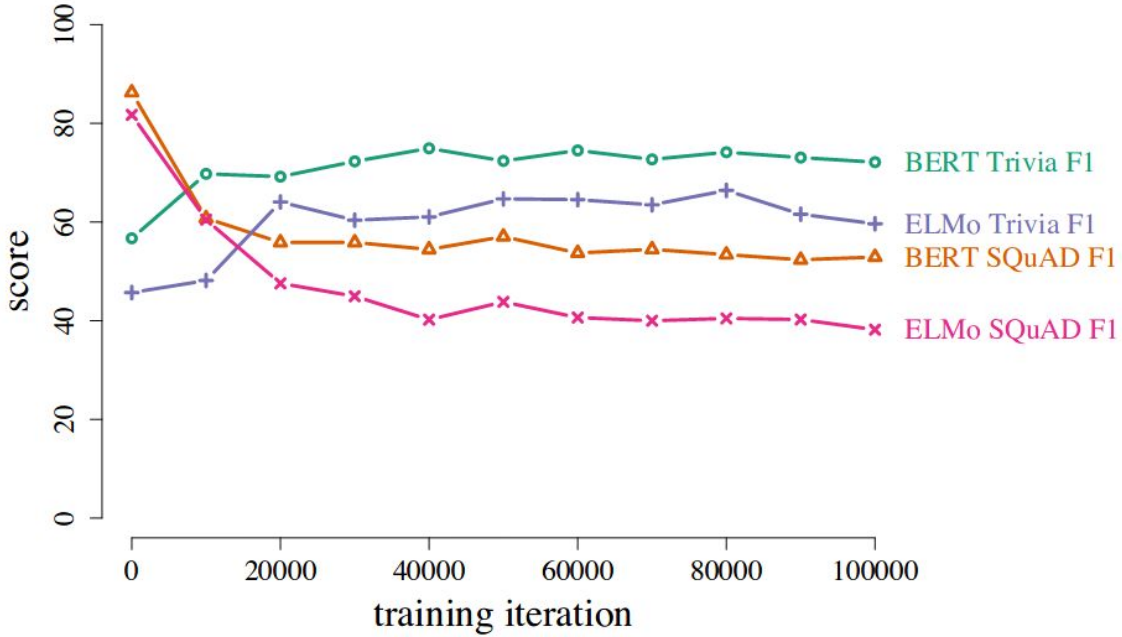
[Yogatama et al. 19] Learning and Evaluating General Linguistic Intelligence

Unsupervised →SQuAD → MNLI



[Yogatama et al. 19] Learning and Evaluating General Linguistic Intelligence

Unsupervised →SQuAD → Trivia



[Yogatama et al. 19] Learning and Evaluating General Linguistic Intelligence

Spasibo