

Основы глубинного обучения

Лекция 11

Предобученные трансформеры

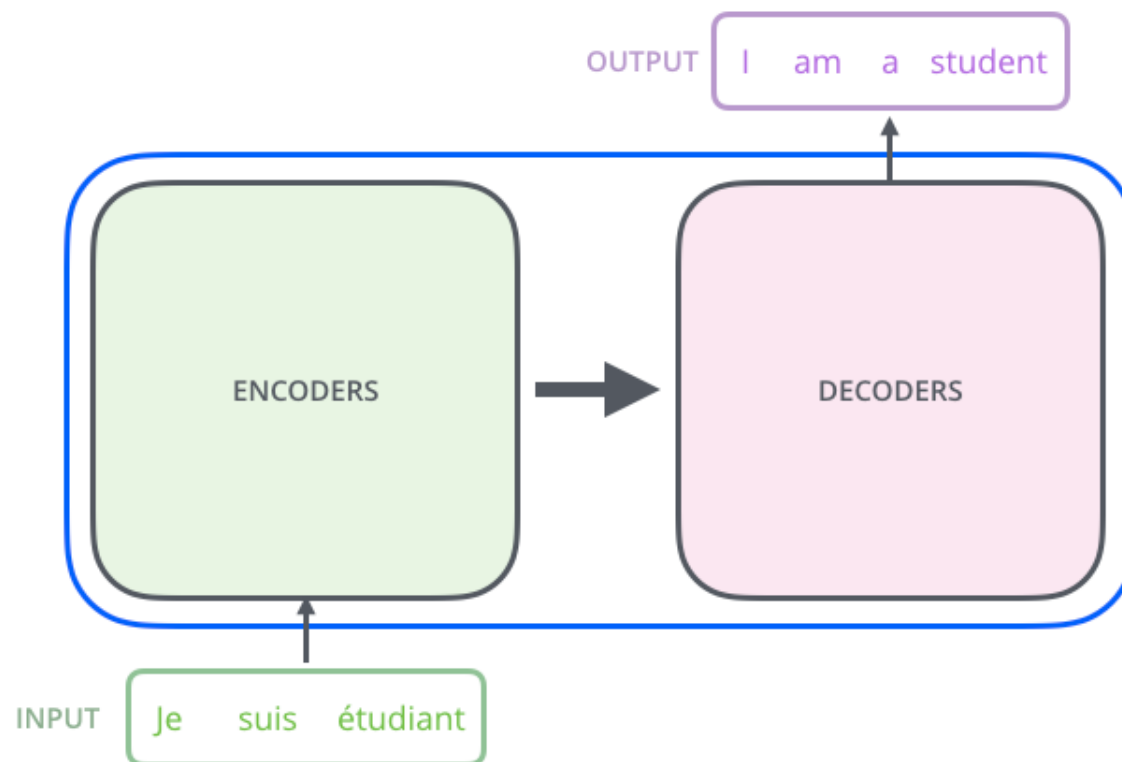
Евгений Соколов

esokolov@hse.ru

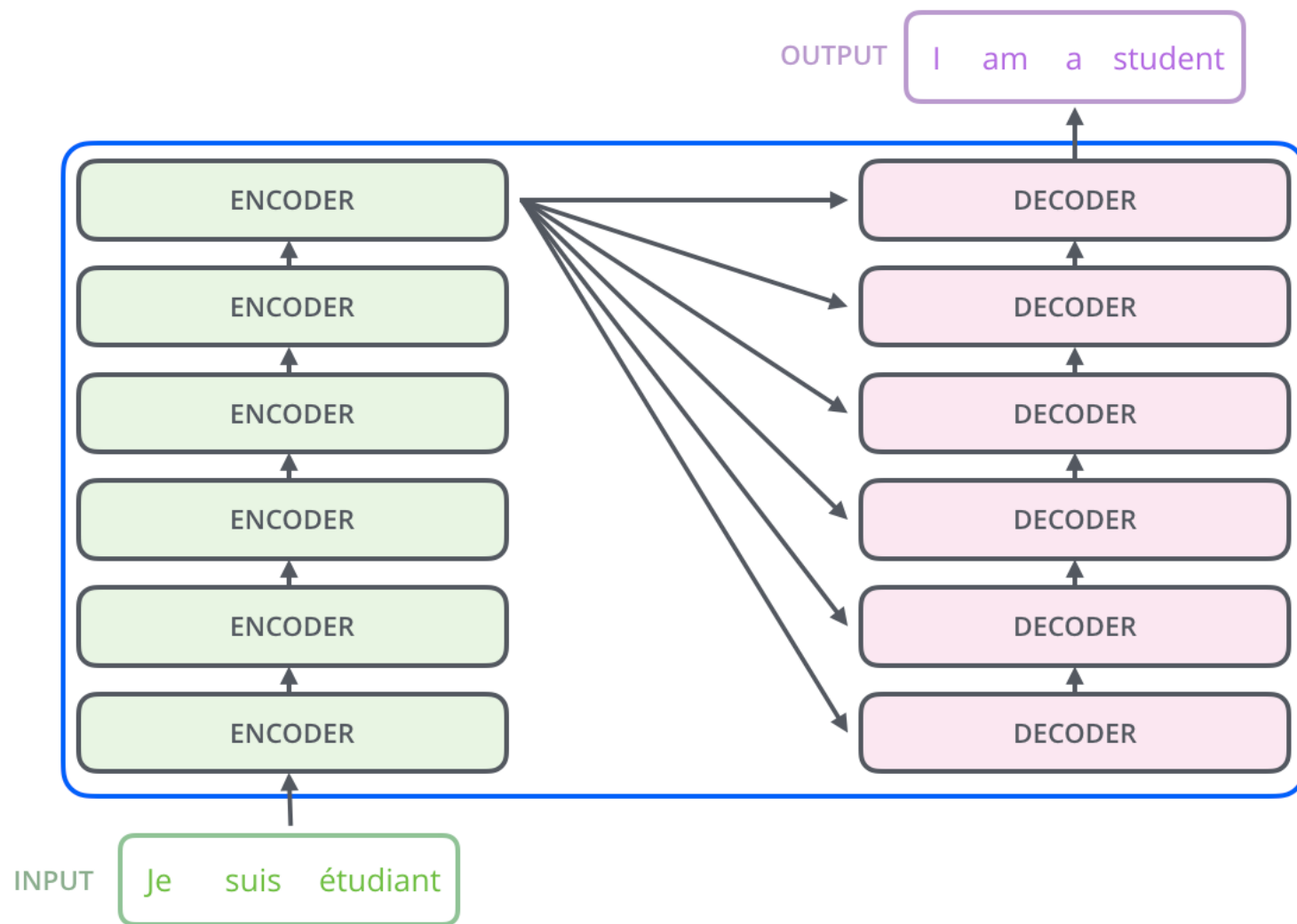
НИУ ВШЭ, 2023

Трансформеры

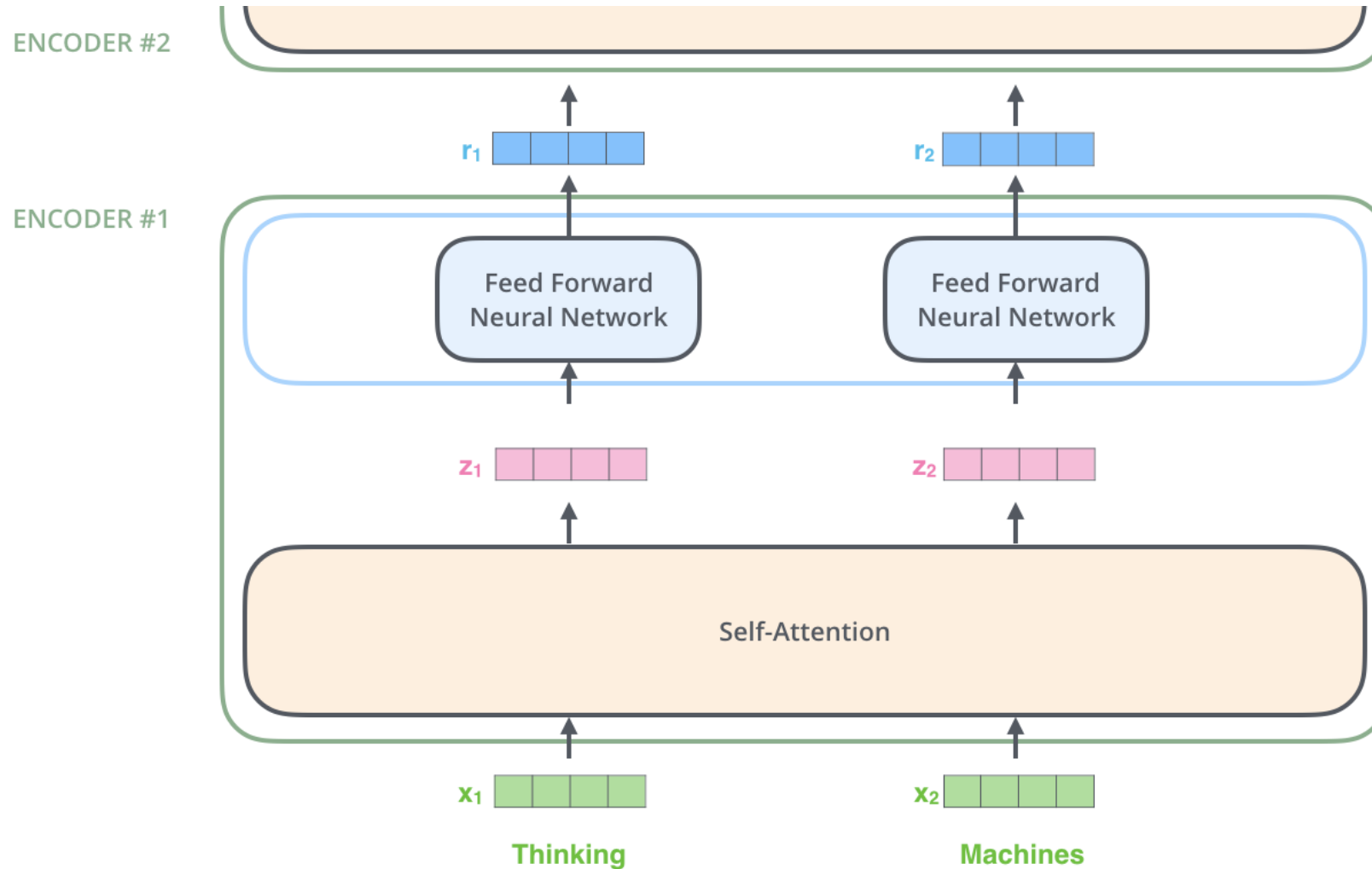
Трансформер



Трансформер



Кодировщик в трансформере



Self-attention

- Будем для каждого слова x_j обучать три вектора:
 - Запрос (query) $q_j = W_Q x_j$
 - Ключ (key) $k_j = W_K x_j$
 - Значение (value) $v_j = W_V x_j$
- «Важность» слова x_i для слова x_j : $\langle q_j, k_i \rangle$

Self-attention

- Вклад слова x_i в новое представление слова x_j :

$$w_{ij} = \frac{\exp\left(\frac{\langle q_j, k_i \rangle}{\sqrt{d}}\right)}{\sum_{p=1}^n \exp\left(\frac{\langle q_j, k_p \rangle}{\sqrt{d}}\right)}$$

- d — размерность векторов q_j и k_i
- n — число слов во входной последовательности

Self-attention

Новое представление слова x_j :

$$z_j = \sum_{p=1}^n w_{pj} v_p$$

Self-attention

То же самое, но в матричном виде:

$$Z = \text{softmax}\left(\frac{1}{\sqrt{d}} Q K^T\right) V$$

The diagram illustrates the self-attention mechanism using matrix dimensions. It shows the calculation of the attention matrix Z as follows:

$$\text{softmax}\left(\frac{\begin{matrix} \text{Q} \\ \begin{array}{|c|c|c|} \hline \square & \square & \square \\ \hline \square & \square & \square \end{array} \times \begin{matrix} \text{K}^T \\ \begin{array}{|c|c|} \hline \square & \square \\ \hline \square & \square \\ \hline \square & \square \end{array} \end{matrix}}{\sqrt{d_k}}\right) \begin{matrix} \text{V} \\ \begin{array}{|c|c|c|} \hline \square & \square & \square \\ \hline \square & \square & \square \end{array} \end{matrix}$$

The result is the attention matrix Z :

$$= \begin{matrix} \text{Z} \\ \begin{array}{|c|c|c|} \hline \square & \square & \square \\ \hline \square & \square & \square \end{array} \end{matrix}$$

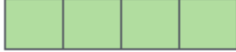
Each matrix is represented by a 2x3 grid of squares. The matrix Q is purple, K^T is orange, V is blue, and the resulting Z is pink. The division by $\sqrt{d_k}$ is indicated by a horizontal line under the multiplication.

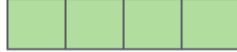
Input

Thinking

Machines

Embedding

x_1 

x_2 

Queries

q_1 

q_2 

Keys

k_1 

k_2 

Values

v_1 

v_2 

Score

$q_1 \cdot k_1 = 112$

$q_1 \cdot k_2 = 96$

Divide by 8 ($\sqrt{d_k}$)

14

12

Softmax

0.88

0.12

Softmax

X

Value

v_1 

v_2 

Sum

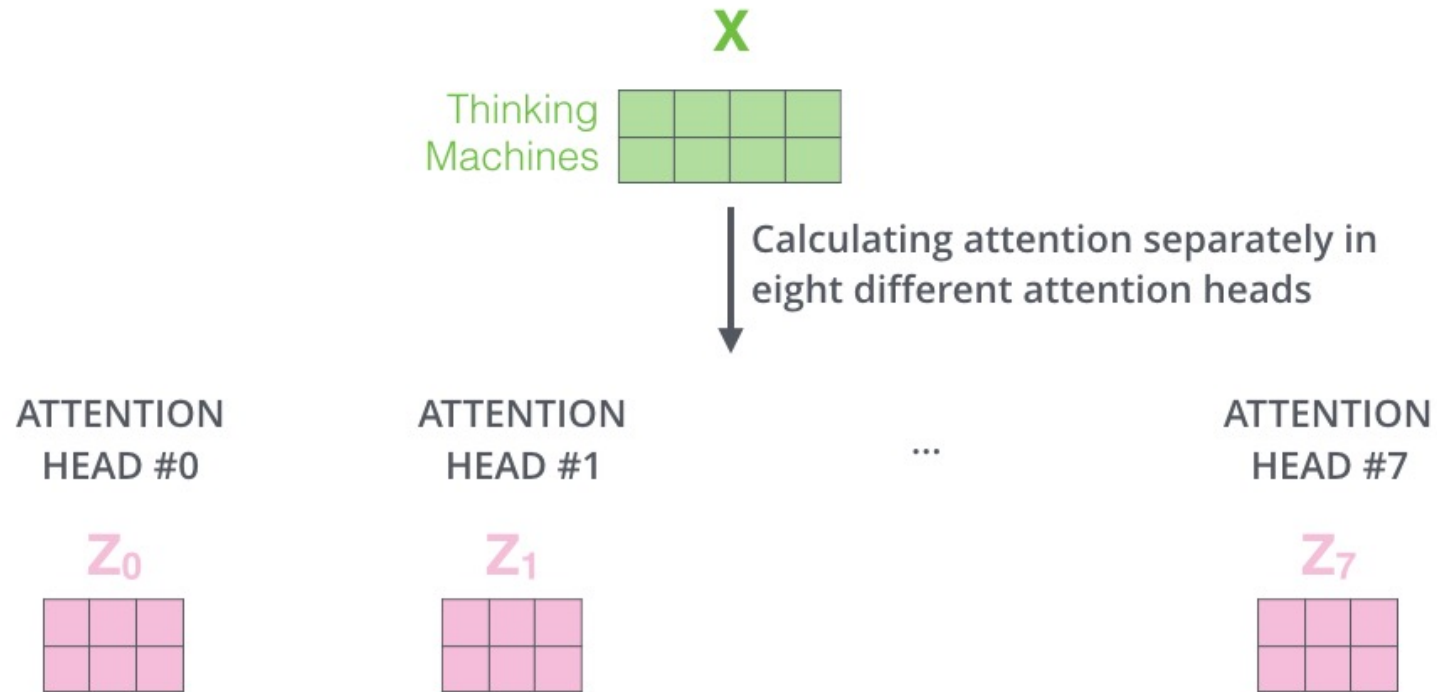
z_1 

z_2 

Multi-headed Self-attention

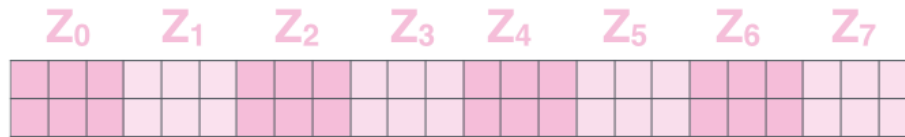
- Механизм работает хорошо
- Если напастать его, станет ещё лучше, наверное

Multi-headed Self-attention



Multi-headed Self-attention

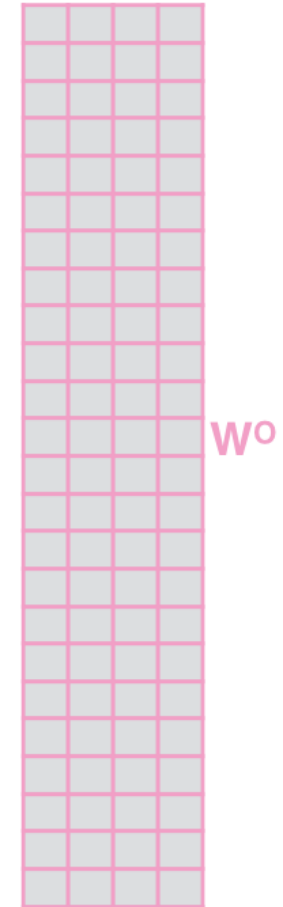
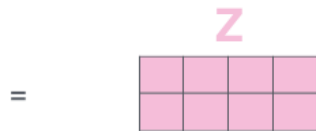
1) Concatenate all the attention heads



2) Multiply with a weight matrix W^O that was trained jointly with the model

\times

3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

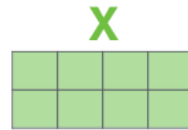


Multi-headed Self-attention

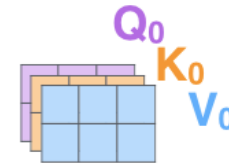
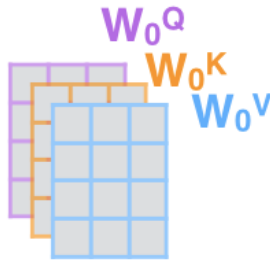
1) This is our
input sentence*

Thinking
Machines

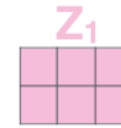
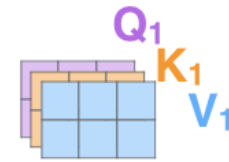
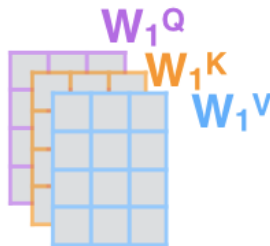
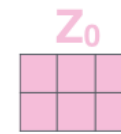
2) We embed
each word*



3) Split into 8 heads.
We multiply X or
 R with weight matrices



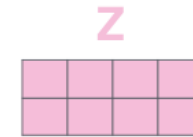
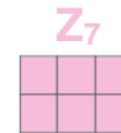
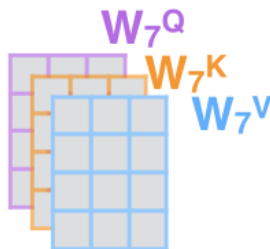
5) Concatenate the resulting Z matrices,
then multiply with weight matrix W^O to
produce the output of the layer



...

...

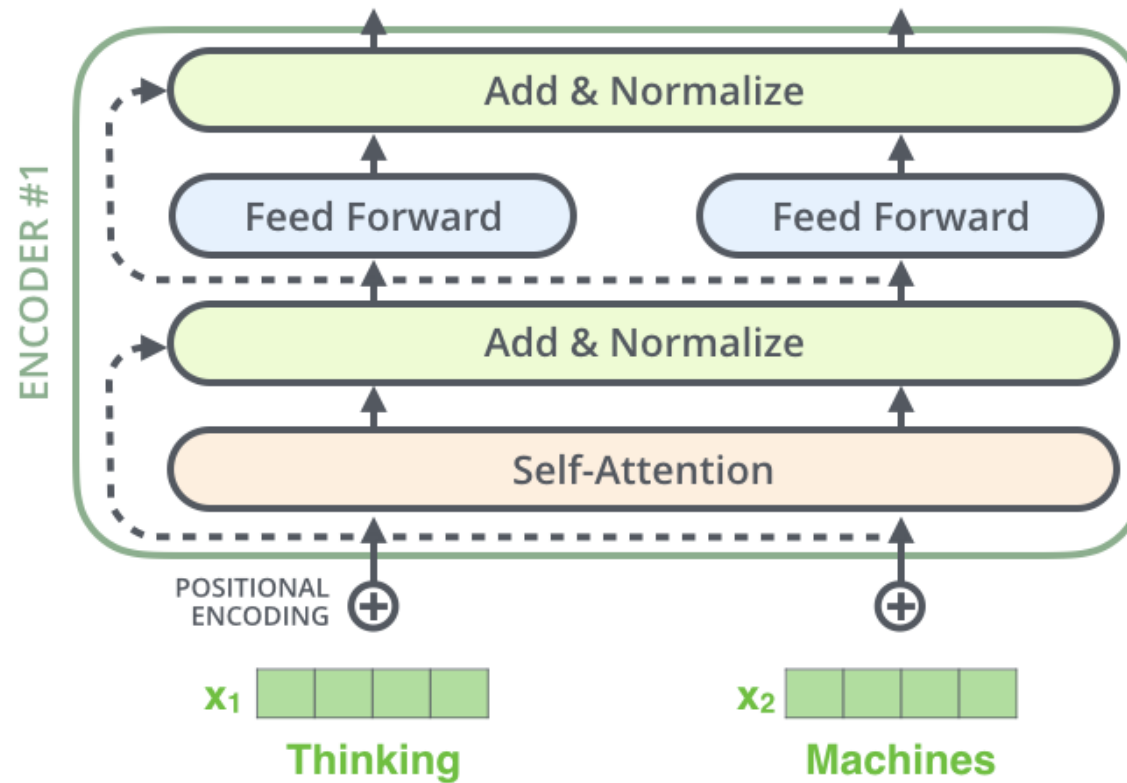
...



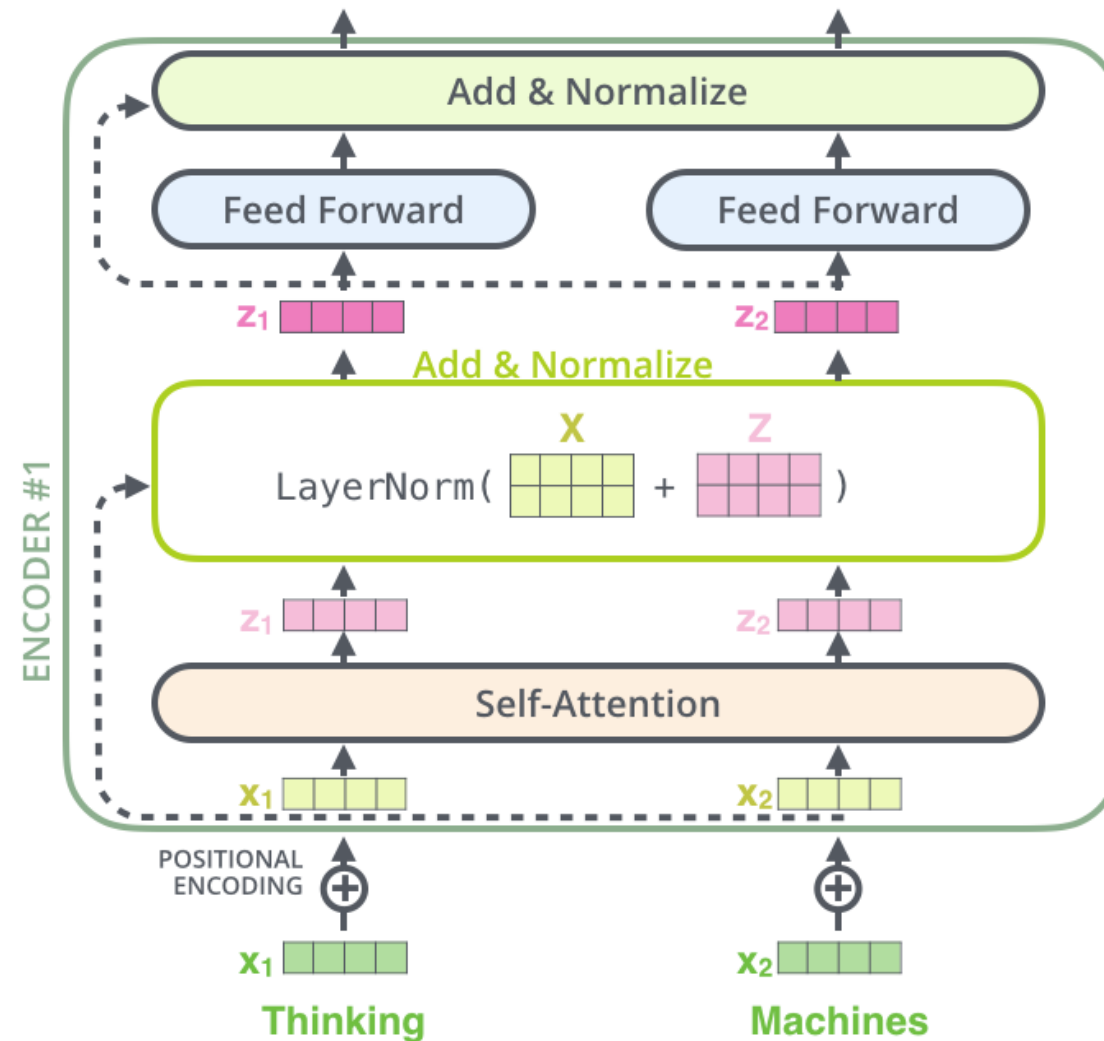
* In all encoders other than #0,
we don't need embedding.
We start directly with the output
of the encoder right below this one



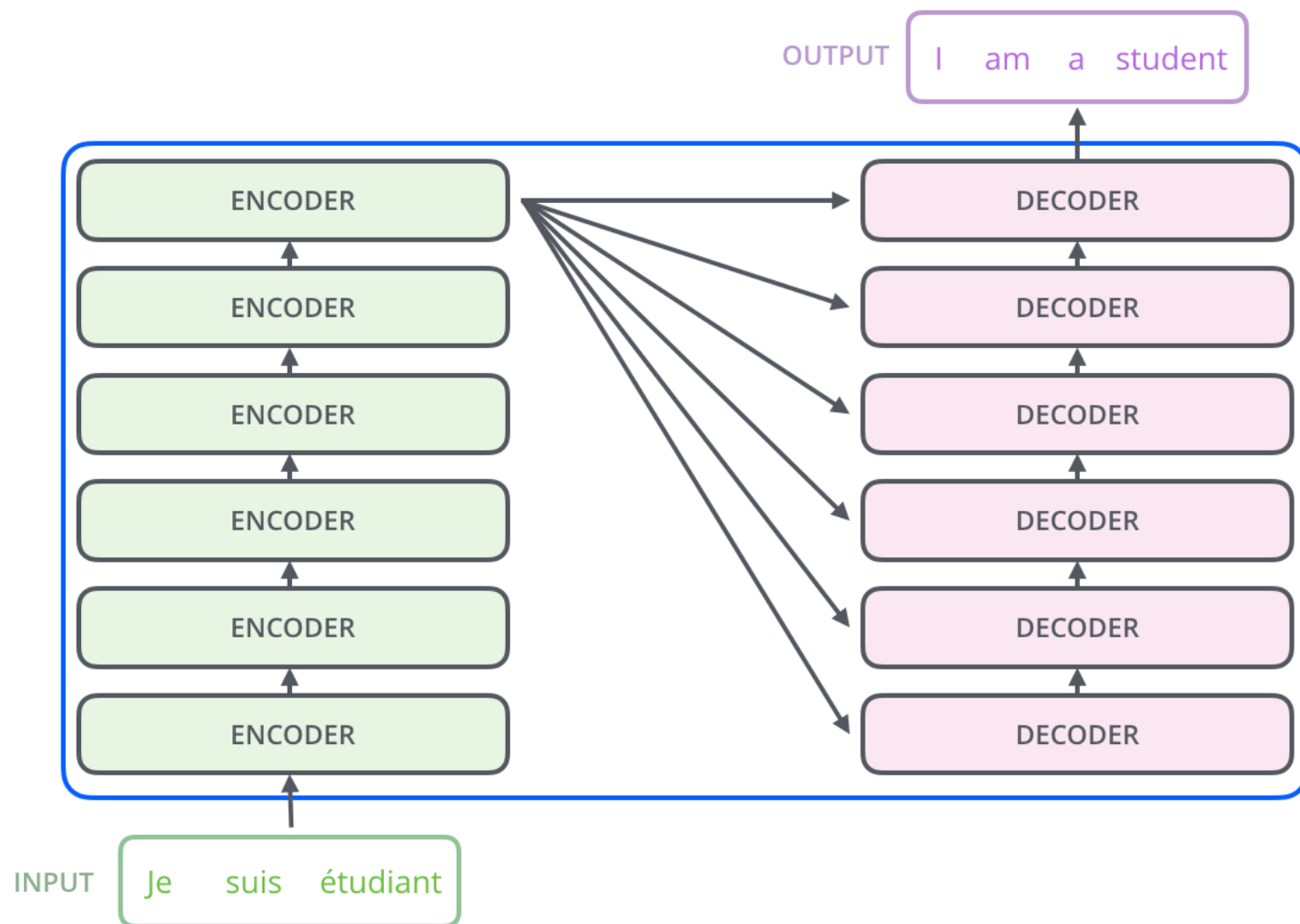
Кодировщик в трансформере



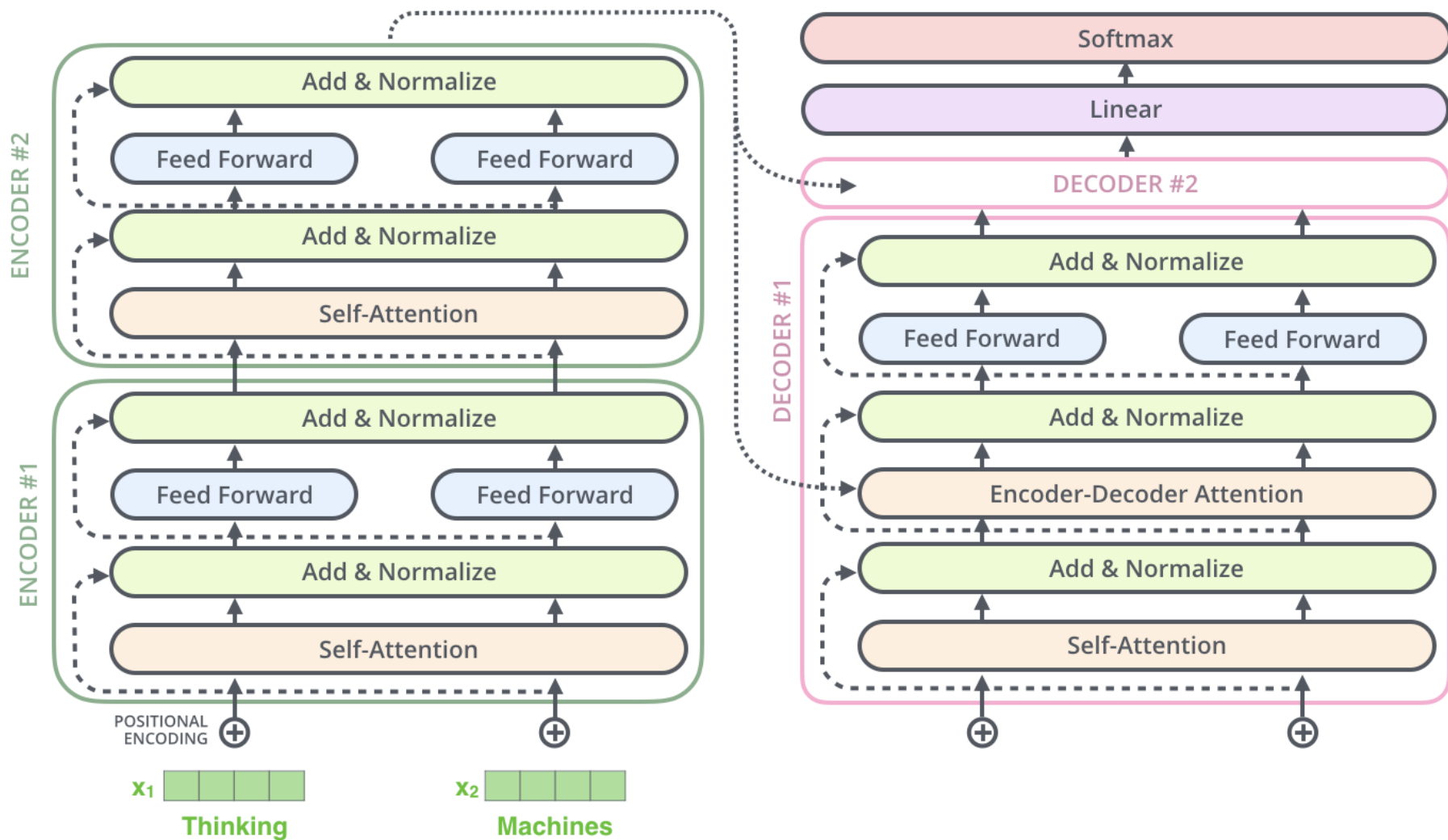
Кодировщик в трансформере



Трансформер



Деодировщик в трансформере



Encoder-Decoder Attention

- Векторы k_j и v_j получаются домножением матриц K_{encdec} и V_{encdec} соответственно на выходы последнего кодировщика
- Векторы q_j получаются стандартным образом из предыдущего слоя декодировщика

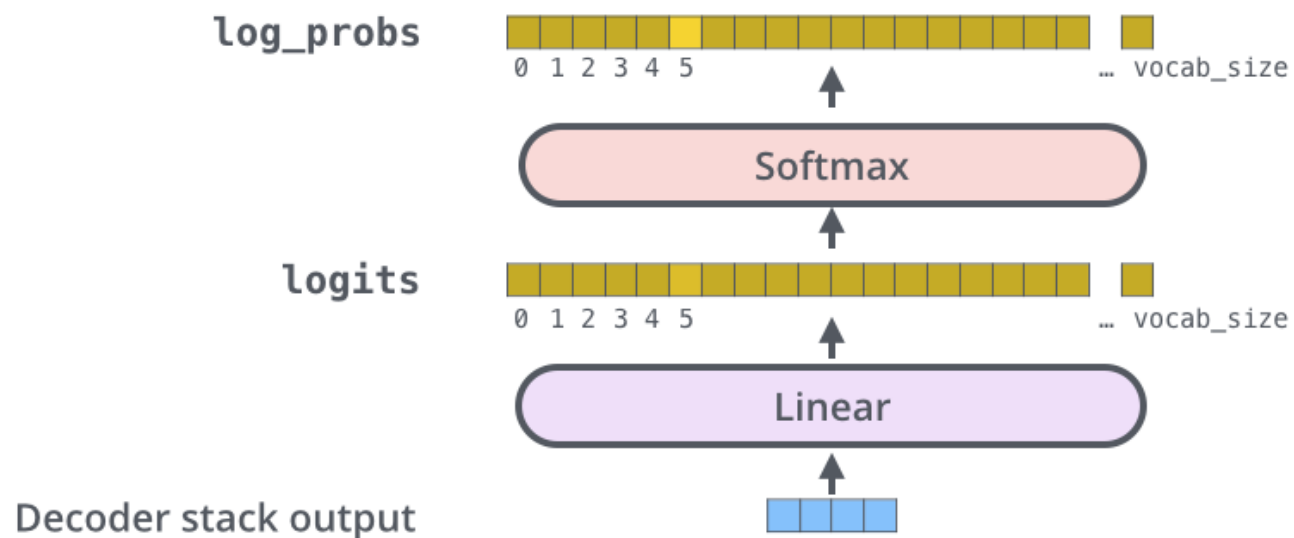
Self-attention в декодировщике

- Разрешается использовать только предыдущие слова

Выходной блок

Which word in our vocabulary
is associated with this index?

Get the index of the cell
with the highest value
(**argmax**)



Выходной блок

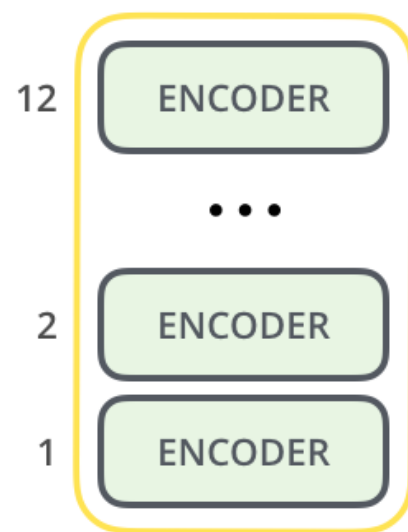
В случае с машинным переводом — «авторегрессионное» применение:

- Сначала декодировщик выдаёт одно слово
- Затем два (первое подаётся как вход)
- Затем три (первые два подаются ему как вход)
- И т.д.

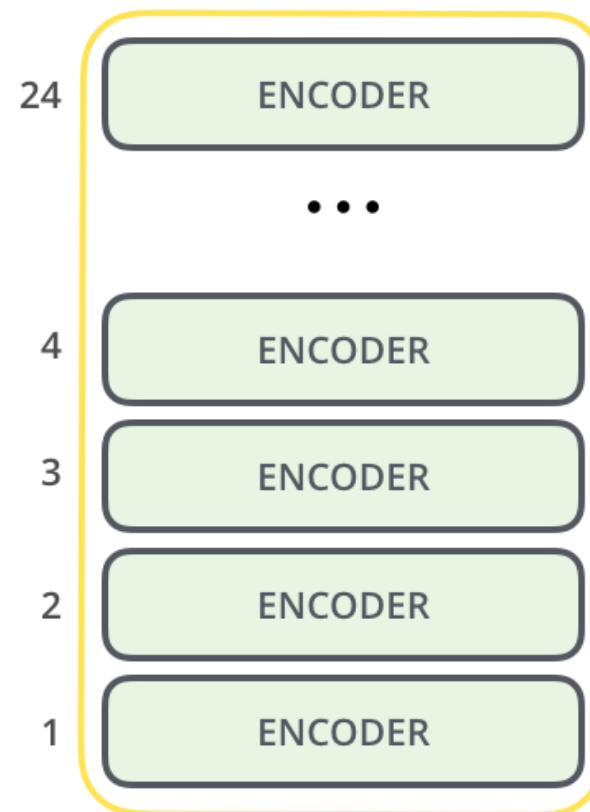
Число параметров

	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	

BERT: архитектура

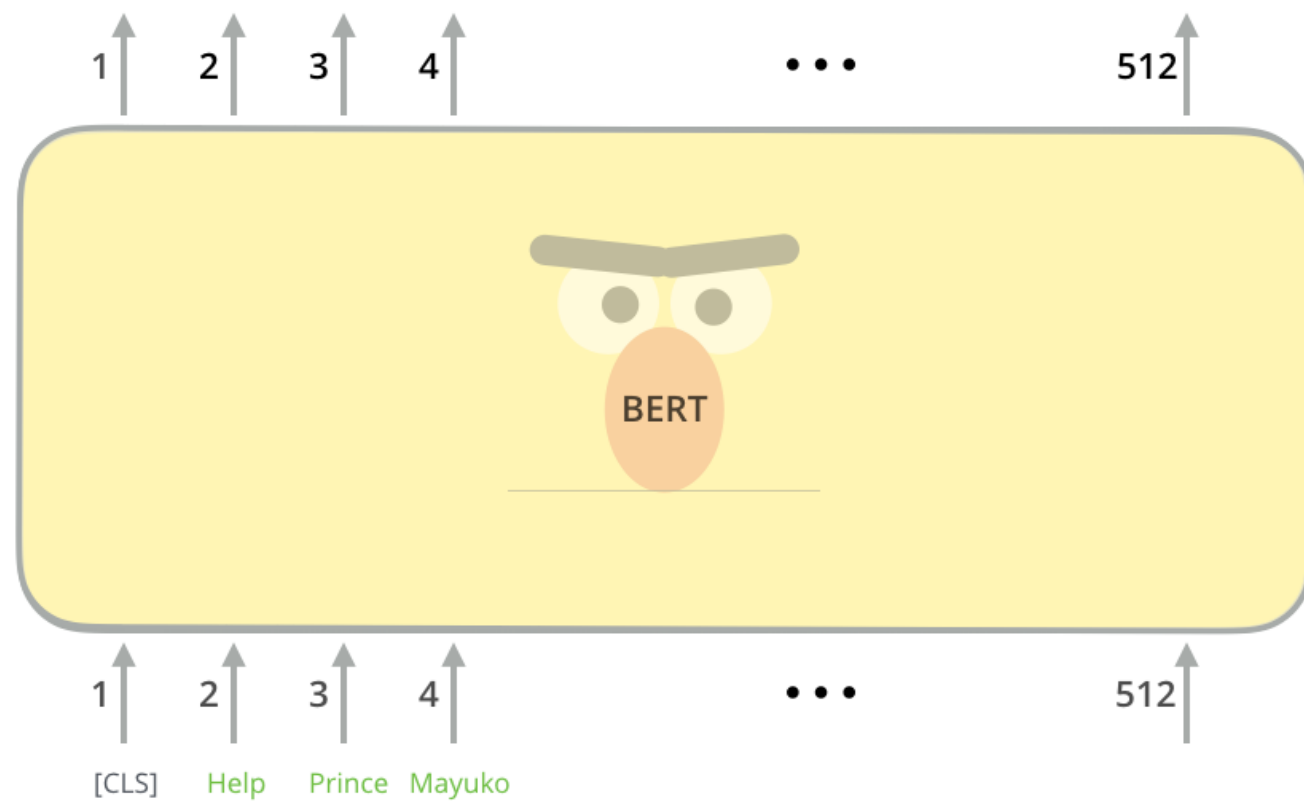


BERT_{BASE}

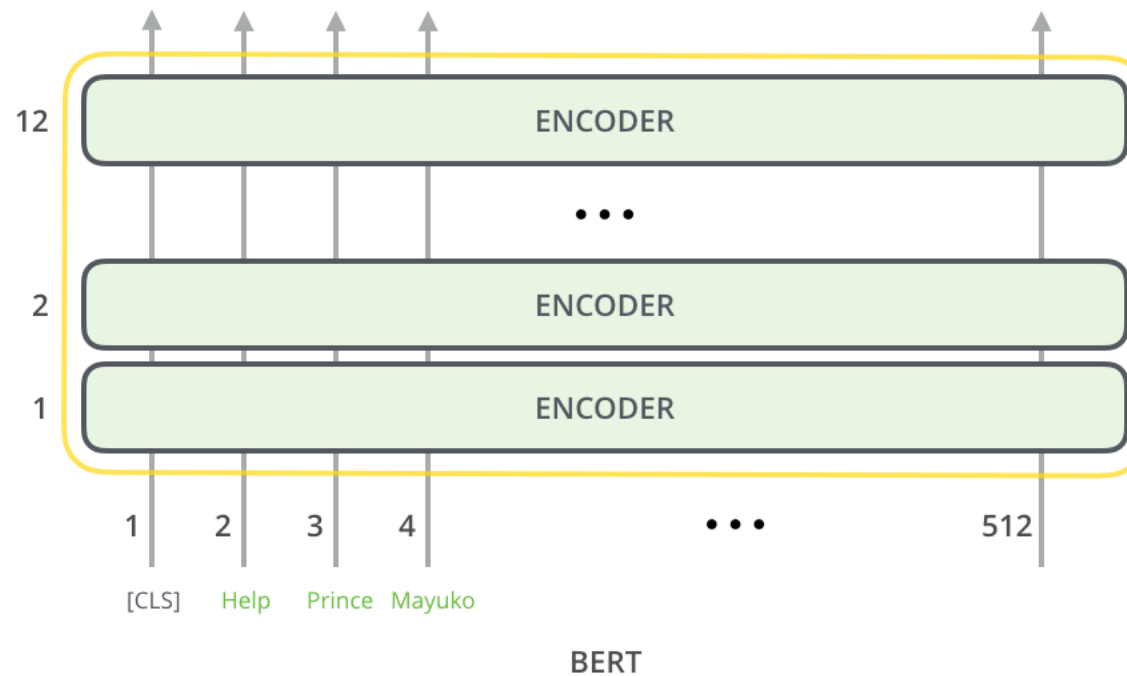


BERT_{LARGE}

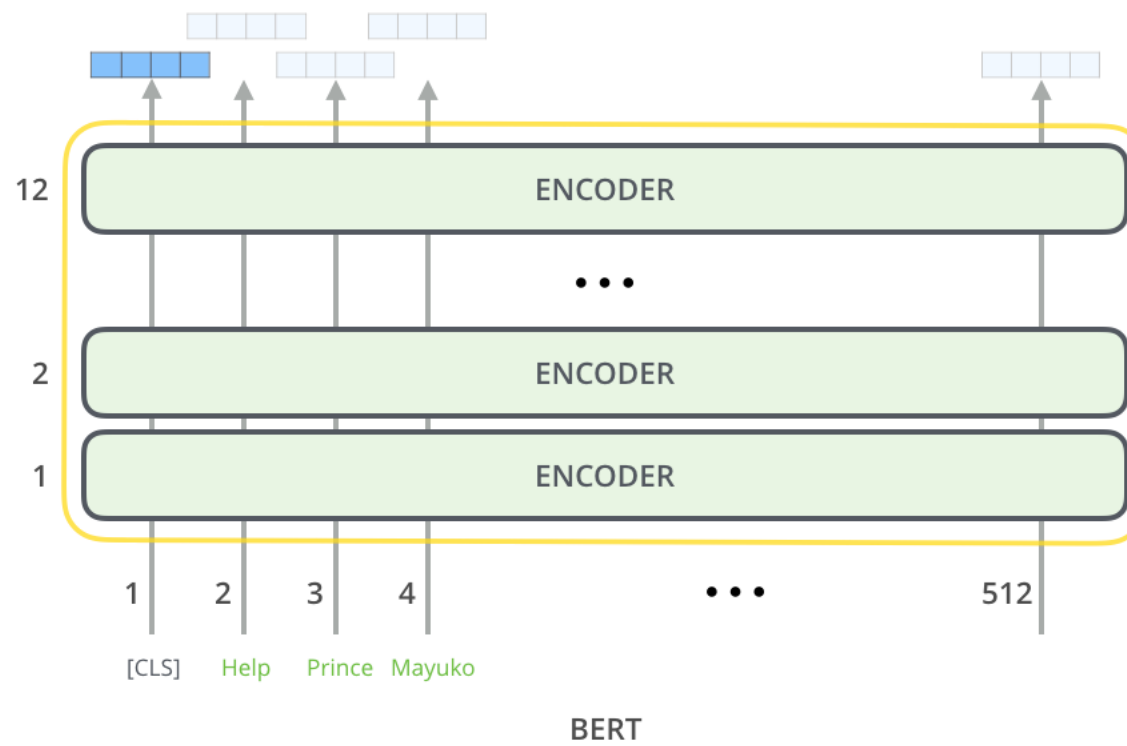
BERT: архитектура



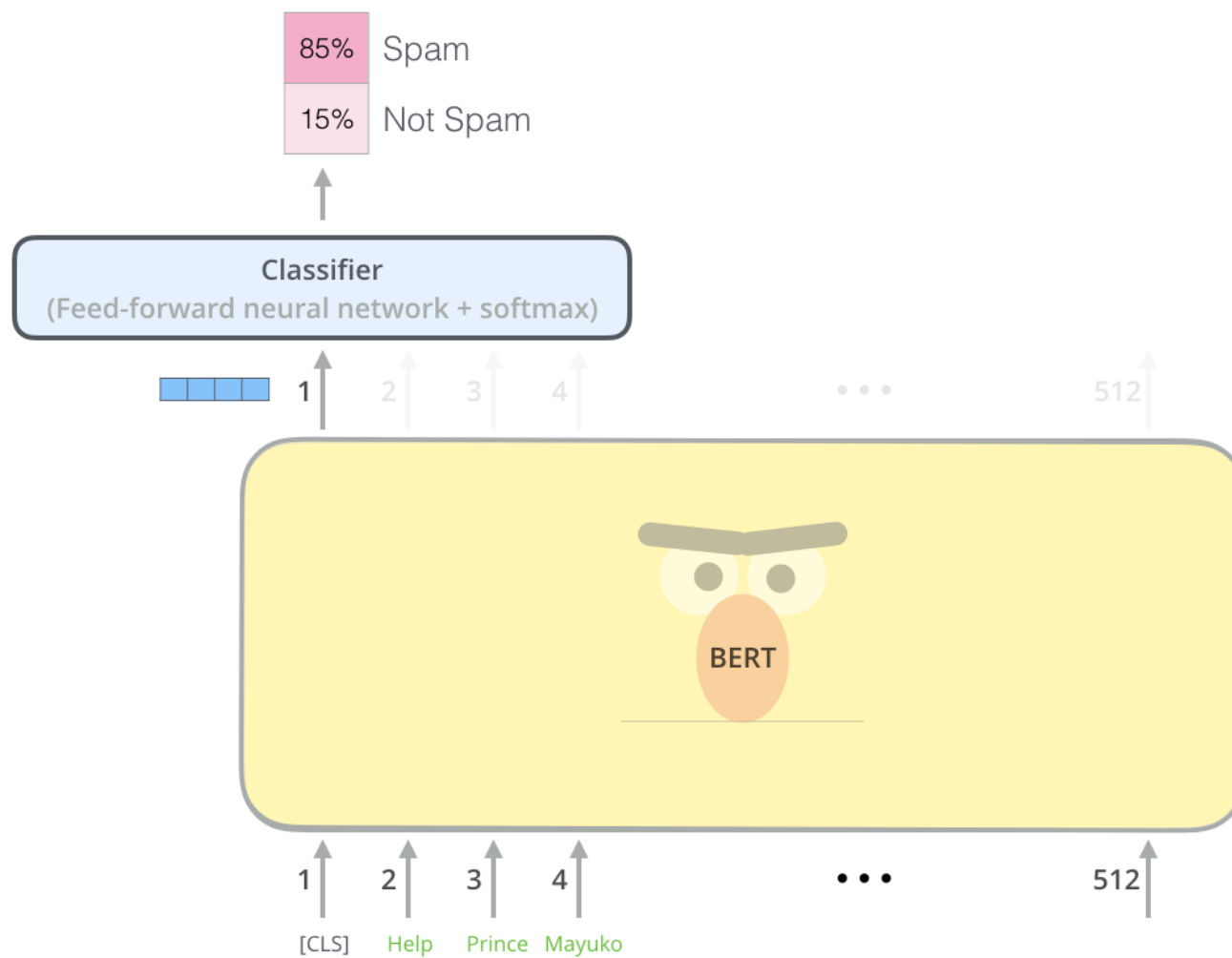
BERT: архитектура



BERT: архитектура

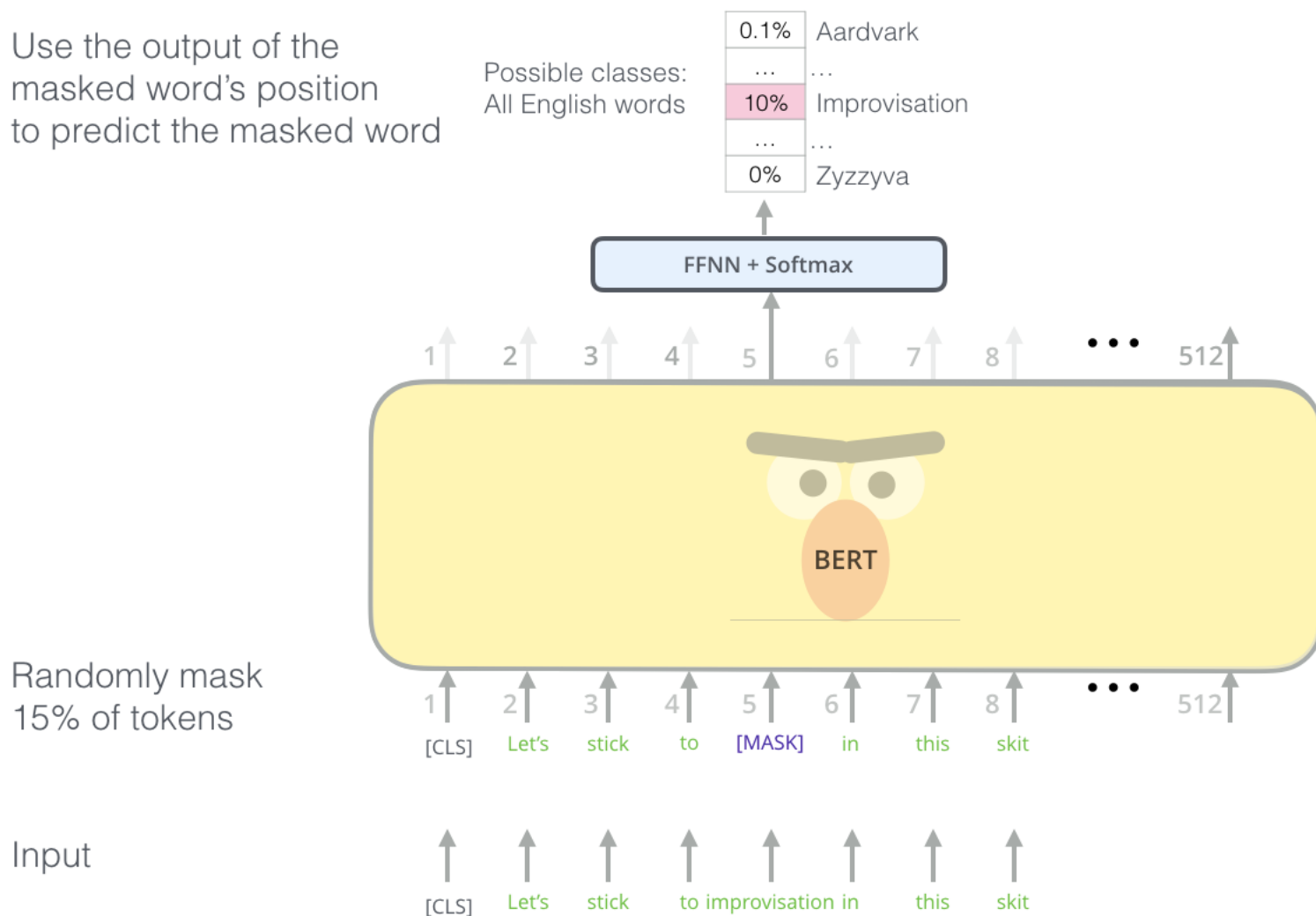


BERT: архитектура

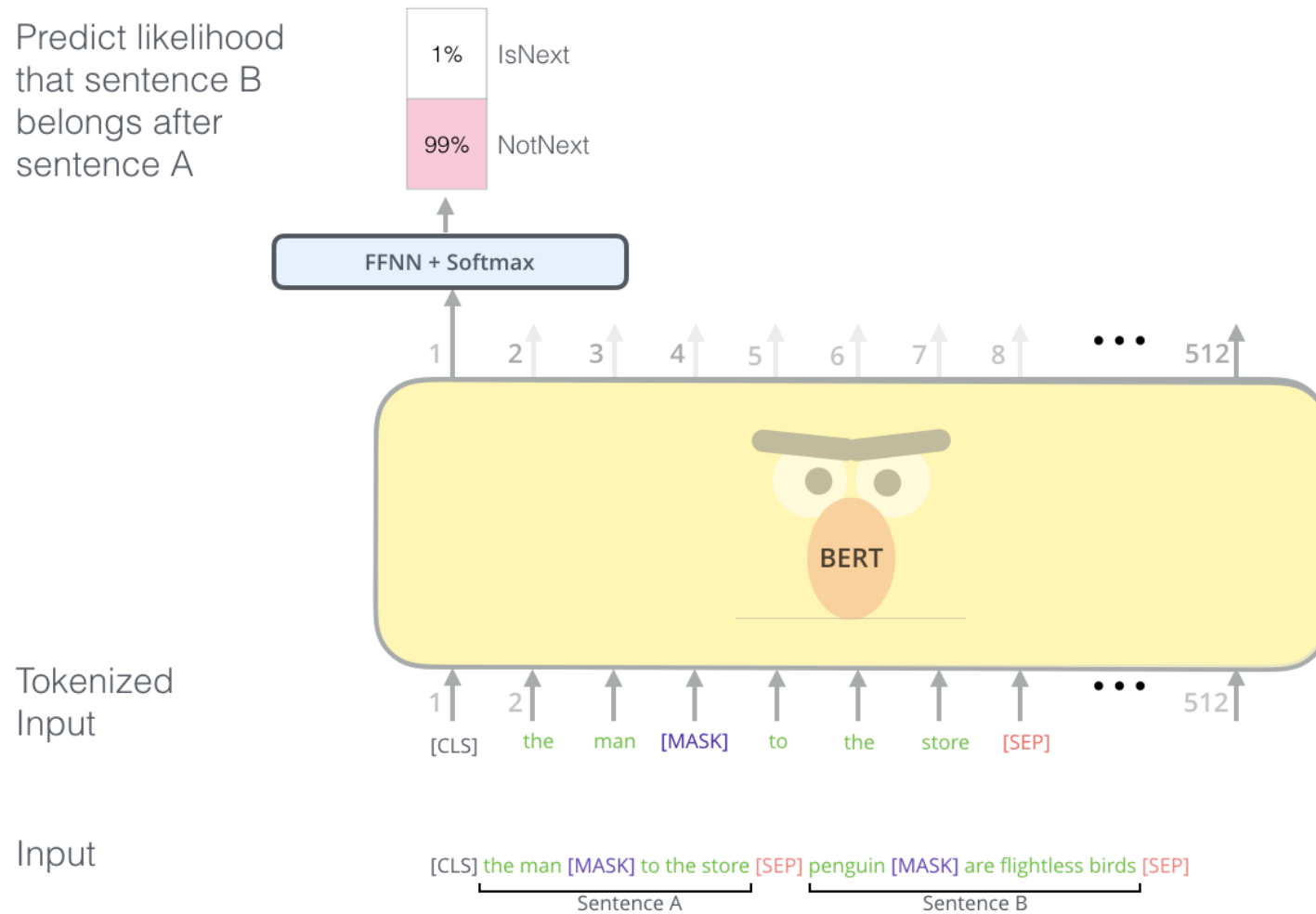


BERT: предобучение

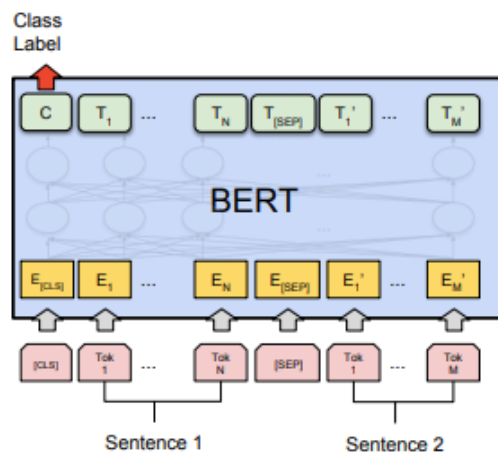
Use the output of the masked word's position to predict the masked word



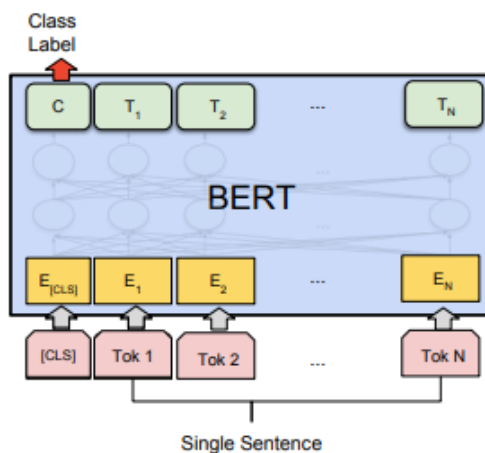
BERT: предобучение



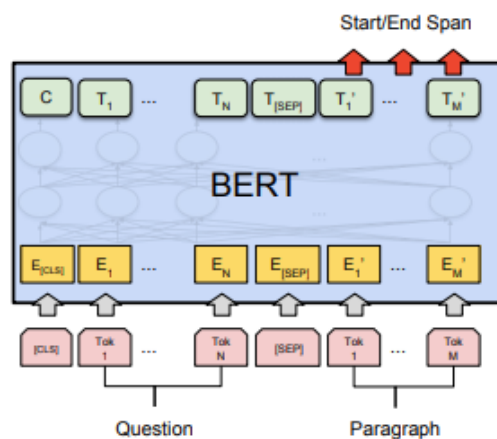
BERT: применение



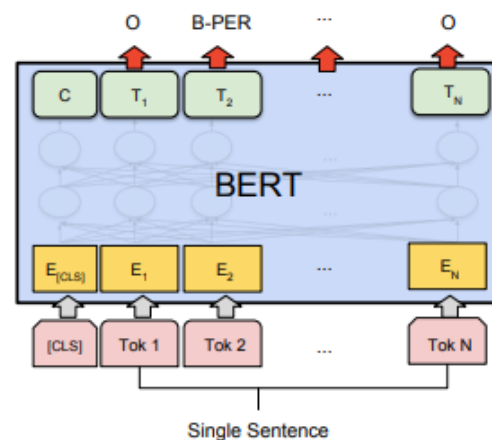
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



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

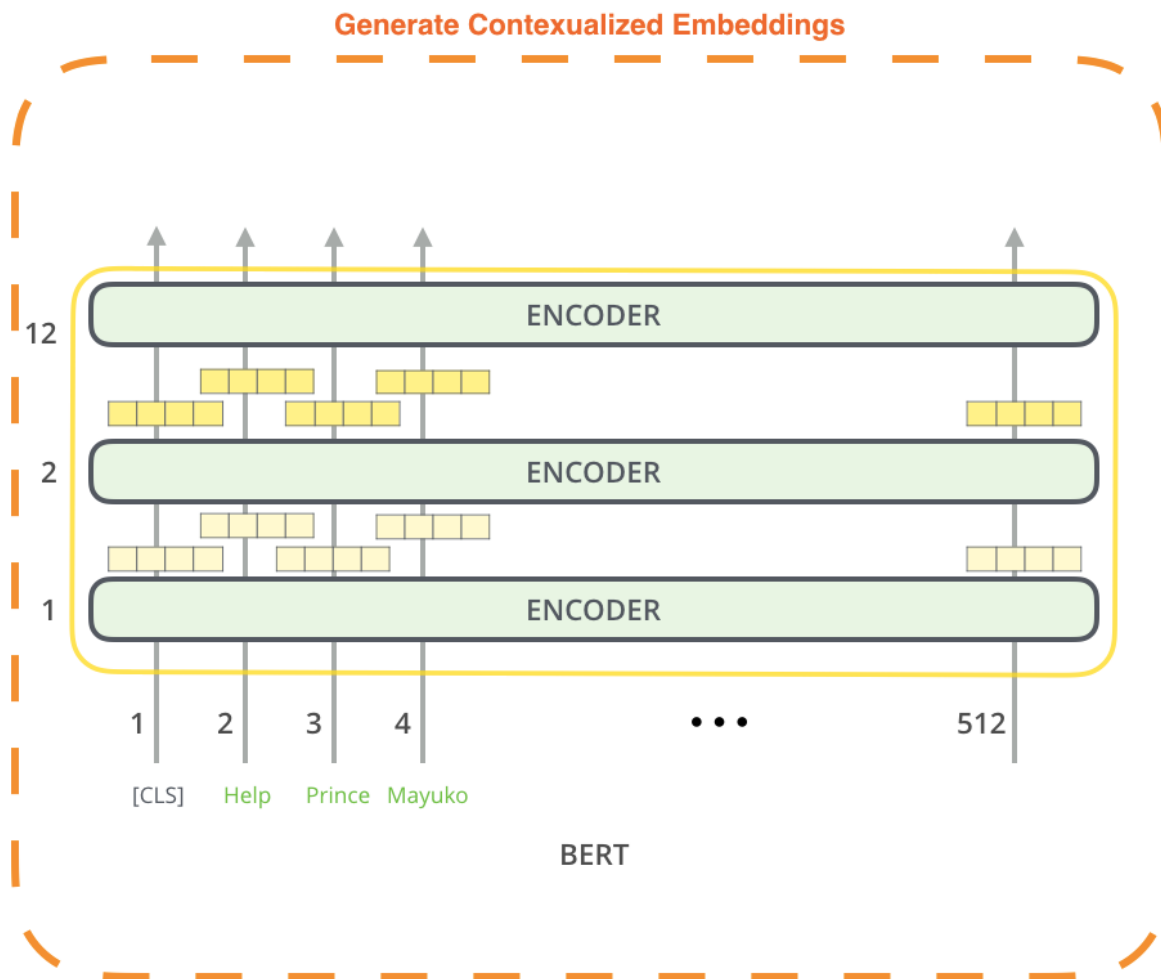


(c) Question Answering Tasks:
SQuAD v1.1

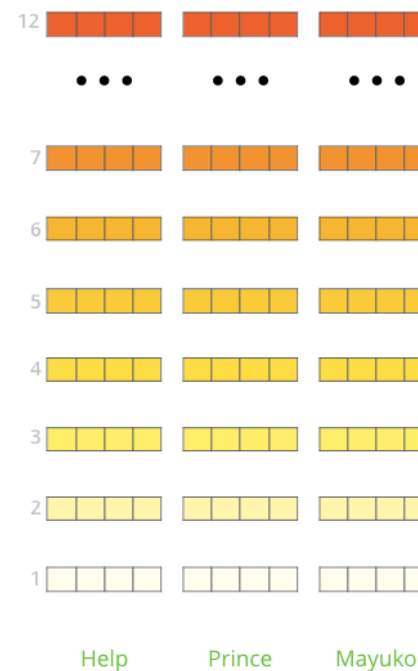


(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

BERT: применение

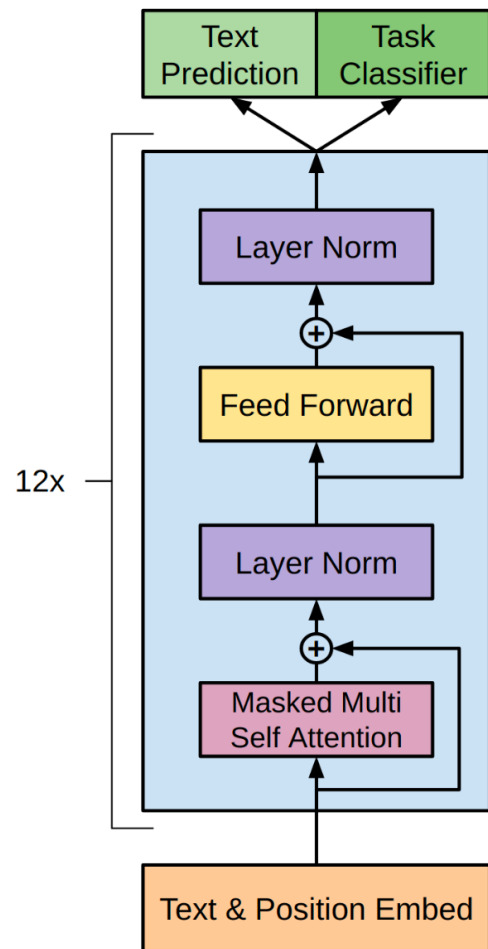


The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

GPT-2



Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

GPT-3

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

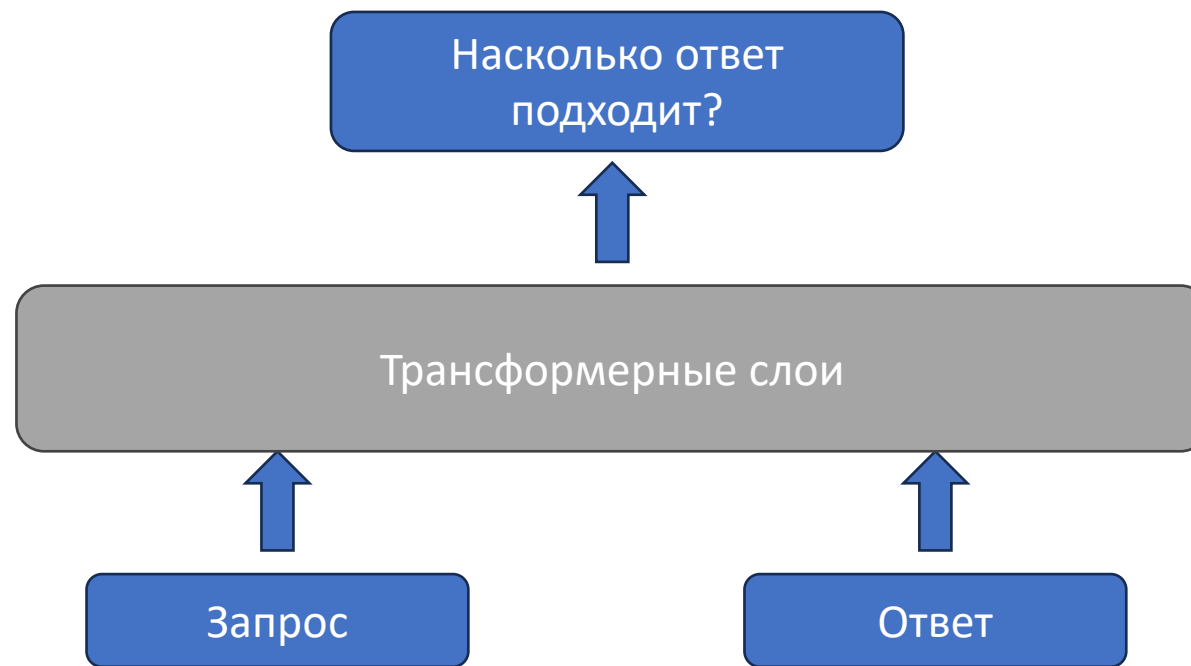
GPT-3

Задача: предсказать следующее слово

Модель: трансформер

Данные: **все тексты, которые можем найти**

Оценивающая модель



Оценивающая модель

Задача: определить «хорошесть» ответа на запрос

Модель: трансформер

Данные: варианты ответов, размеченные ассессорами

InstructGPT

