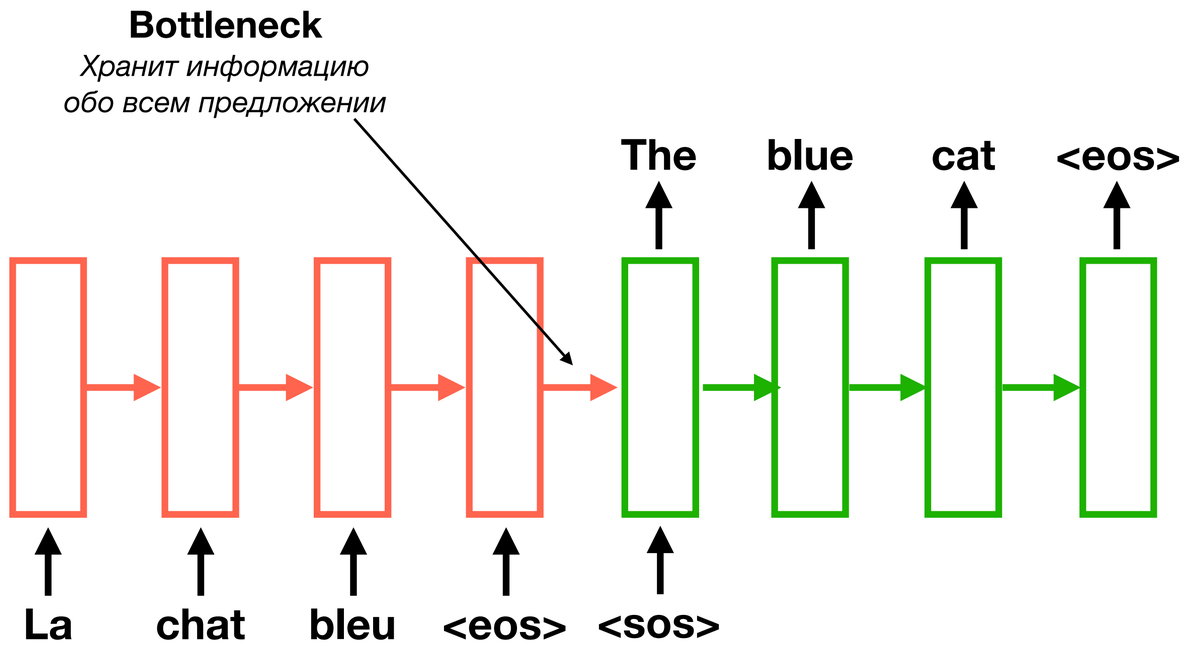
Машинное обучение

Lecture 2 - Self-Attention, Transformers, ELMo, BERT

Власов Кирилл Вячеславович

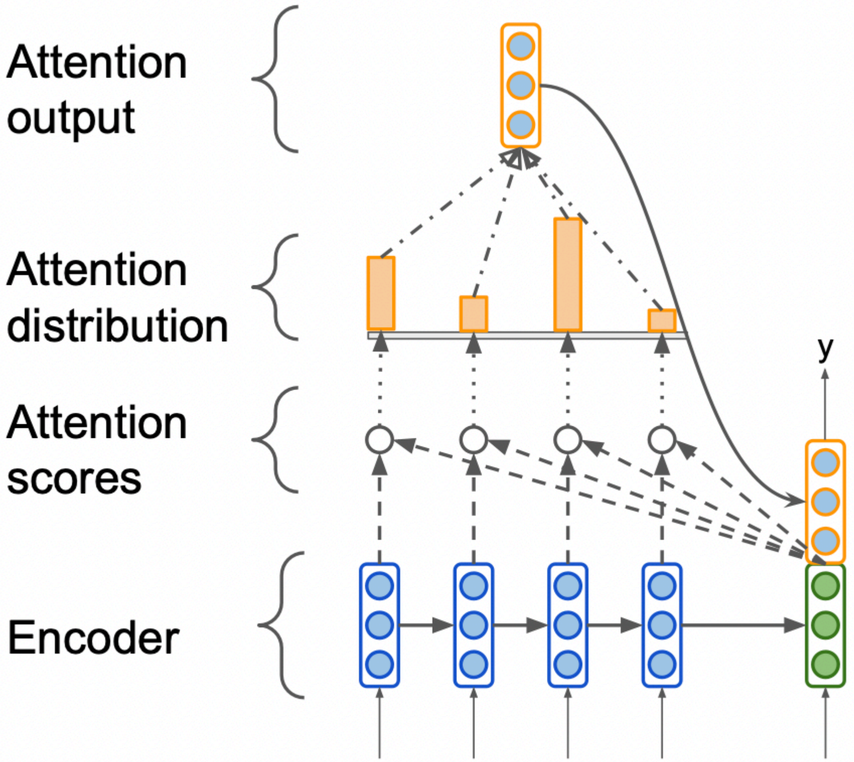
2021

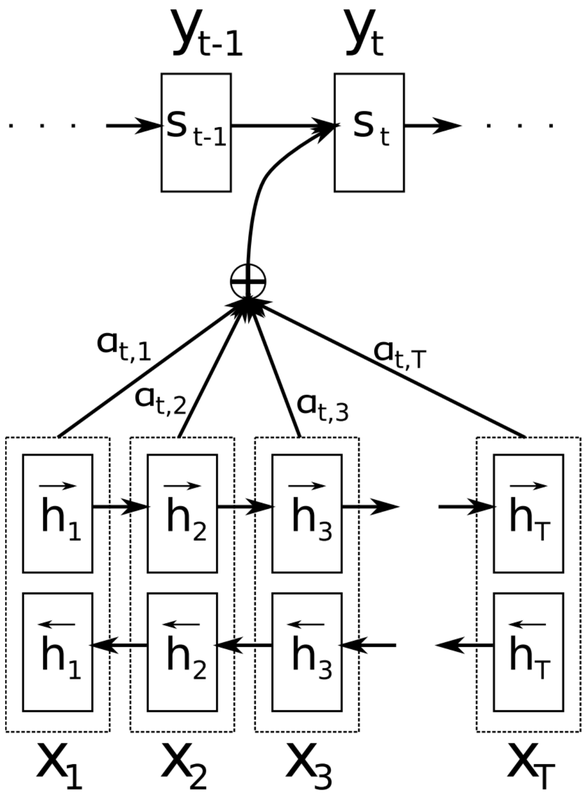
Recap: Sequence to sequence



Sequence to Sequence Learning with Neural Networks

Recap: Attention





Скрытый слой RNN

$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

Взвешенная сумма

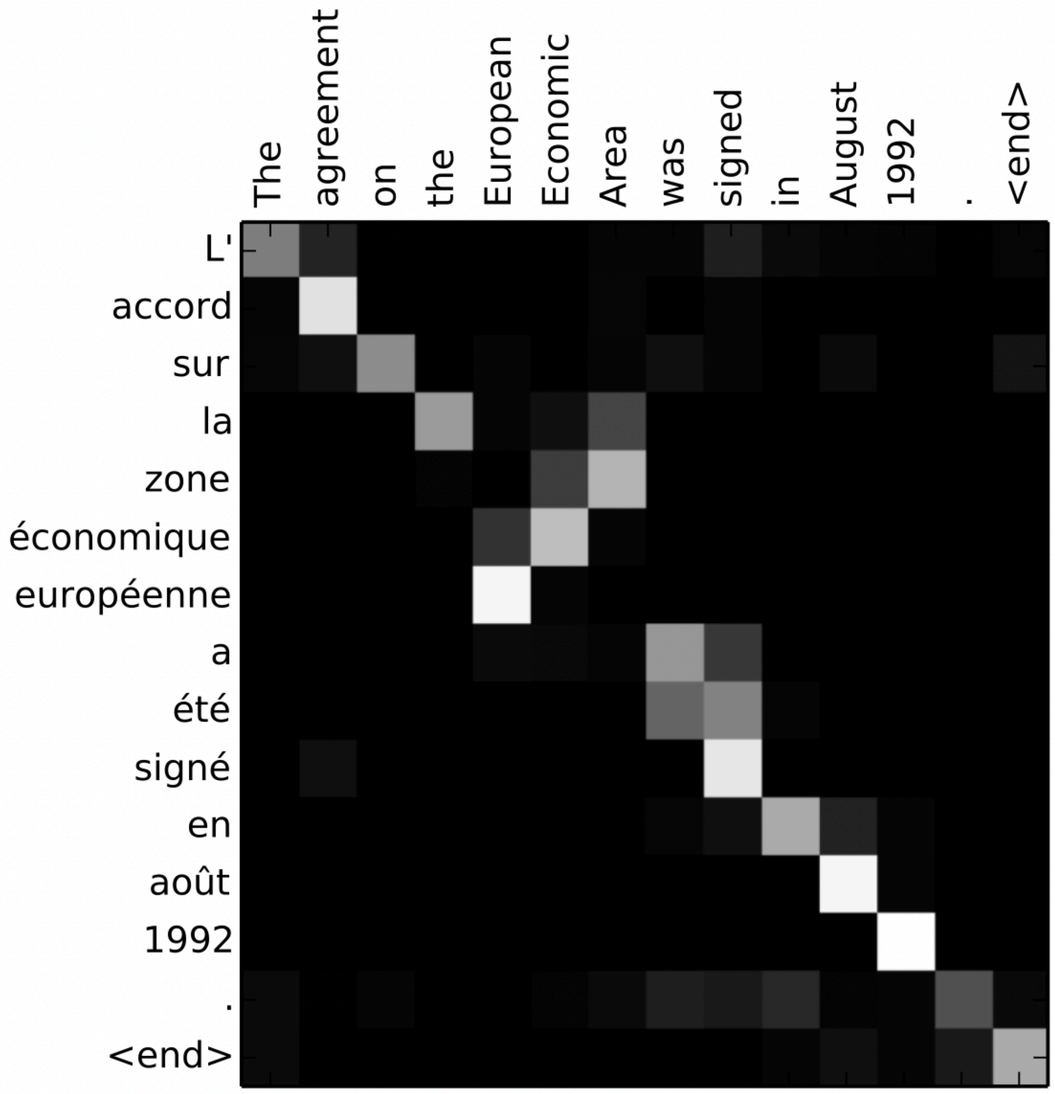
$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

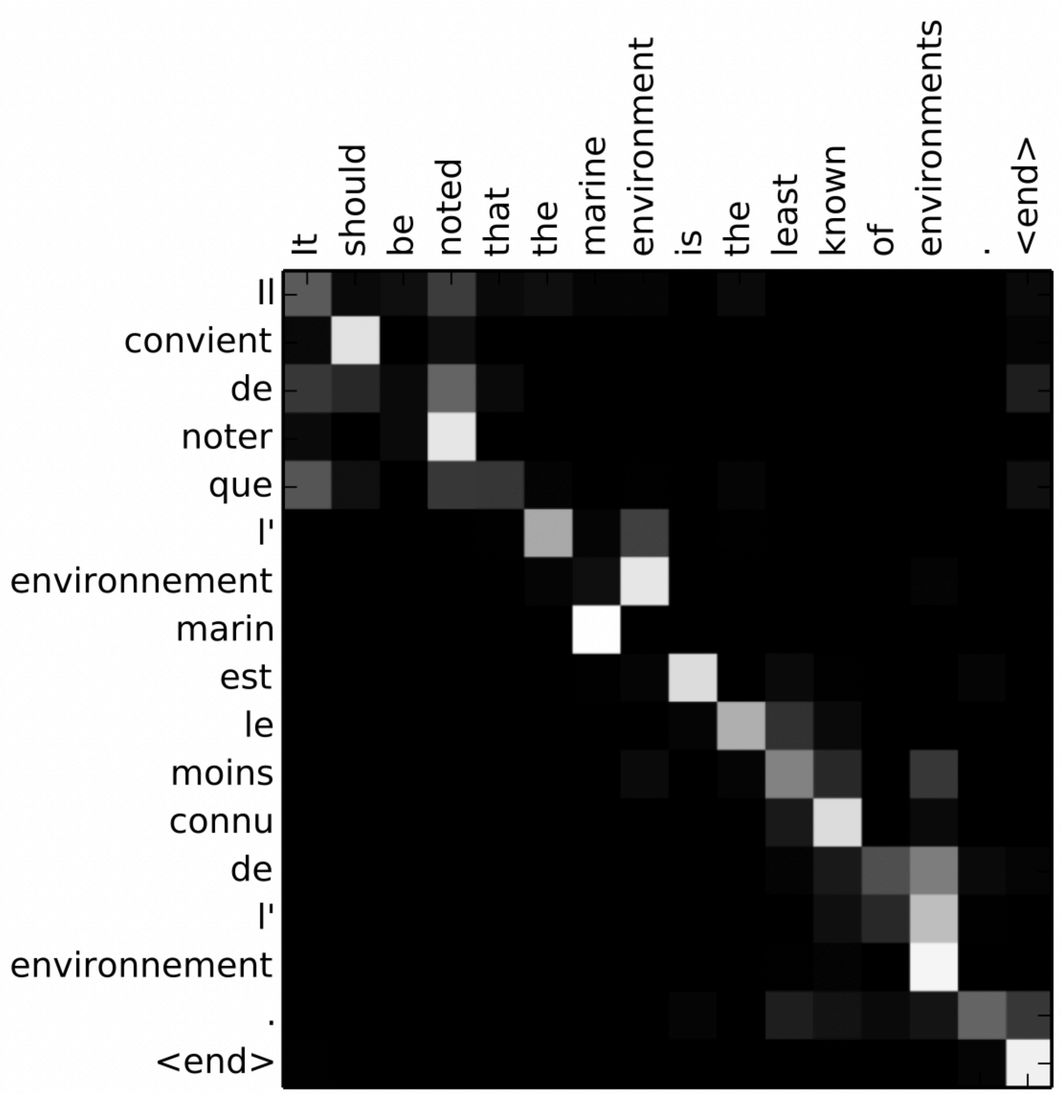
$$e_{ij} = a(s_{i-1}, h_j)$$

https://github.com/girafe-ai/ml-mipt/blob/advanced_f20/week1_04_Transformer_and_PoS_Tagging/ml-mipt_f20_lect104_Self_Attention_and_Transformer.pdf

Recap: Attention



(a)



(b)

А что если без RNN?

А что если без RNN?

Но ведь нам нужно обрабатывать последовательности…

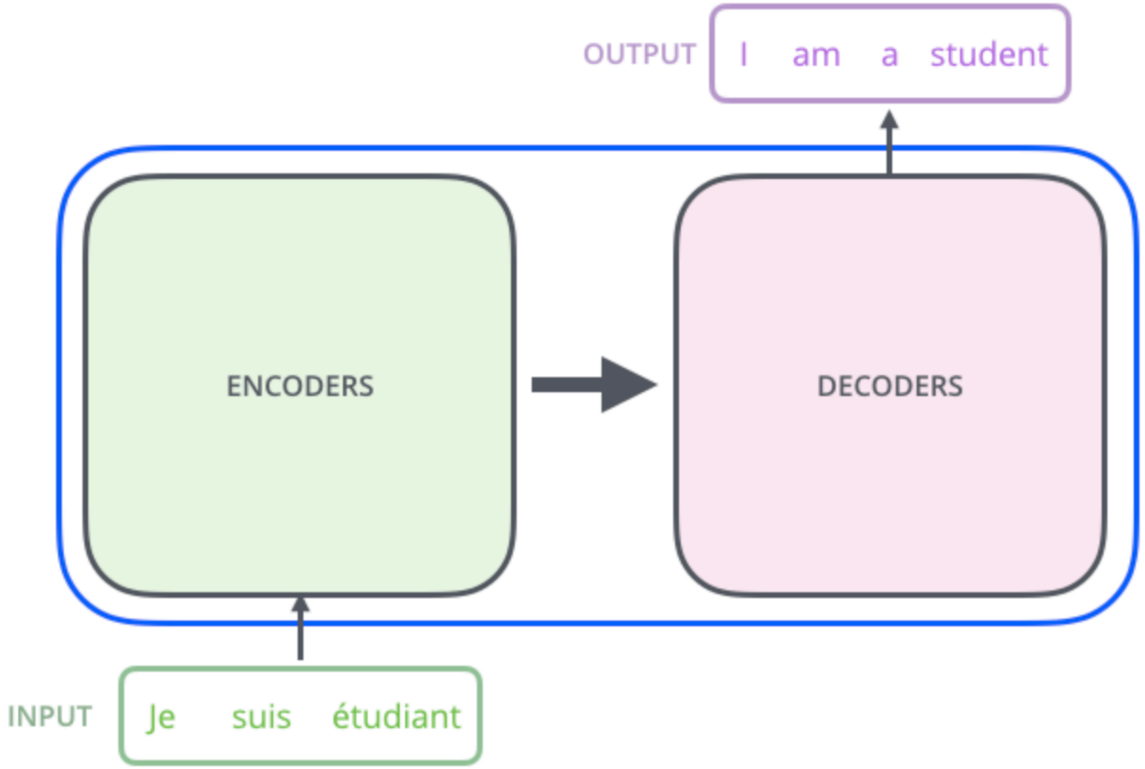
Self-Attention

The Illustrated Transformer

Self-Attention

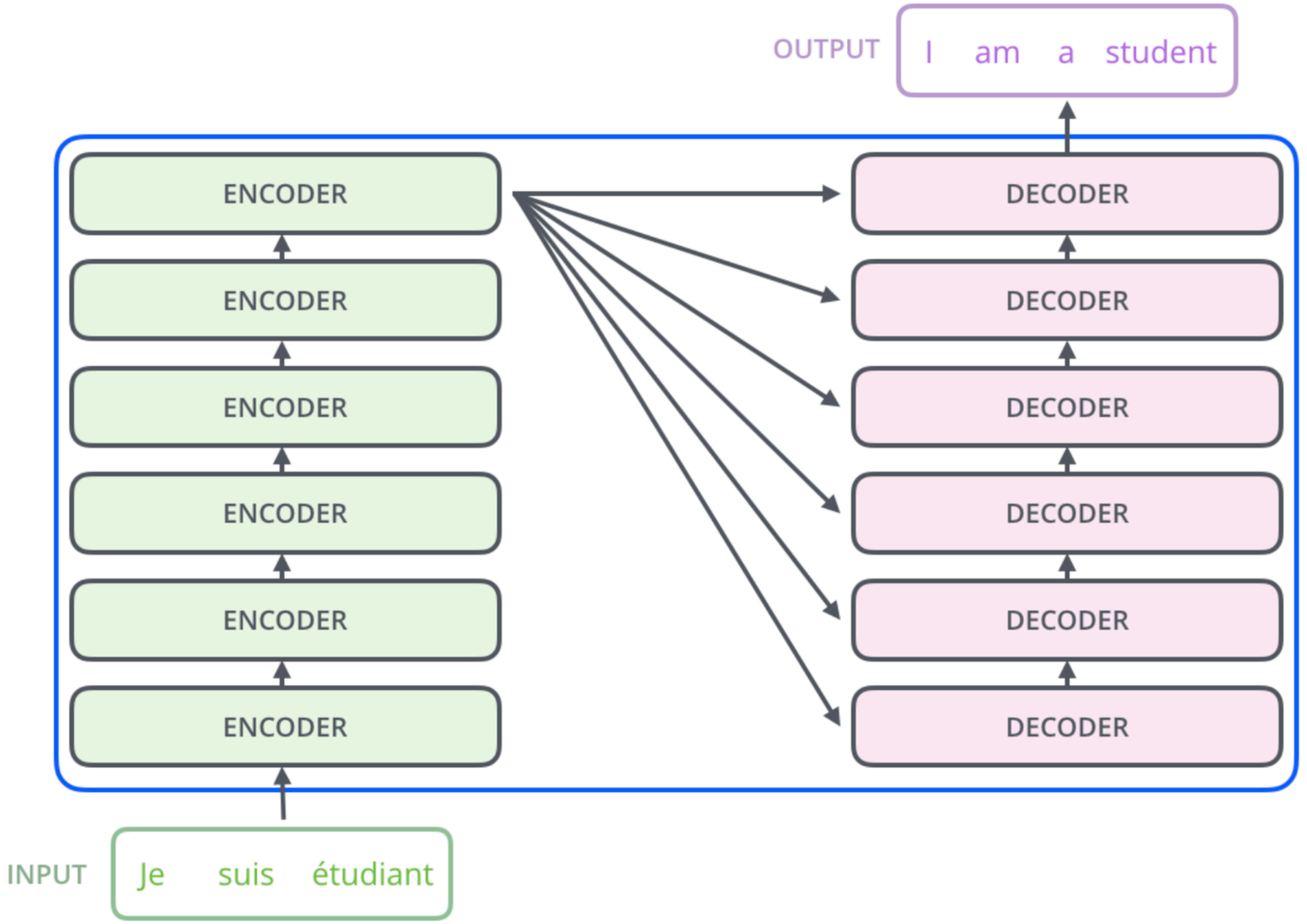
The Illustrated Transformer

Self-Attention



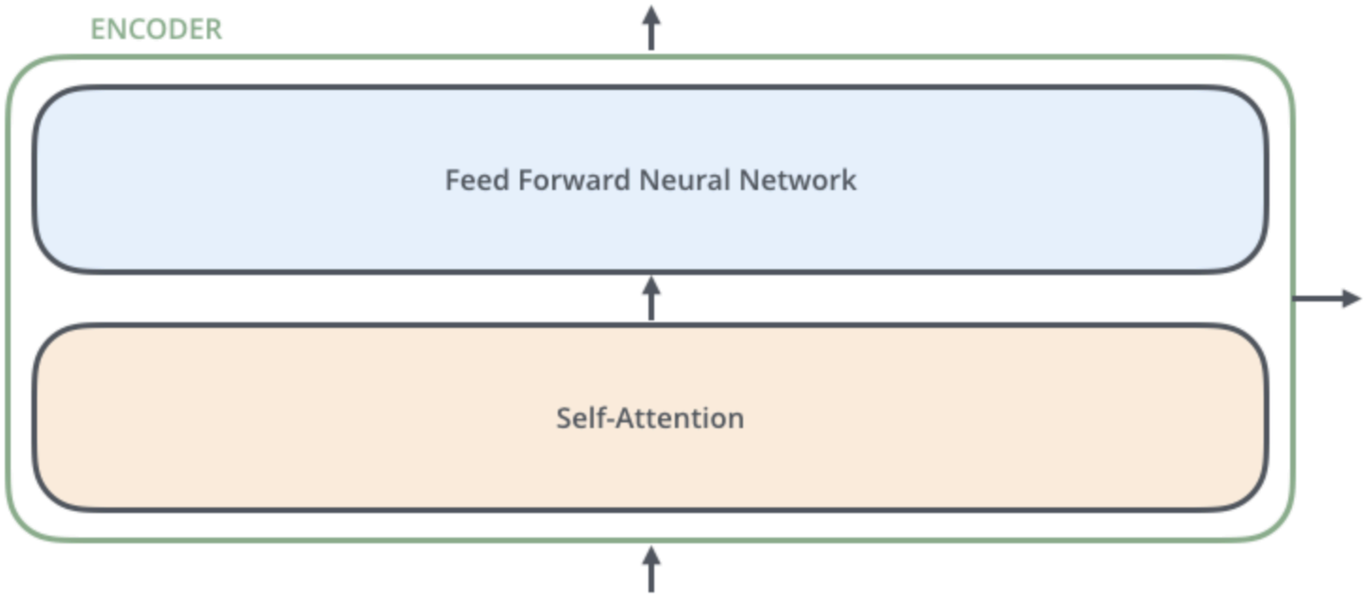
The Illustrated Transformer

Self-Attention



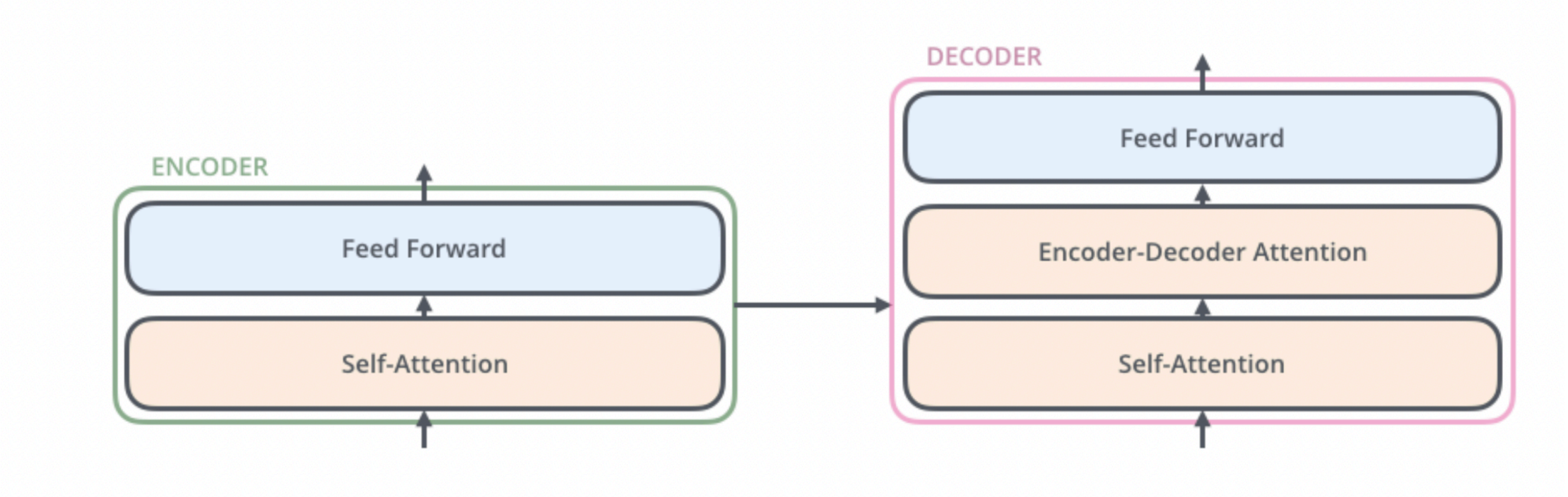
The Illustrated Transformer

Self-Attention



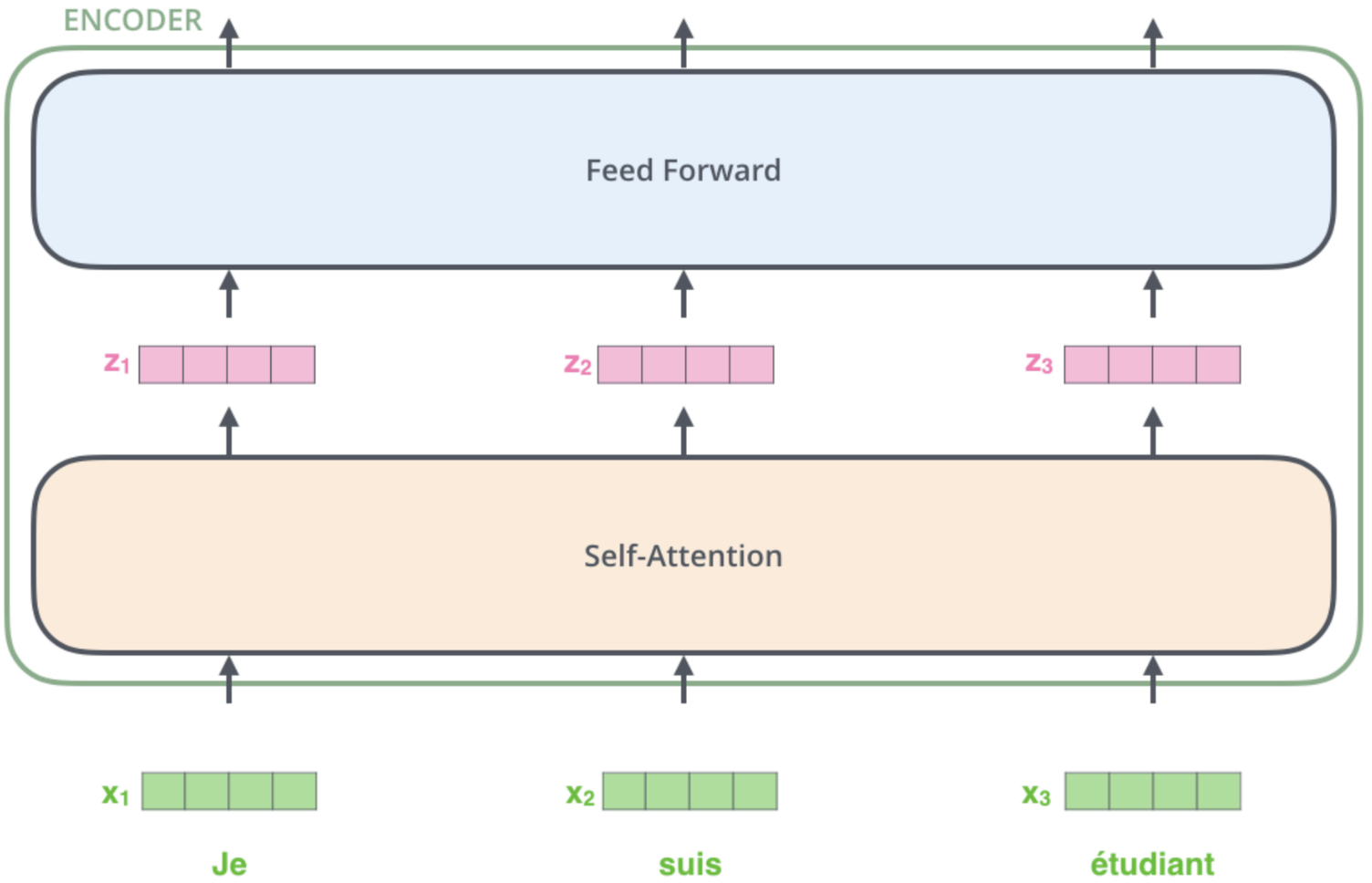
The Illustrated Transformer

Self-Attention



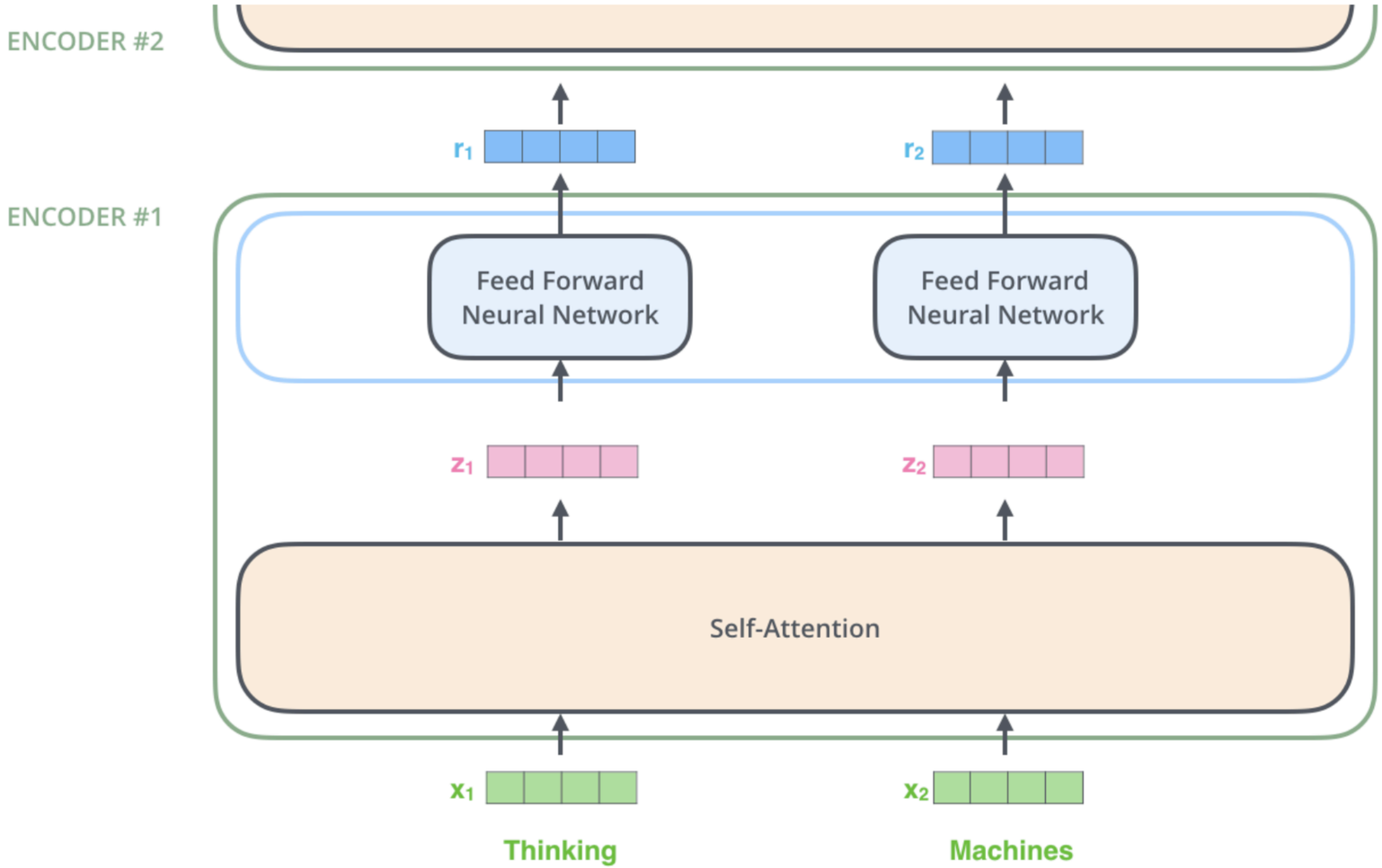
The Illustrated Transformer

Self-Attention



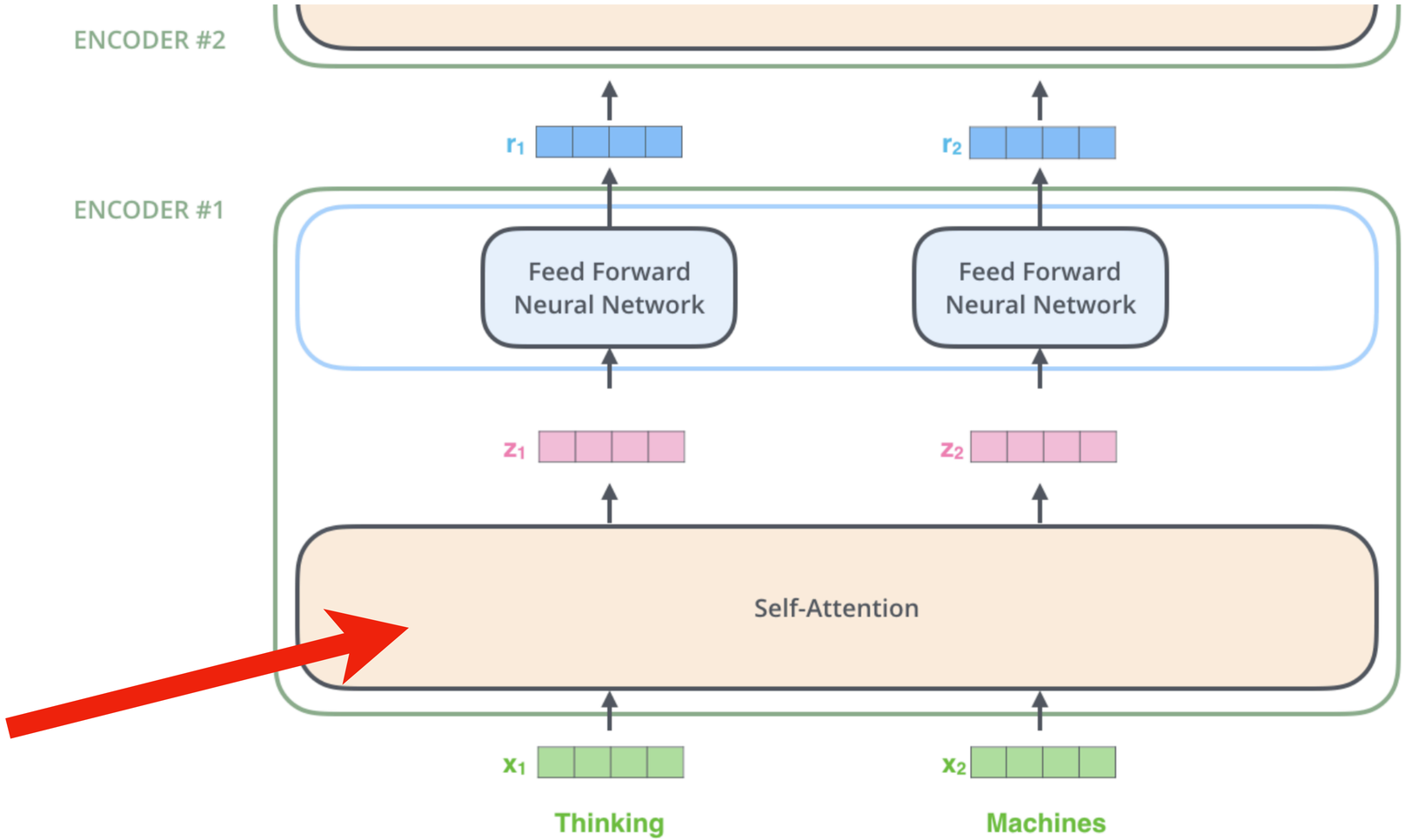
The Illustrated Transformer

Self-Attention



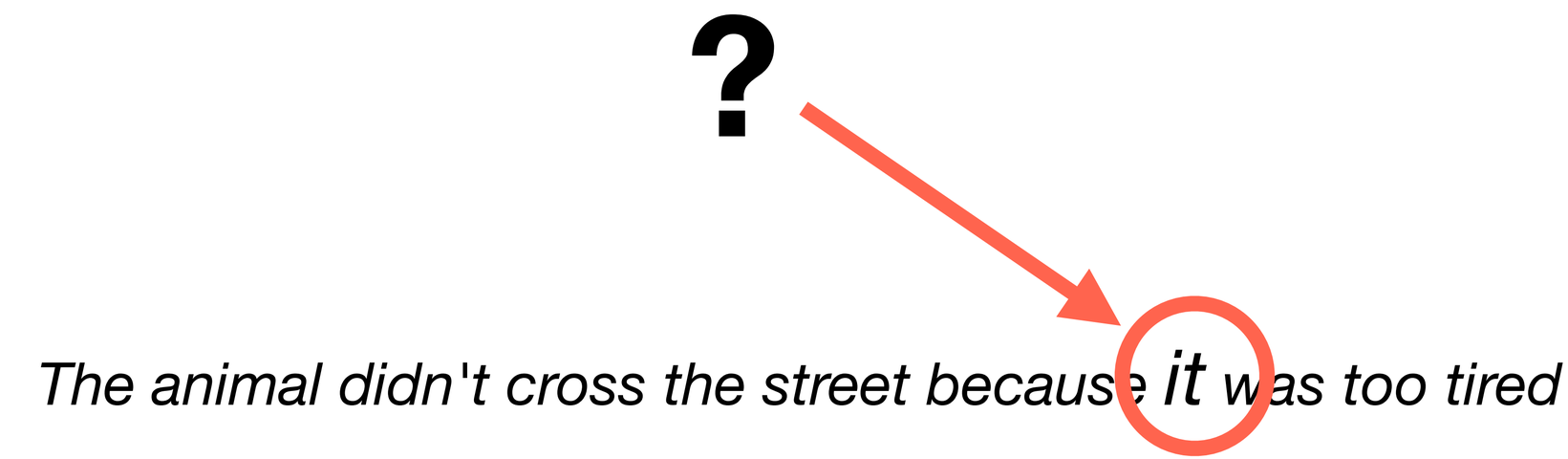
The Illustrated Transformer

Self-Attention

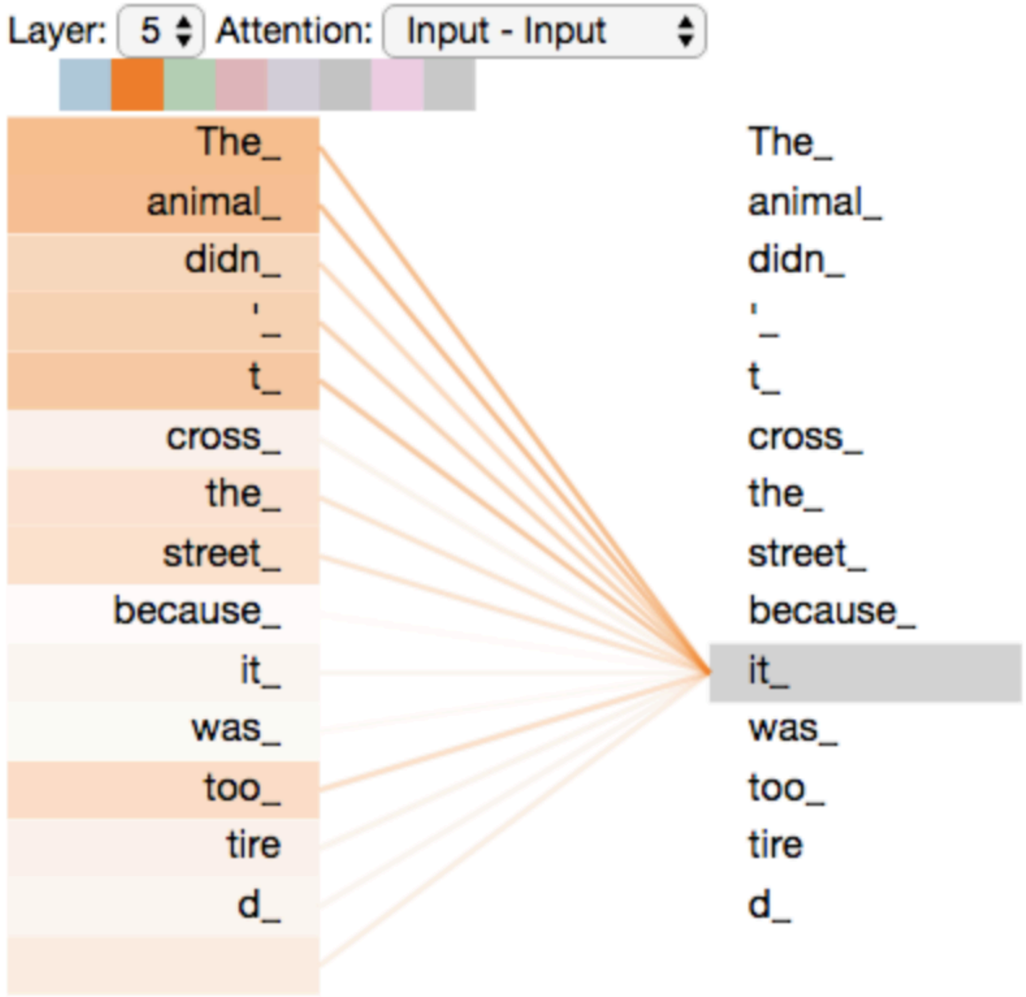


The Illustrated Transformer

The animal didn't cross the street because it was too tired

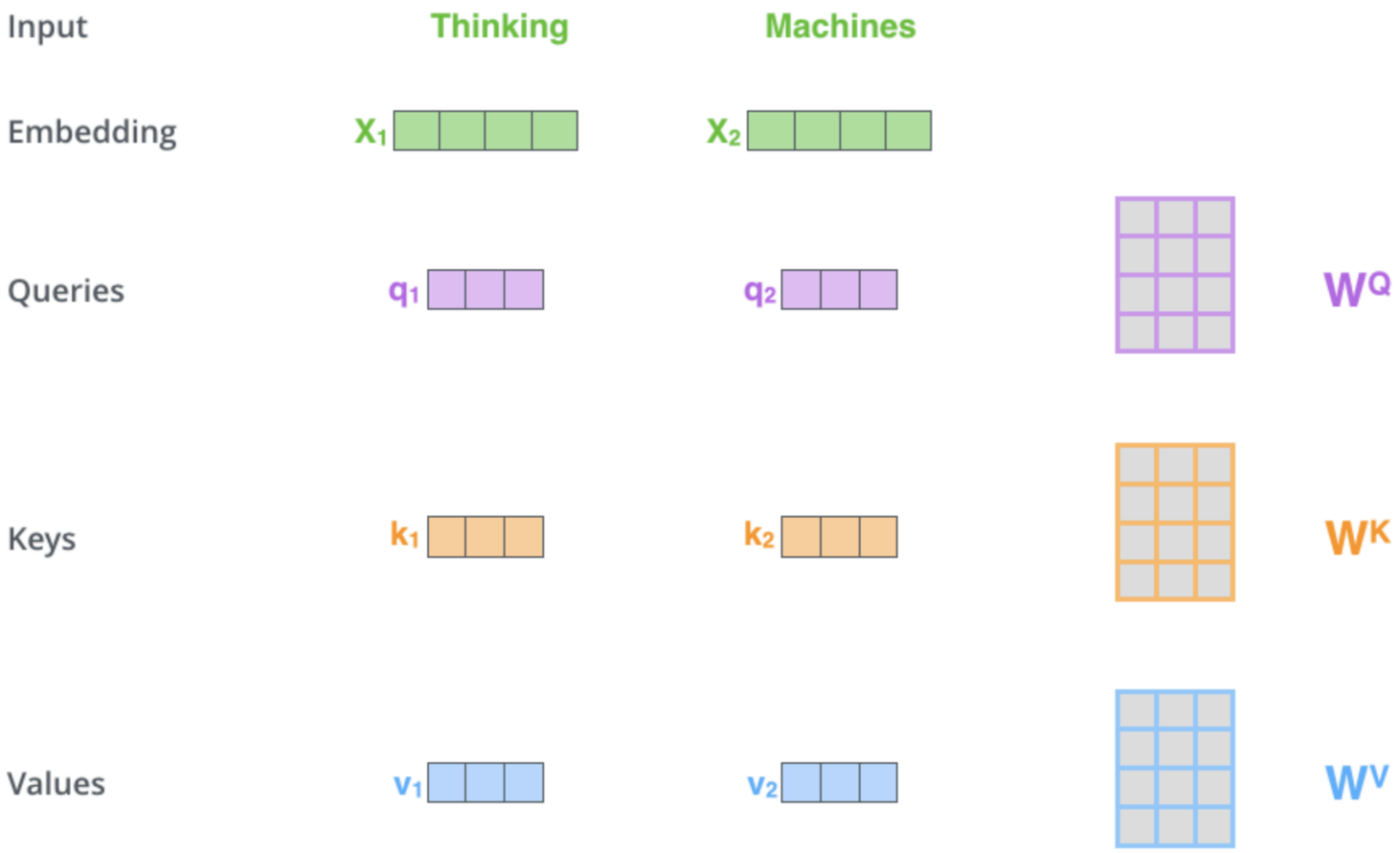

?
The animal didn't cross the street because it was too tired

Self-Attention



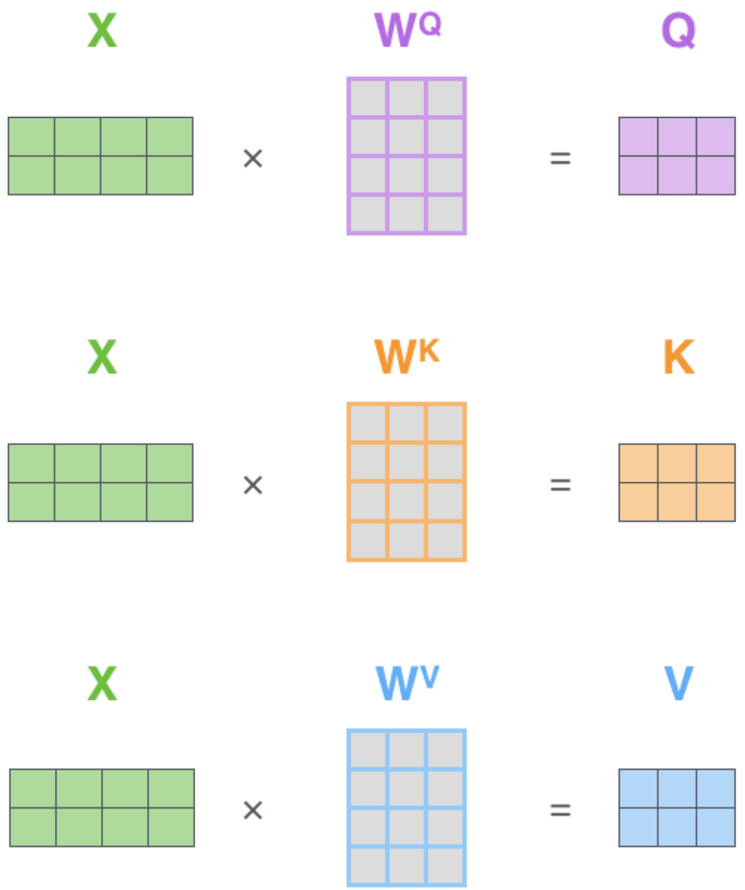
The Illustrated Transformer

Шаг 1 - Векторы Queries, Keys, Values



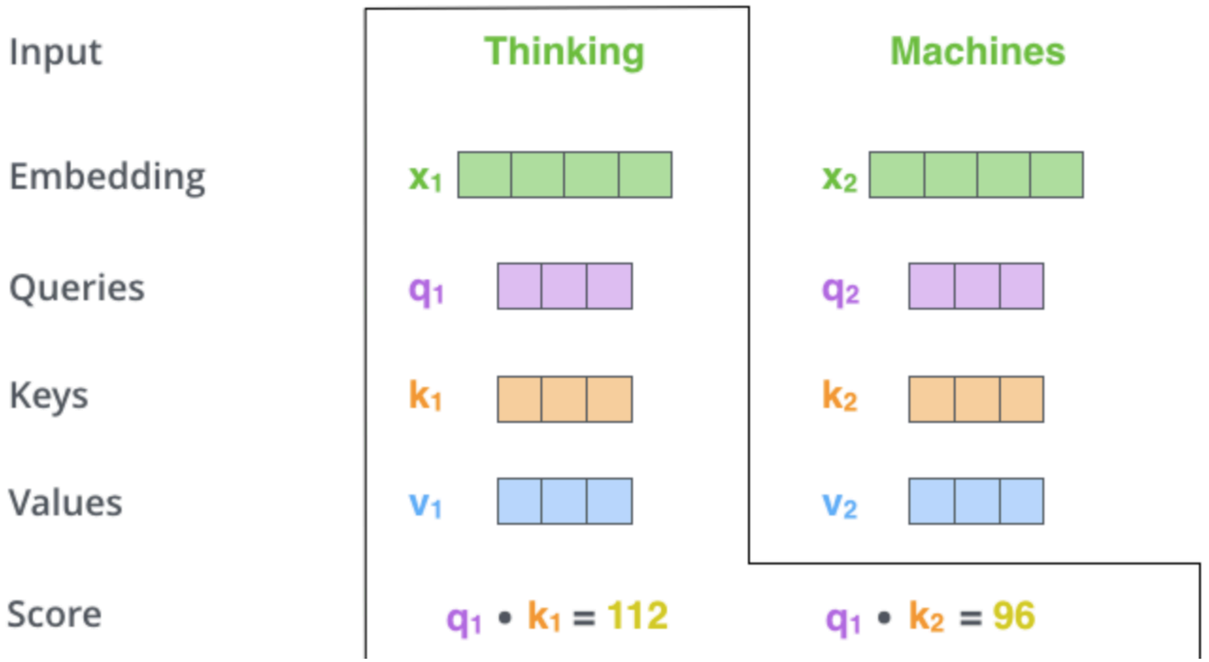
The Illustrated Transformer

Шаг 1 - Векторы Queries, Keys, Values



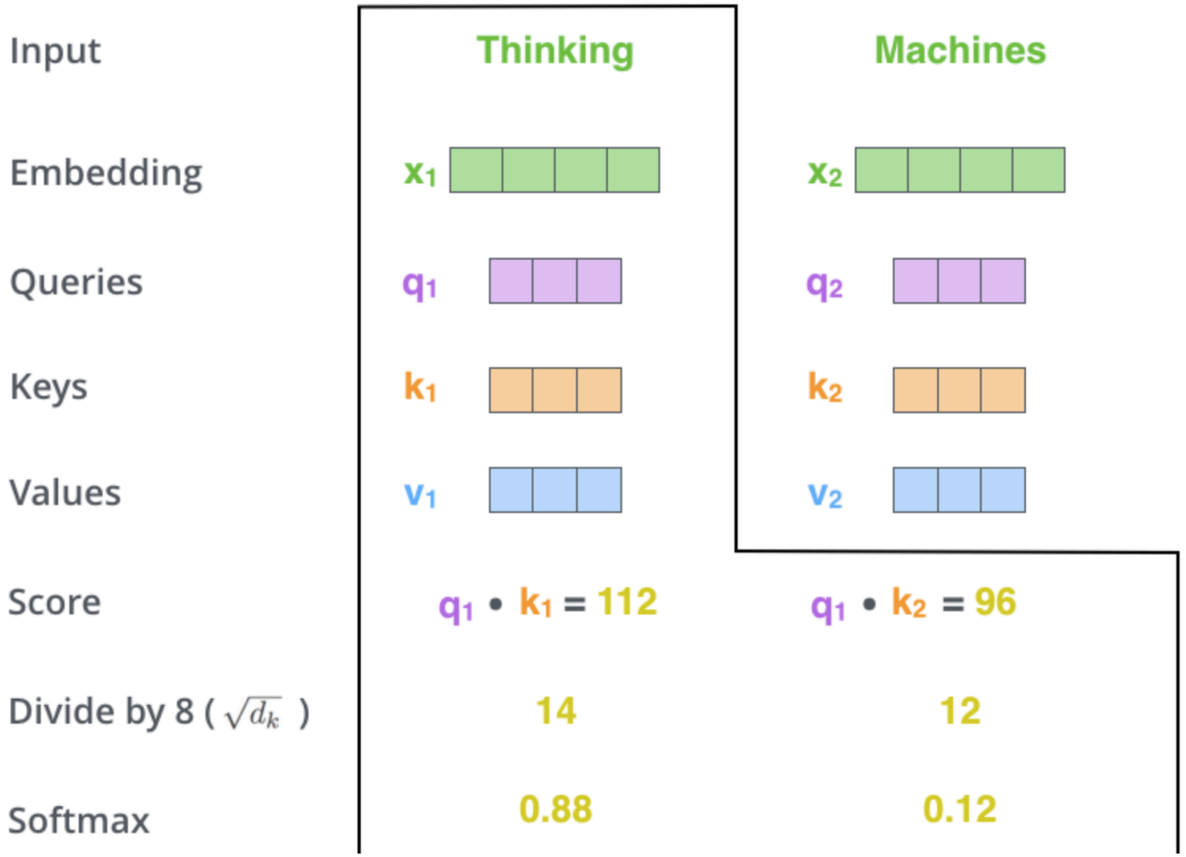
The Illustrated Transformer

Шаг 2 - Вычисление Score



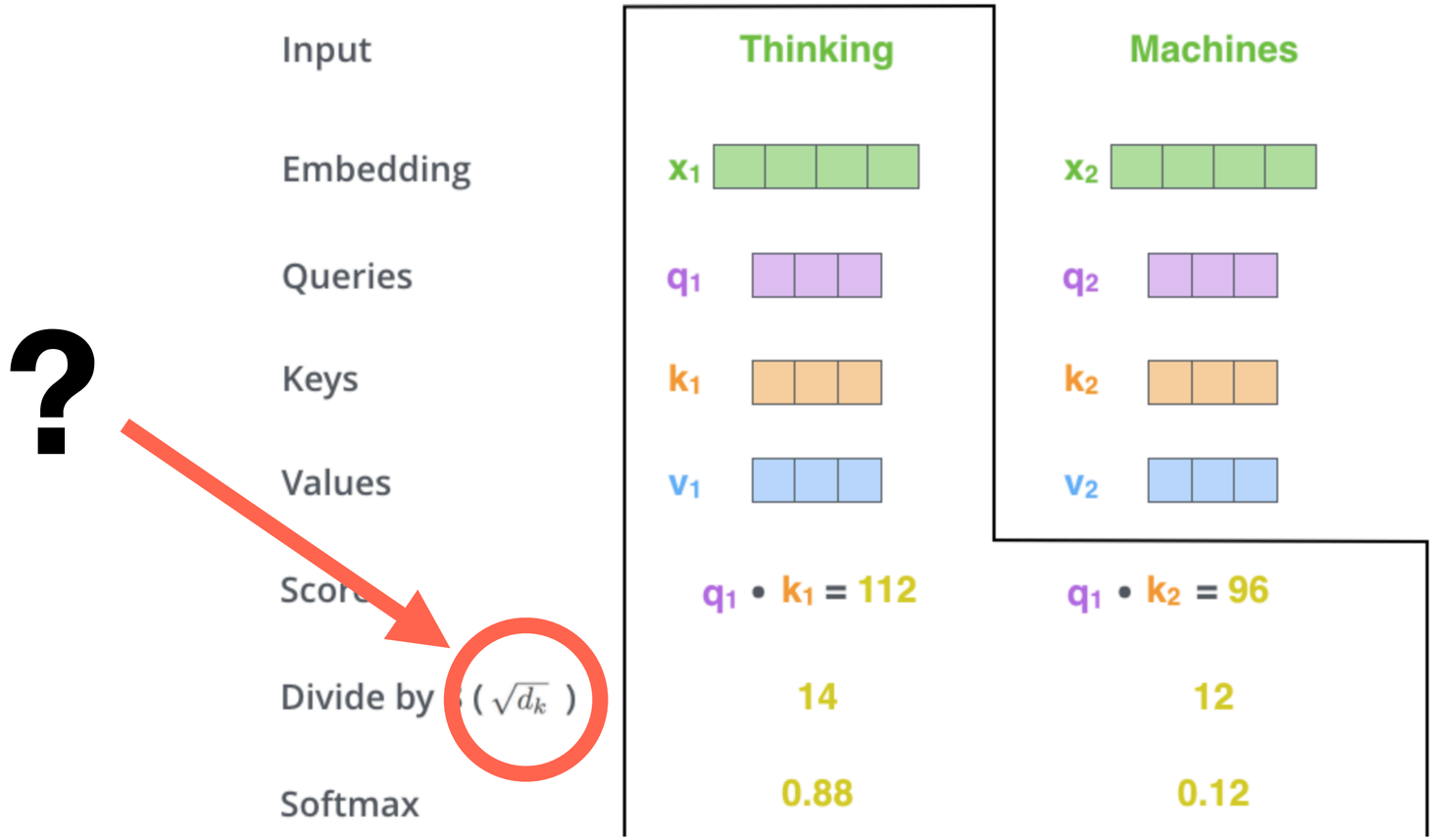
The Illustrated Transformer

Шаг 3: Делим Скор на sqrt(d)
Шаг 4: Softmax



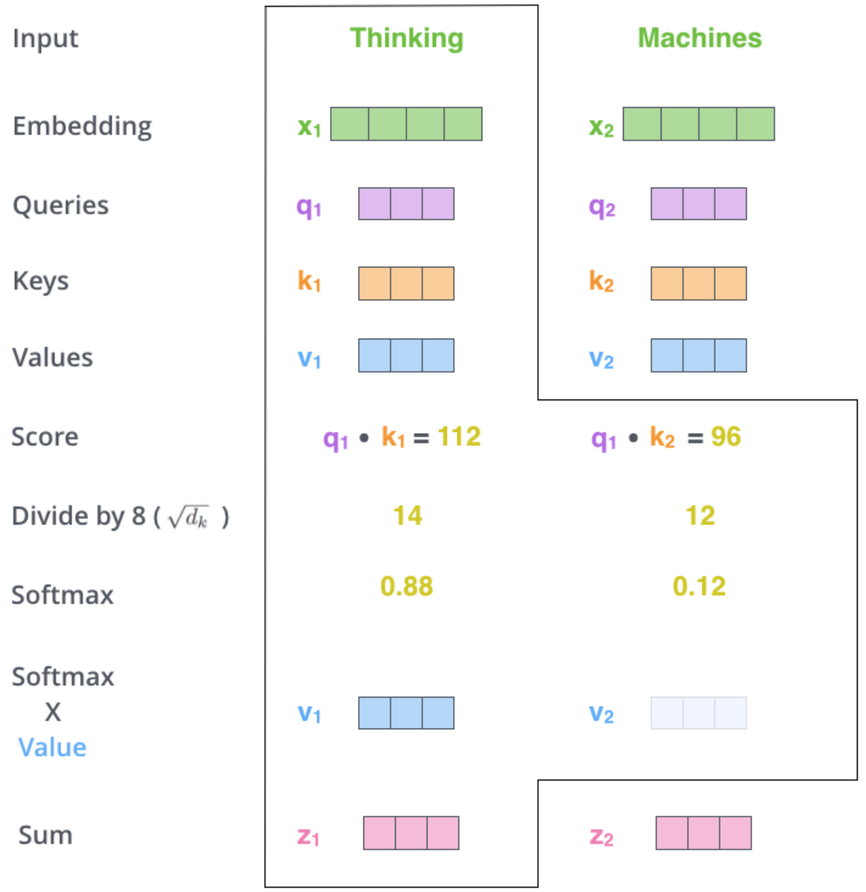
The Illustrated Transformer

Шаг 3: Делим Скор на sqrt(d)
Шаг 4: Softmax



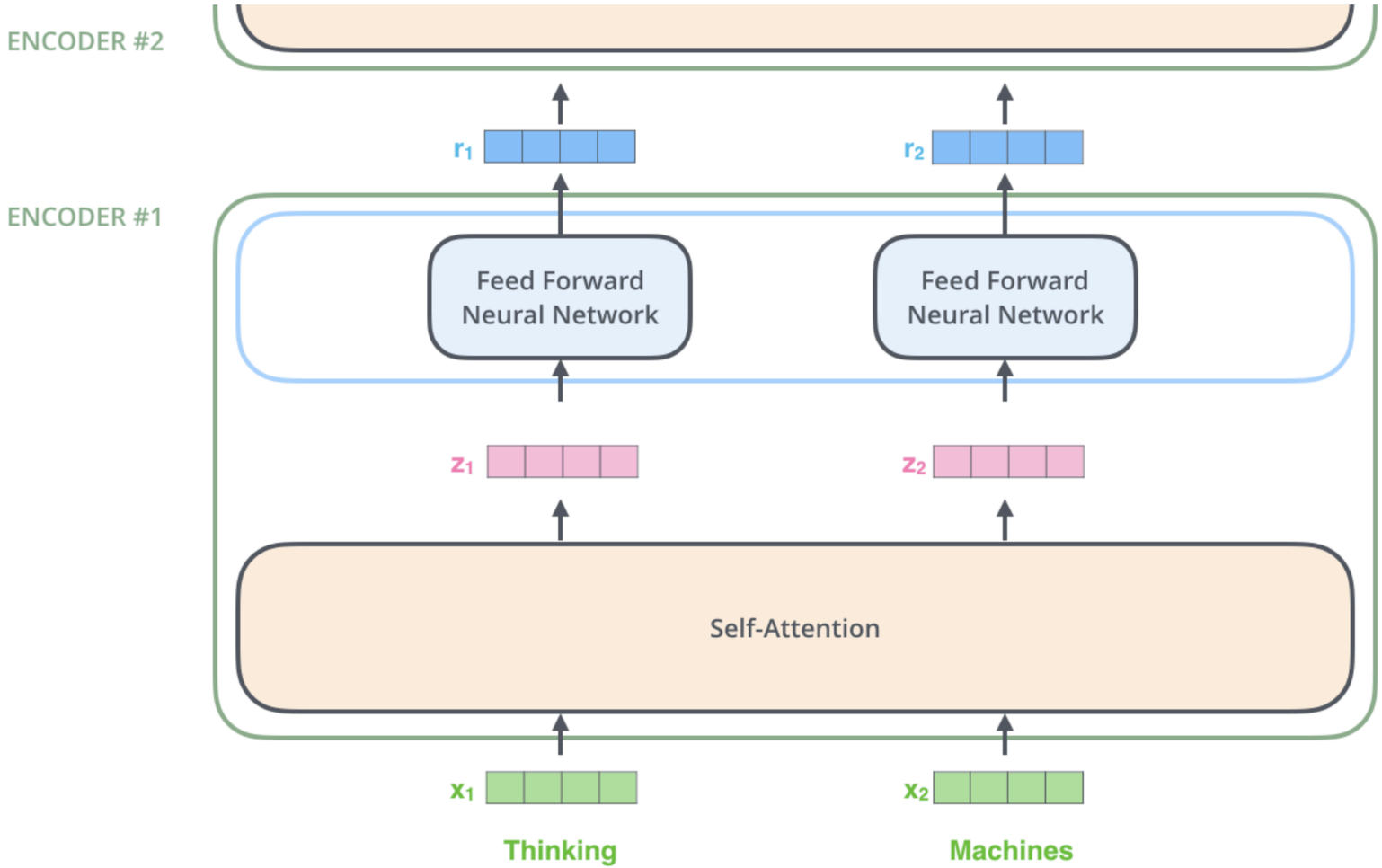
The Illustrated Transformer

Шаг 5: Умножение каждого Values вектора на значение softmax
Шаг 6: Суммирование векторов



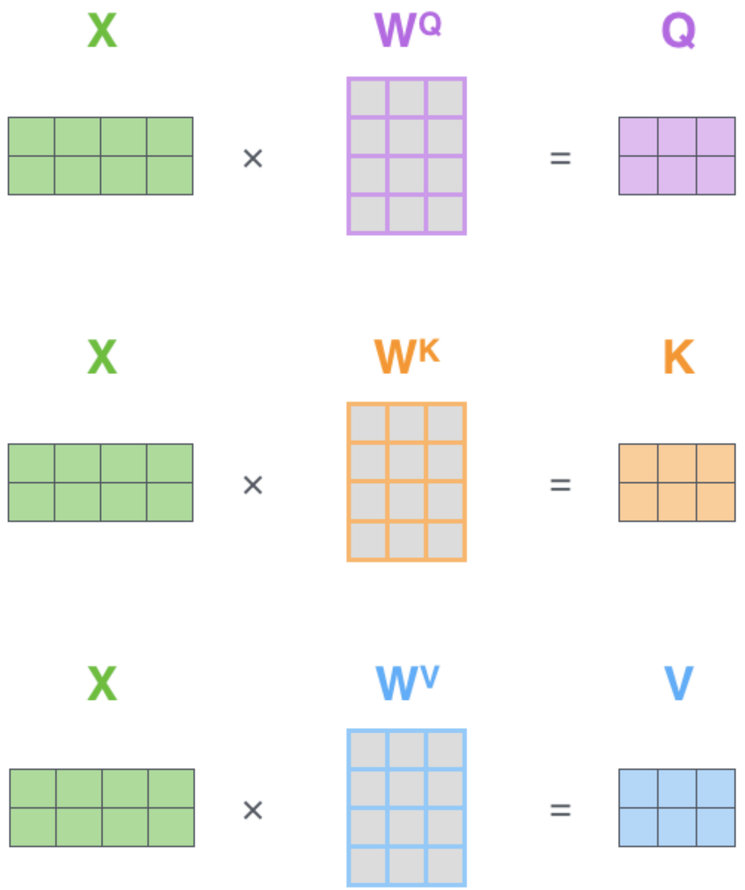
The Illustrated Transformer

Self-Attention



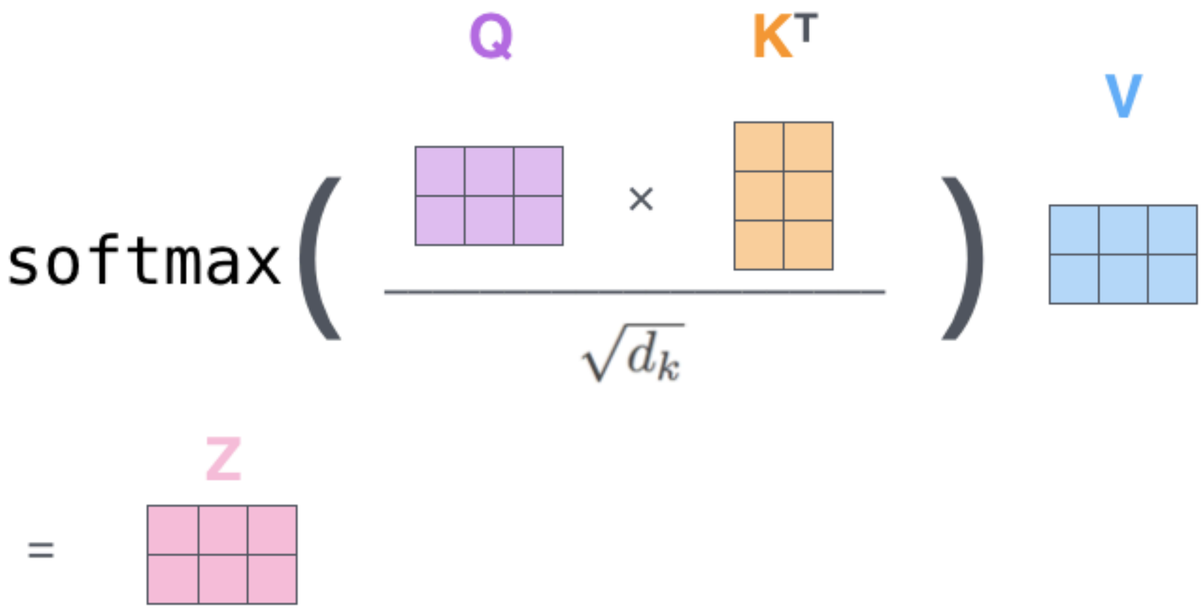
The Illustrated Transformer

Шаг 1 - Векторы Queries, Keys, Values



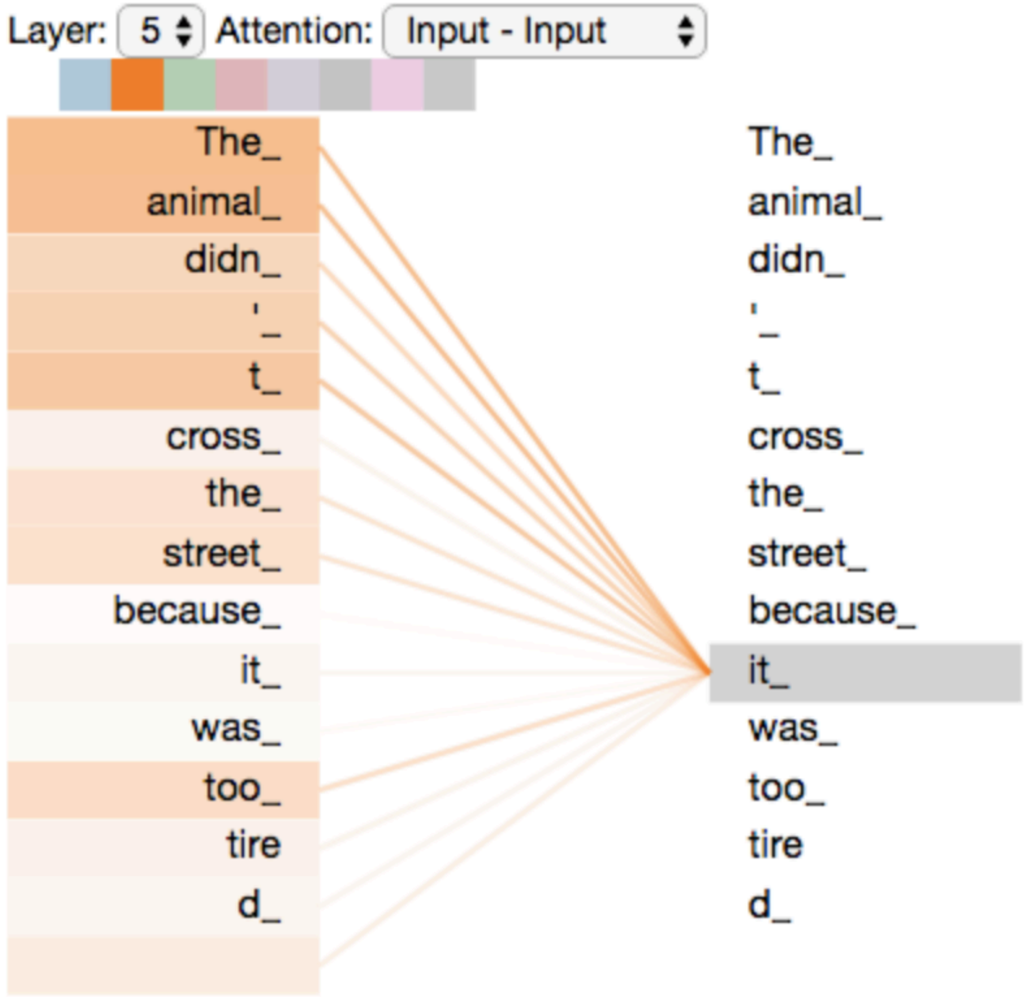
The Illustrated Transformer

Self-Attention

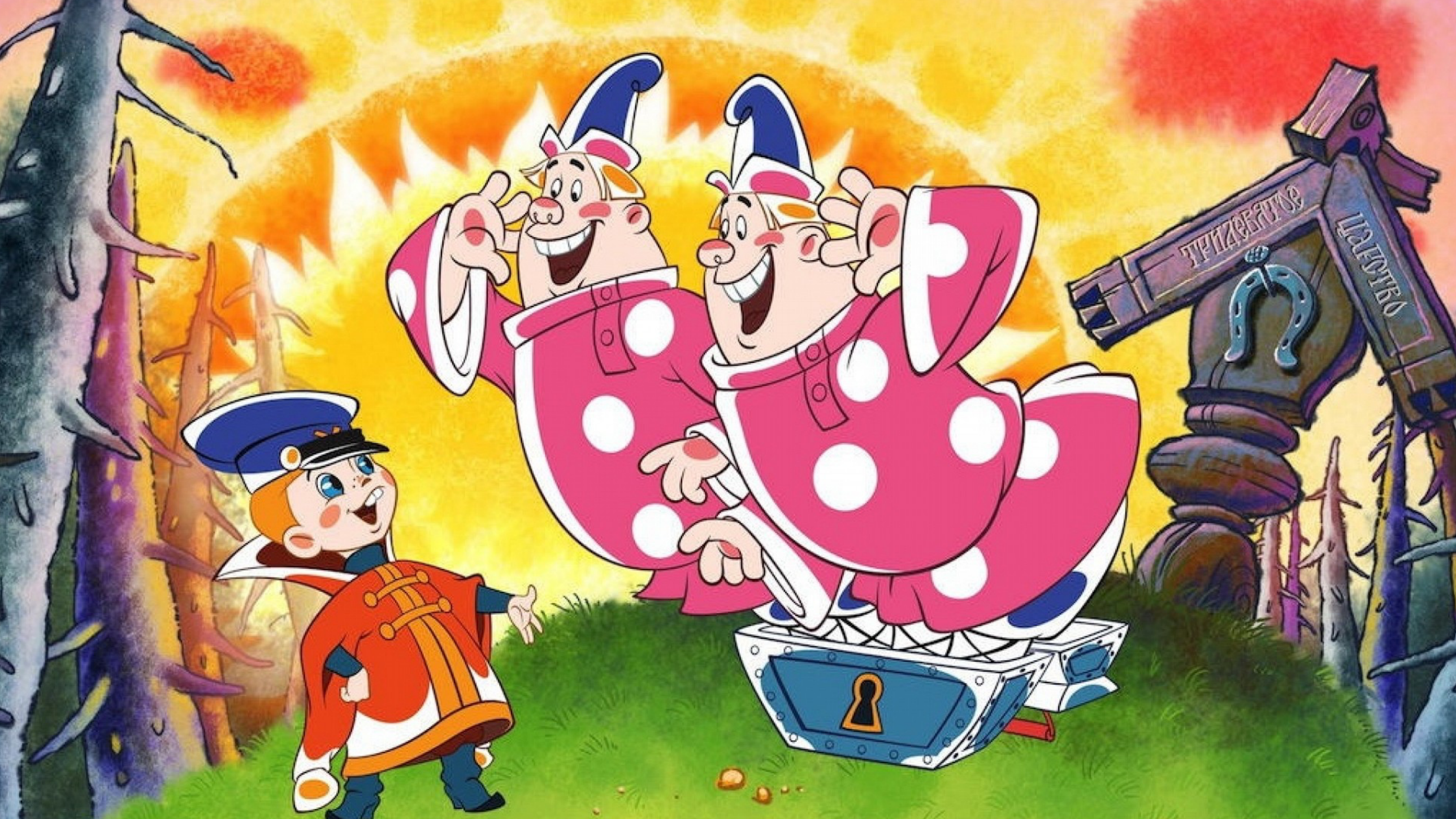


The Illustrated Transformer

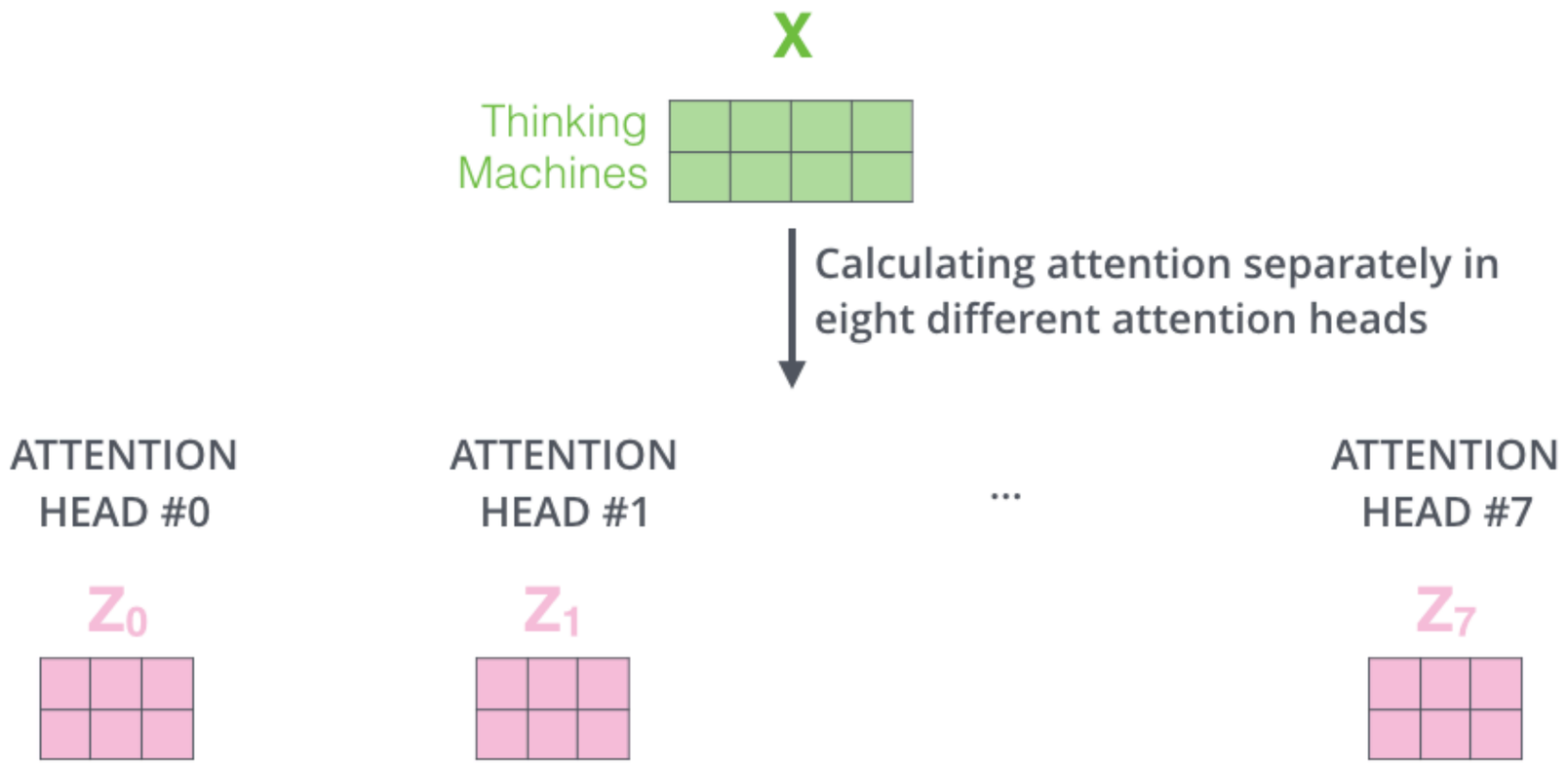
Self-Attention



The Illustrated Transformer

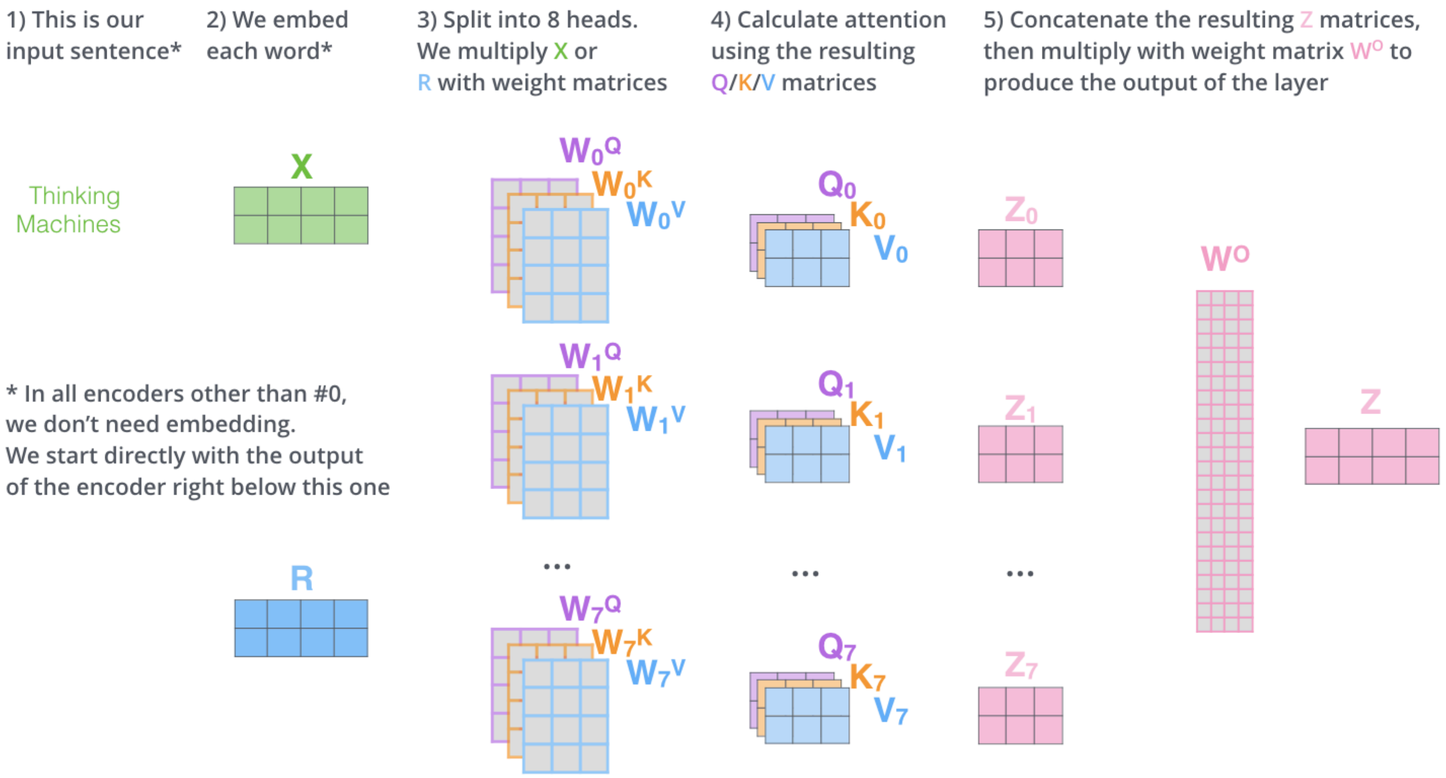

ТРИДЕВЯТОЕ
ЦАРСТВО

Multihead Self-Attention



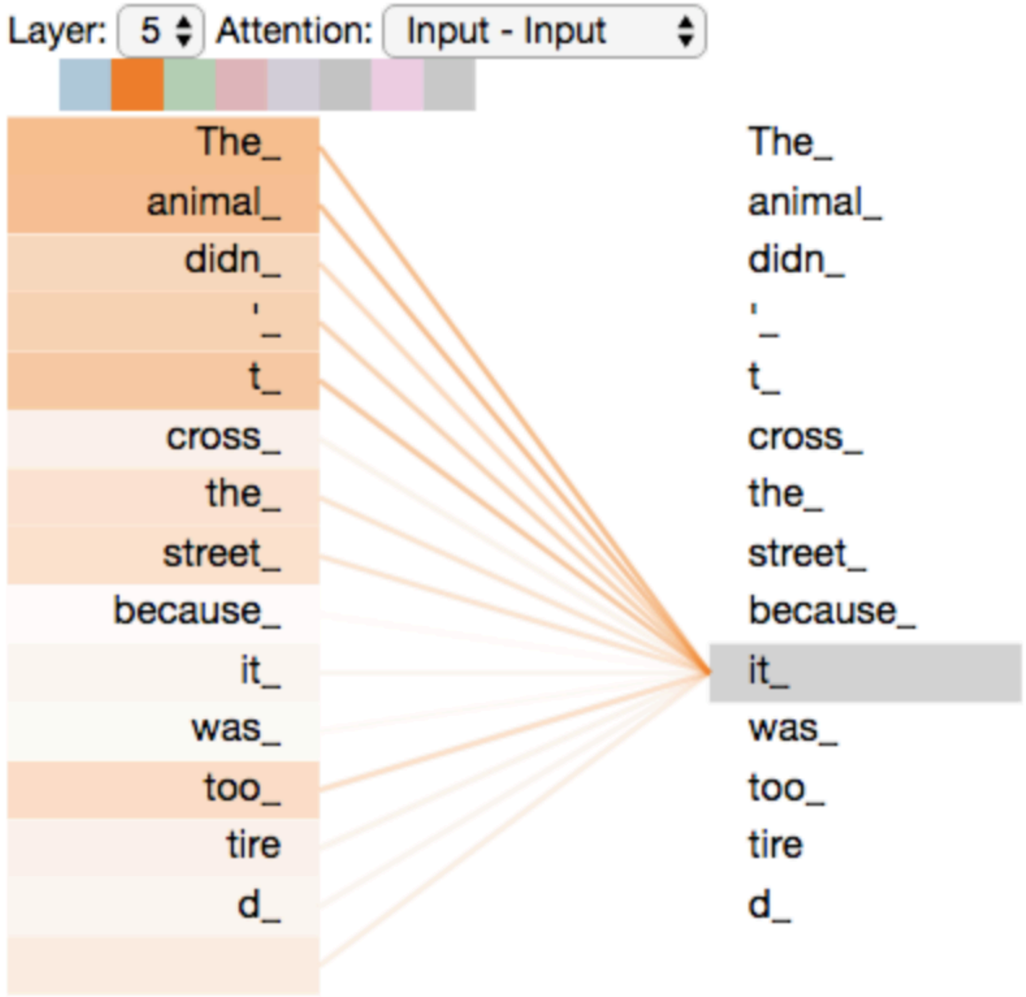
The Illustrated Transformer

Multihead Self-Attention



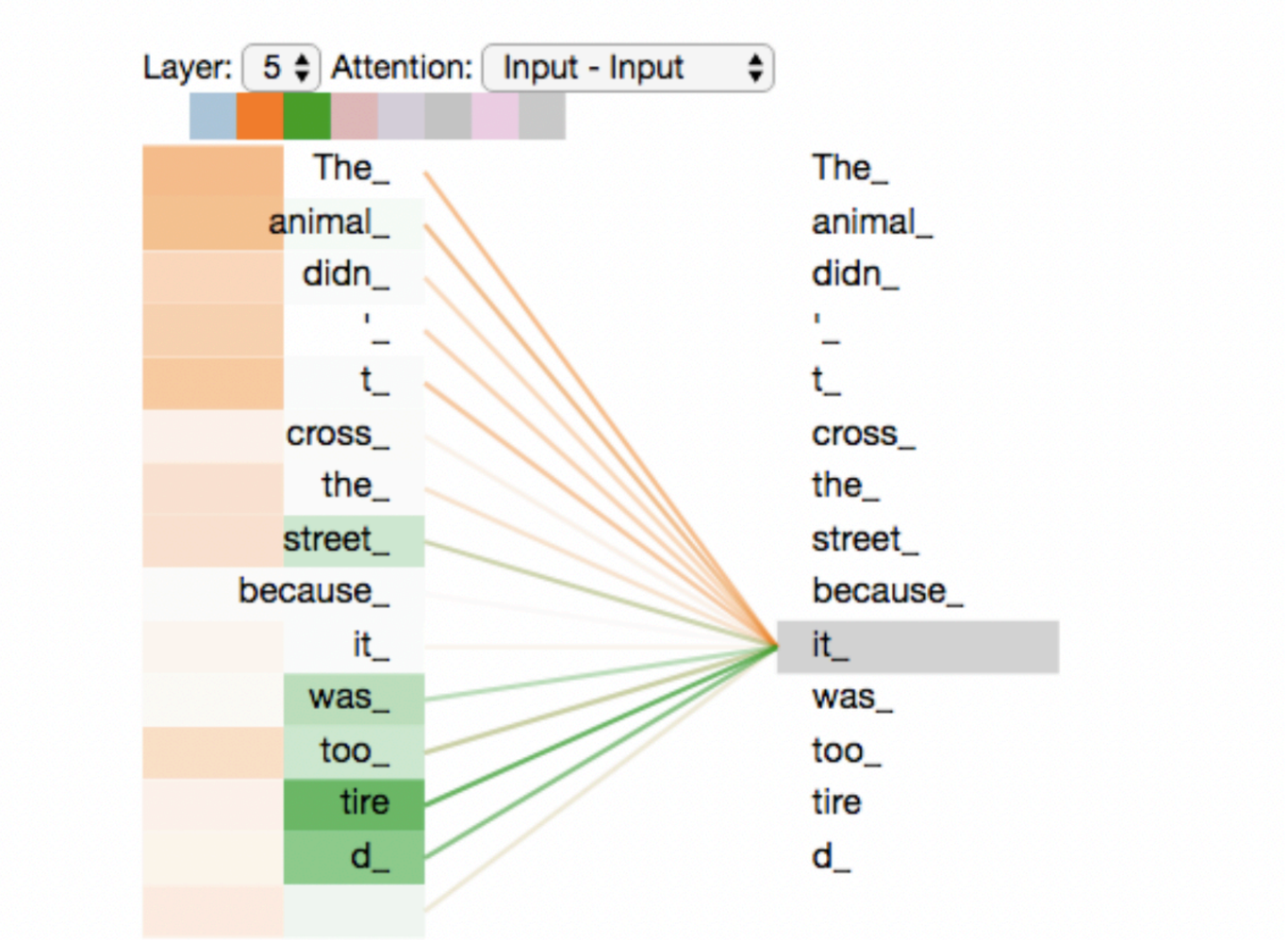
The Illustrated Transformer

Self-Attention



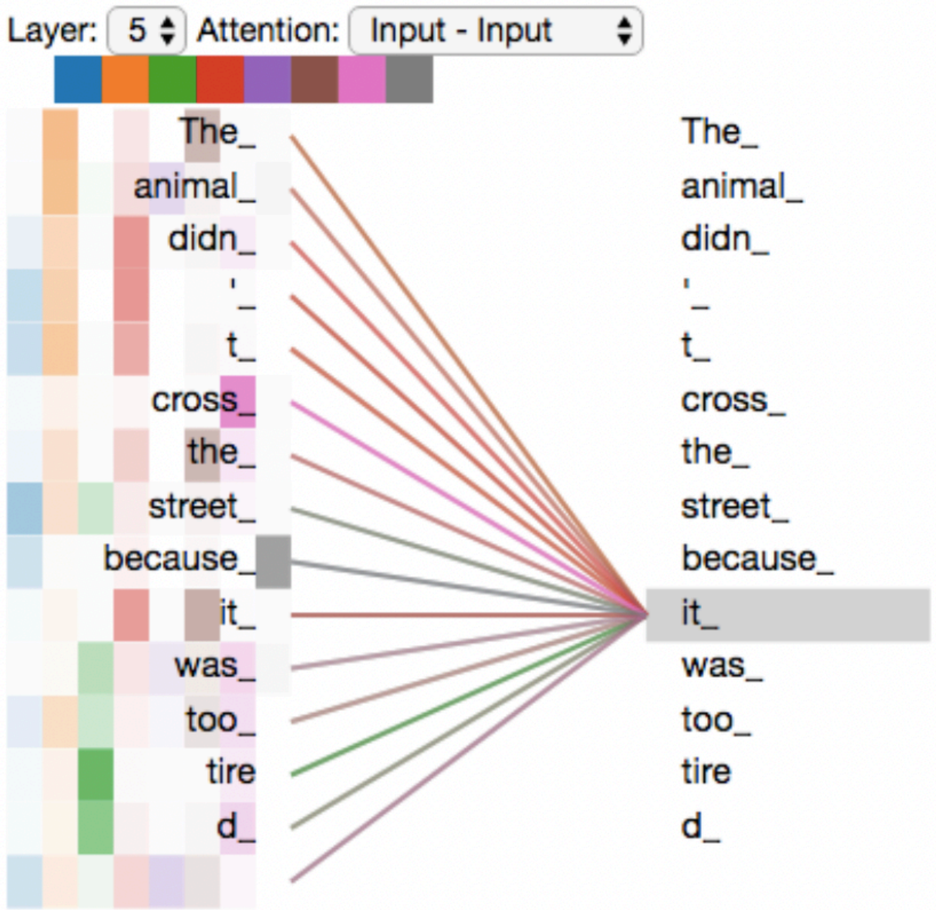
The Illustrated Transformer

Self-Attention



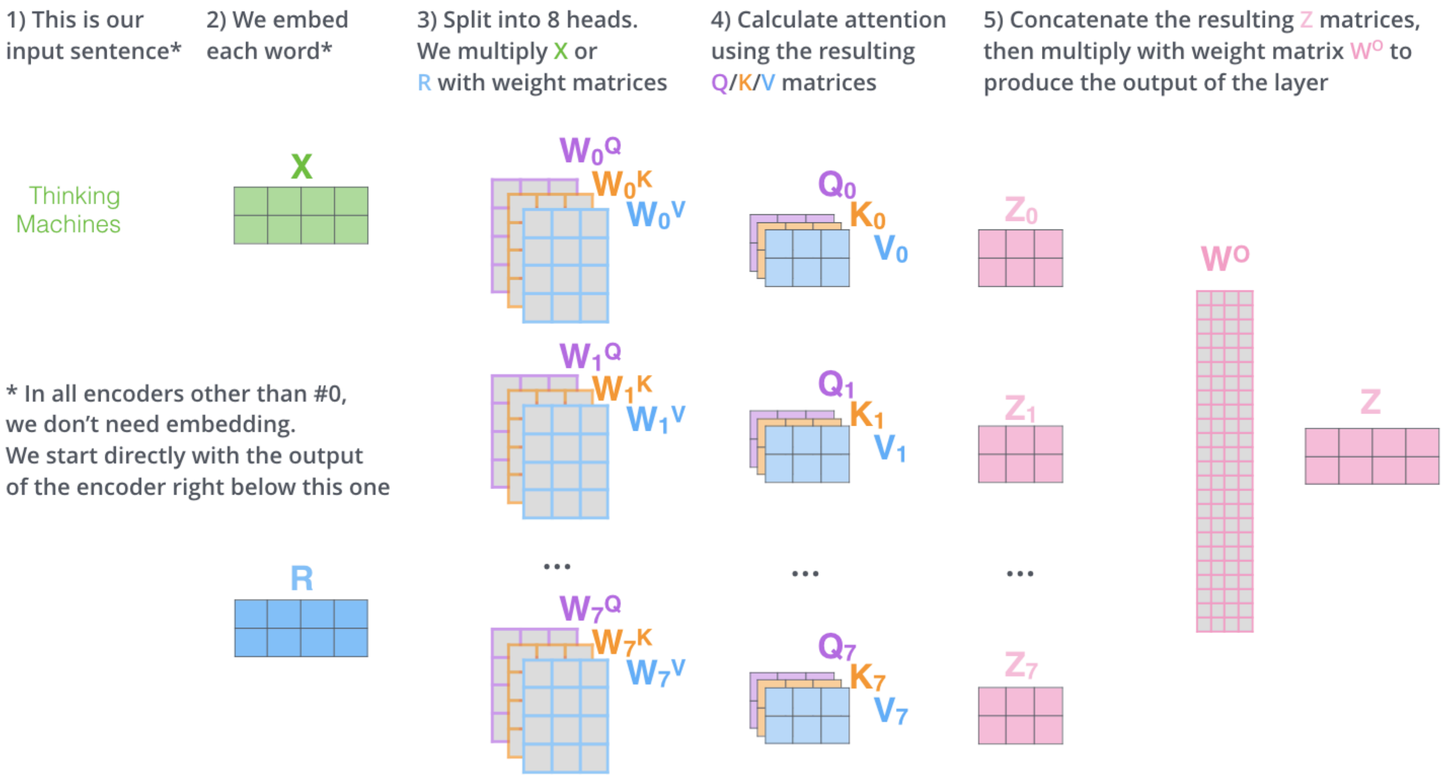
The Illustrated Transformer

Self-Attention



The Illustrated Transformer

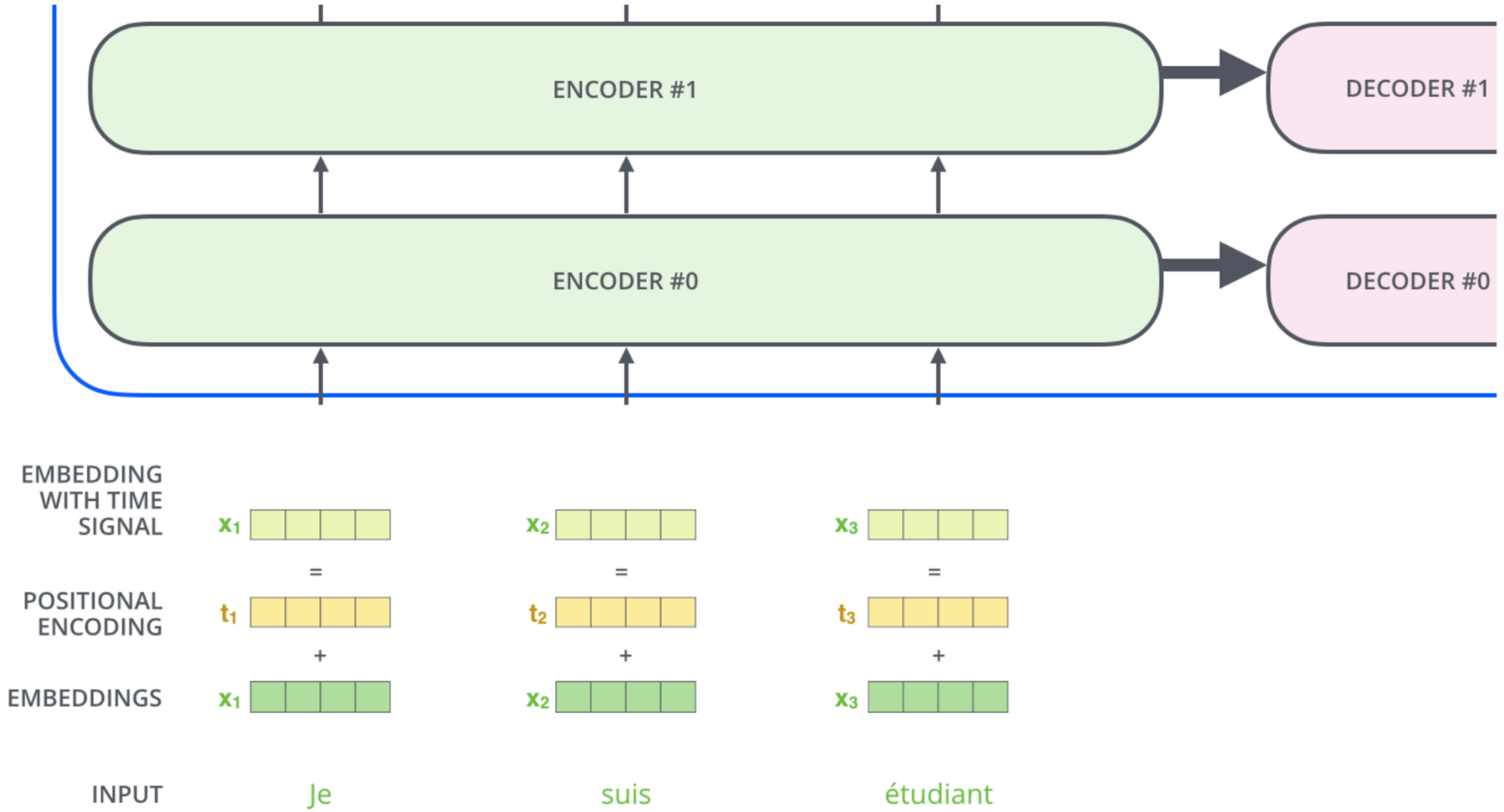
Multihead Self-Attention



The Illustrated Transformer

Но ведь нам нужно обрабатывать последовательности…

Positional Encoding



The Illustrated Transformer

Positional Encoding

$$\vec{p_t}^{(i)} = f(t)^{(i)} = \begin{cases} \sin(\omega_k t), & \text{if } i = 2k \\ \cos(\omega_k t), & \text{if } i = 2k+1 \end{cases}$$

$$\omega_k = \frac{1}{10000^{2k/d}}$$

https://github.com/girafe-ai/ml-mipt/blob/advanced_f20/week1_04_Transformer_and_PoS_Tagging/ml-mipt_f20_lect104_Self_Attention_and_Transformer.pdf

Positional Encoding

$$\vec{p_t}^{(i)} = f(t)^{(i)} = \begin{cases} \sin(\omega_k t), & \text{if } i = 2k \\ \cos(\omega_k t), & \text{if } i = 2k+1 \end{cases}$$

$$\omega_k = \frac{1}{10000^{2k/d}}$$

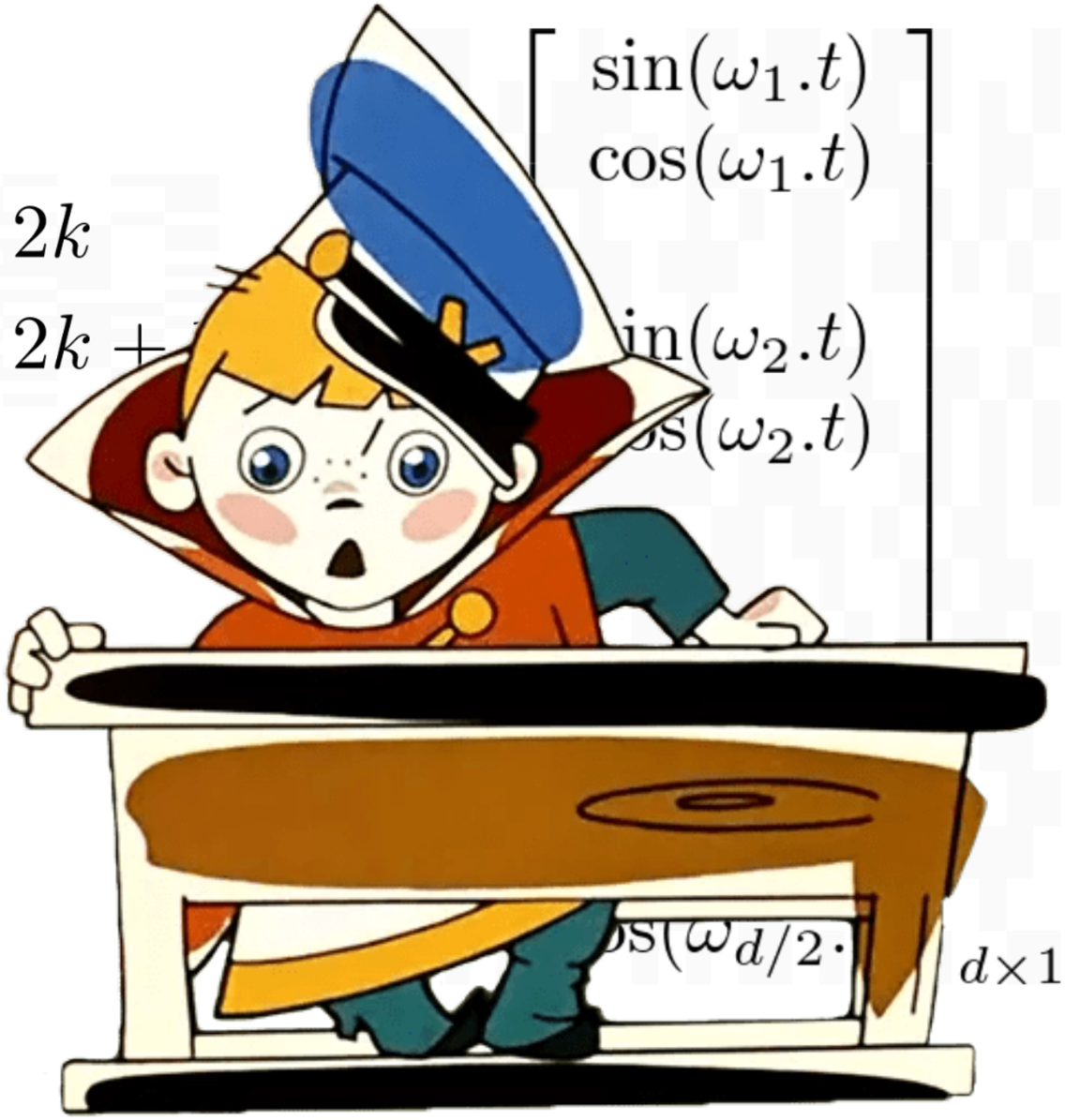
$$\vec{p_t} = \begin{bmatrix} \sin(\omega_1.t) \\ \cos(\omega_1.t) \\ \\ \sin(\omega_2.t) \\ \cos(\omega_2.t) \\ \\ \vdots \\ \\ \sin(\omega_{d/2}.t) \\ \cos(\omega_{d/2}.t) \end{bmatrix}_{d \times 1}$$

https://github.com/girafe-ai/ml-mipt/blob/advanced_f20/week1_04_Transformer_and_PoS_Tagging/ml-mipt_f20_lect104_Self_Attention_and_Transformer.pdf

Positional Encoding

$$\vec{p_t}^{(i)} = f(t)^{(i)} = \begin{cases} \sin(\omega_k t), & \text{if } i = 2k \\ \cos(\omega_k t), & \text{if } i = 2k + \end{cases}$$

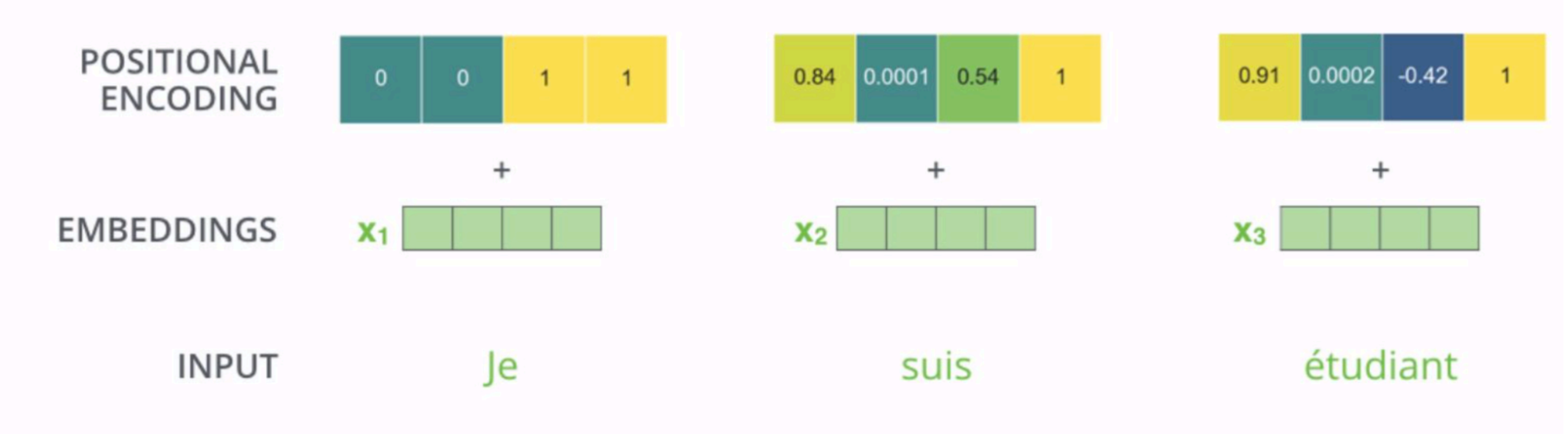
$$\omega_k = \frac{1}{10000^{2k/d}}$$



https://github.com/girafe-ai/ml-mipt/blob/advanced_f20/week1_04_Transformer_and_PoS_Tagging/ml-mipt_f20_lect104_Self_Attention_and_Transformer.pdf


Token Position
Embedding Dimension
0
1
2
3
4
5
6
7
8
9
0
10
20
30
40
1.00
0.75
0.50
0.25
−0.25

Positional Encoding



Positional Encoding

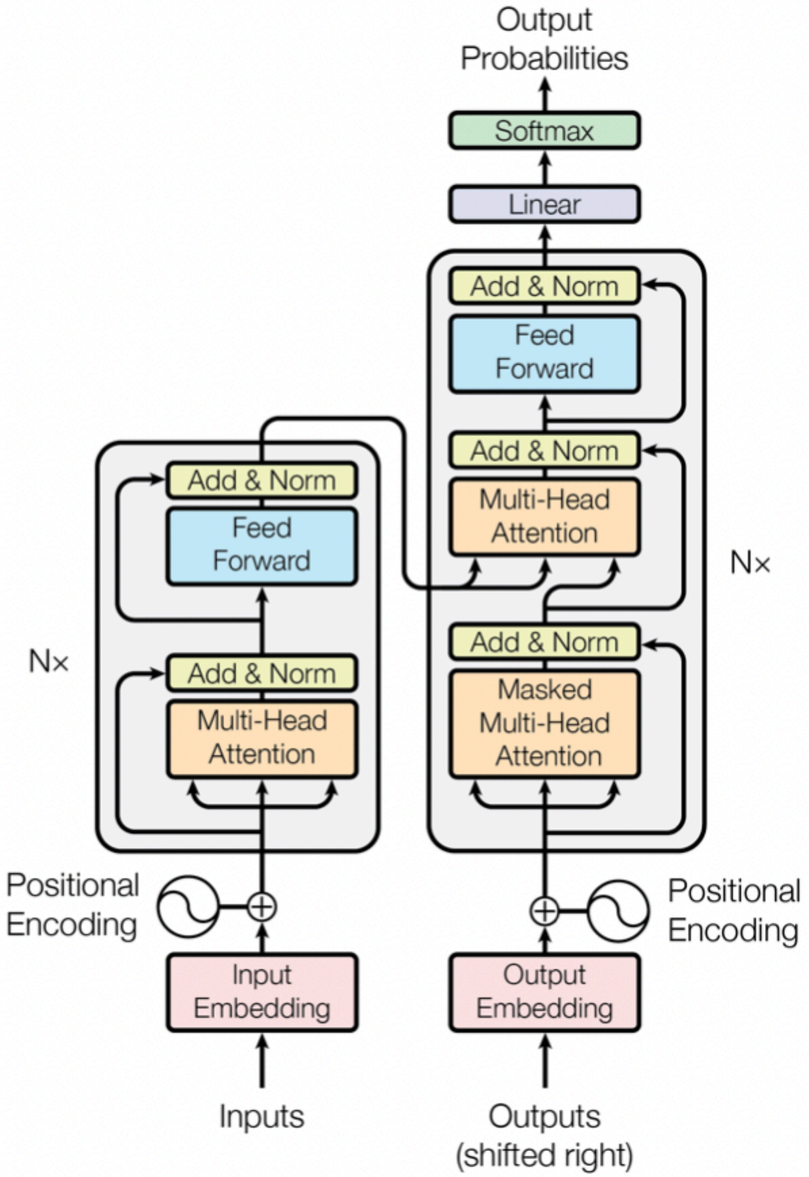
We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PEpos+k can be represented as a linear function of PEpos.

$$M \begin{bmatrix} \sin(\omega_k t) \\ \cos(\omega_k t) \end{bmatrix} = \begin{bmatrix} \sin(\omega_k (t + \phi)) \\ \cos(\omega_k (t + \phi)) \end{bmatrix}$$

https://github.com/girafe-ai/ml-mipt/blob/advanced_f20/week1_04_Transformer_and_PoS_Tagging/ml-mipt_f20_lect104_Self_Attention_and_Transformer.pdf

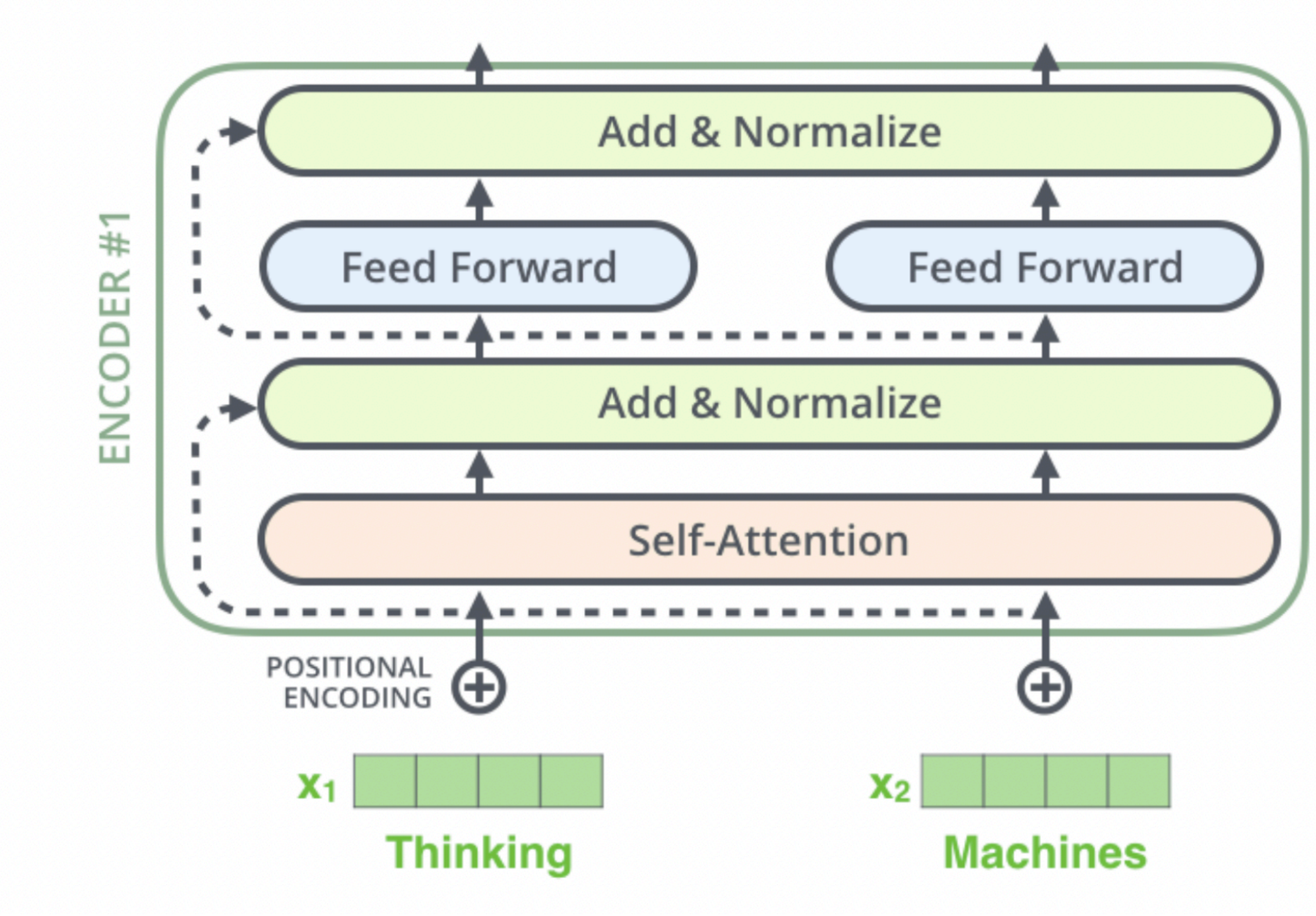
The Residuals & layer-normalization

The Residuals & layer-normalization



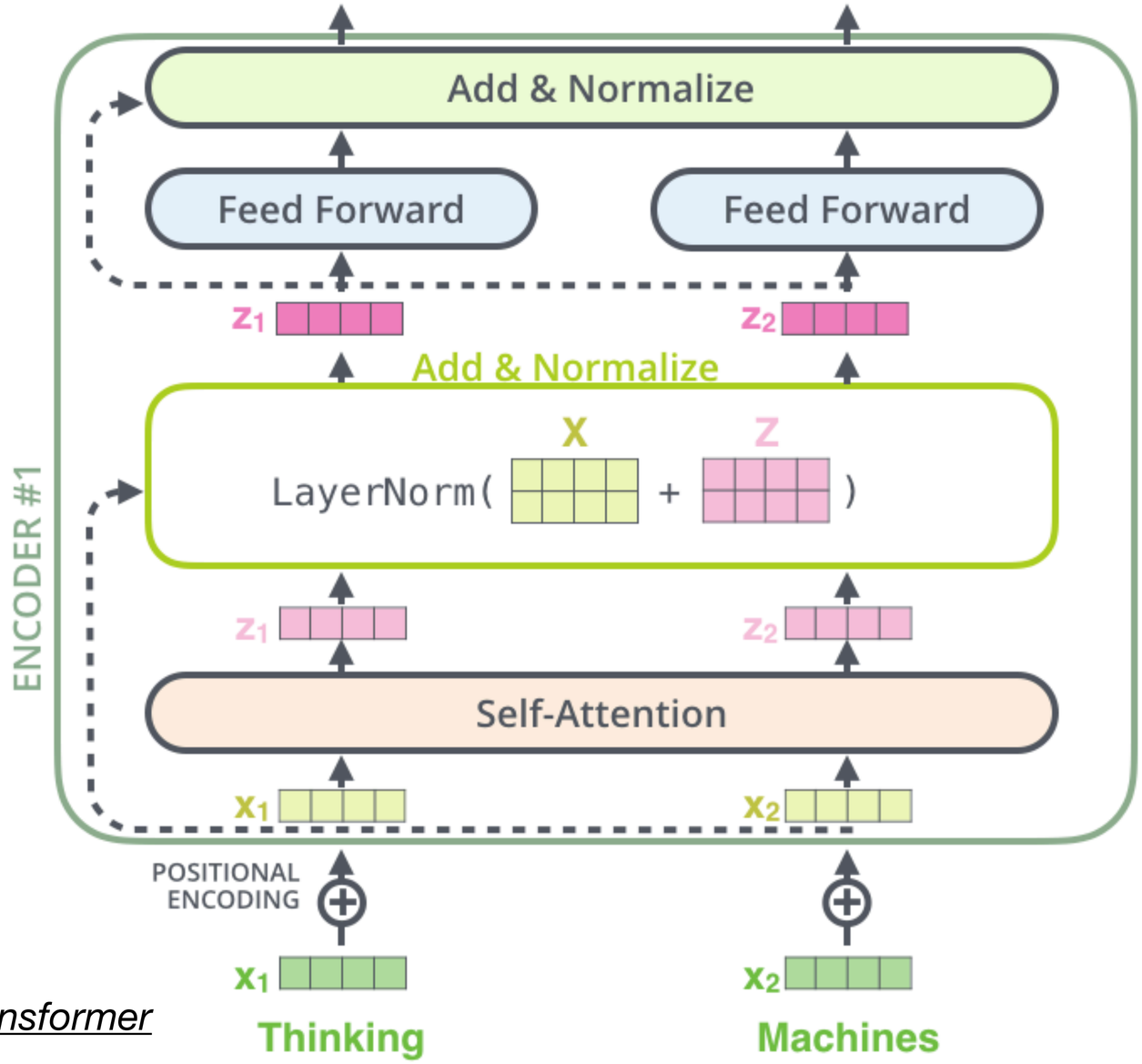
Attention Is All You Need

The Residuals & layer-normalization



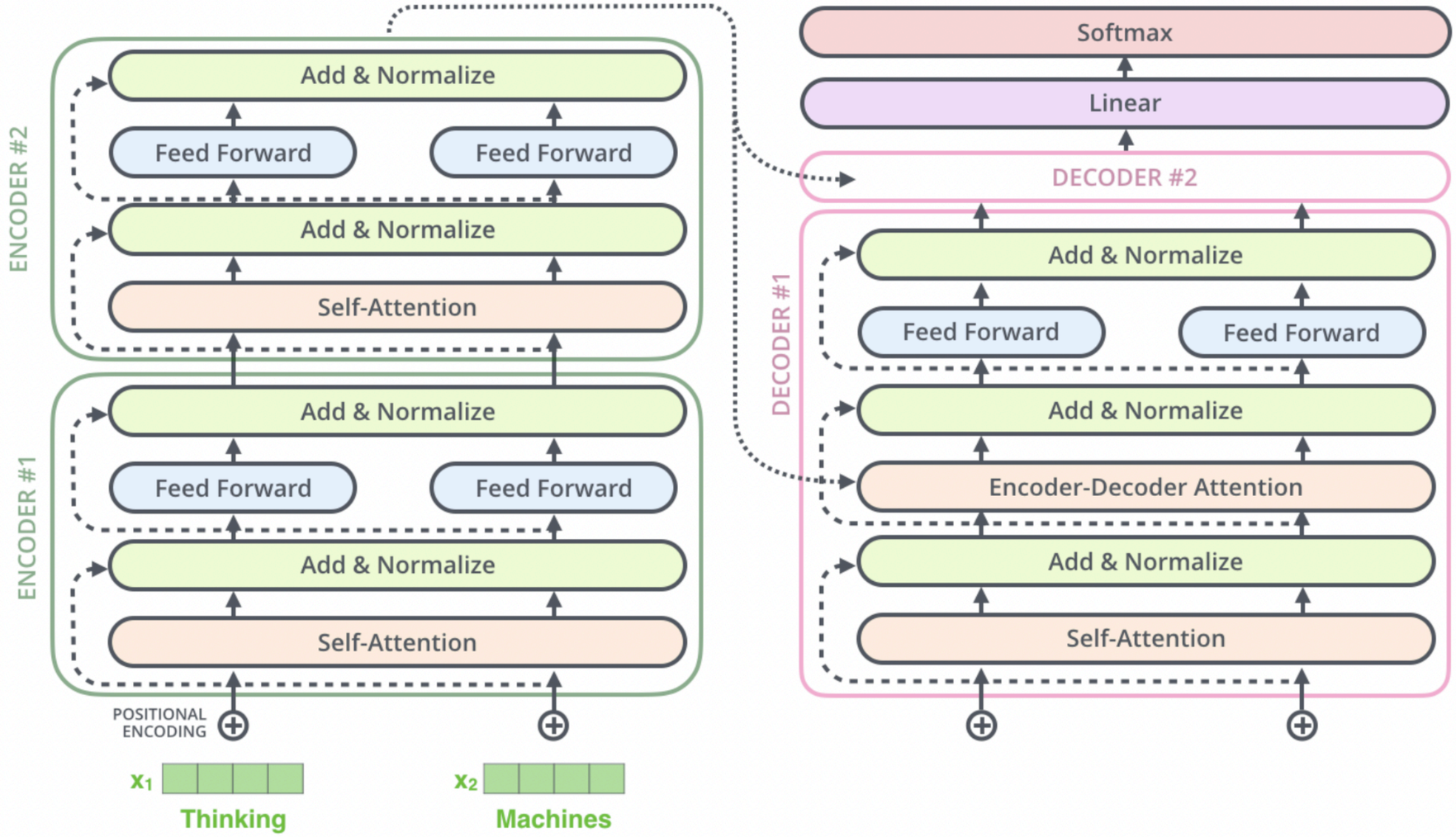
The Illustrated Transformer

The Residuals & layer-normalization



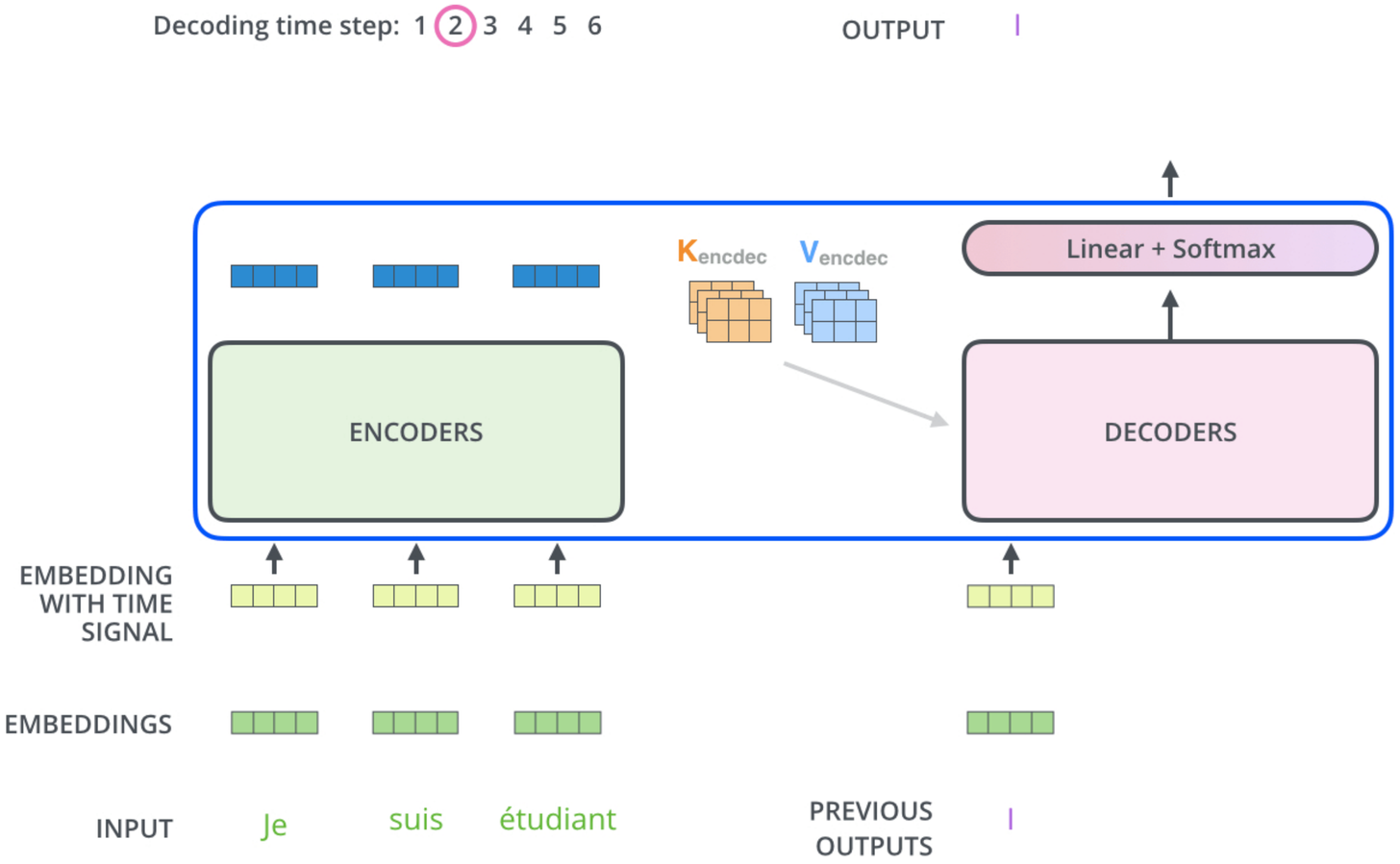
The Illustrated Transformer

Transformer: Decoder Side

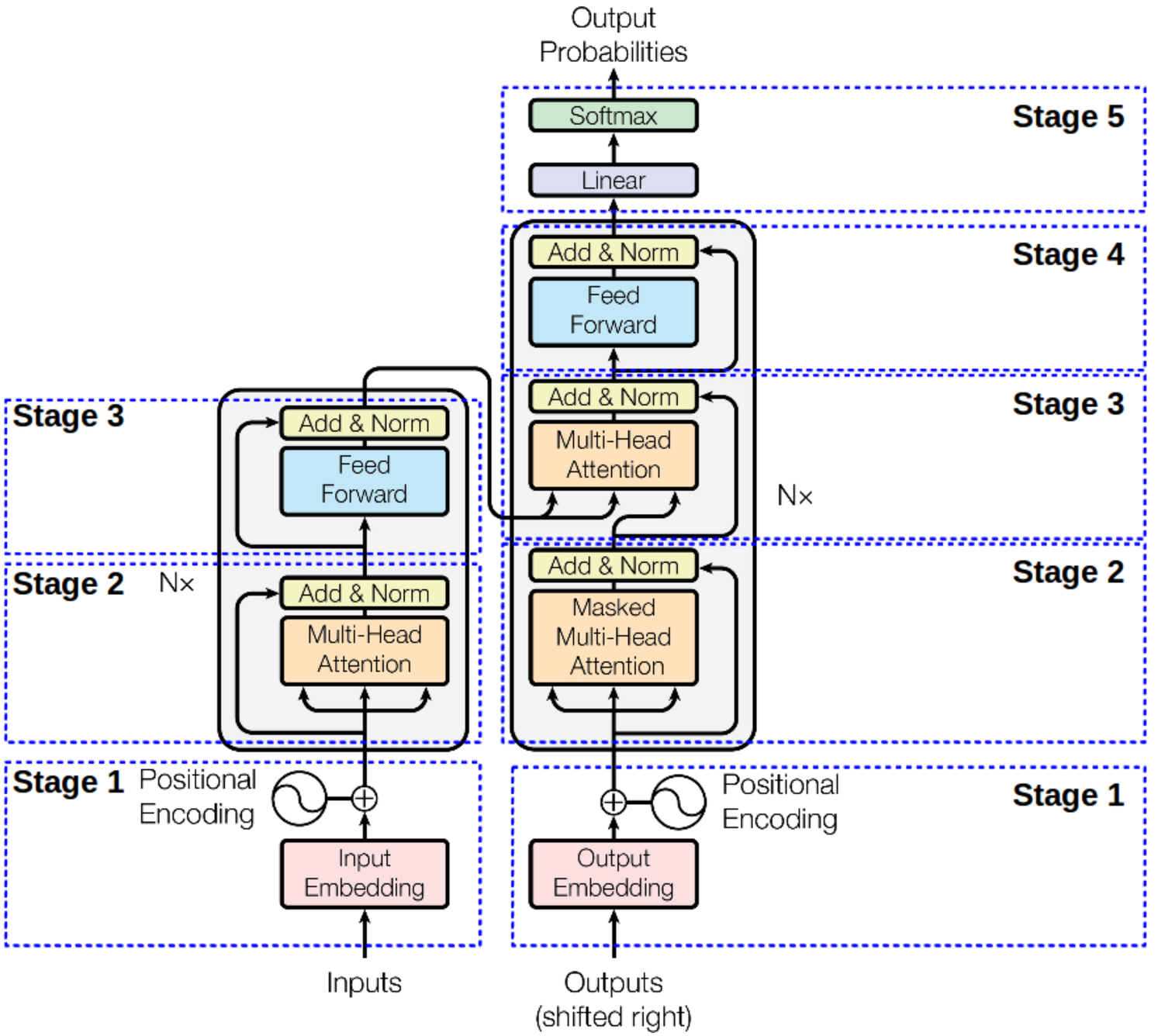


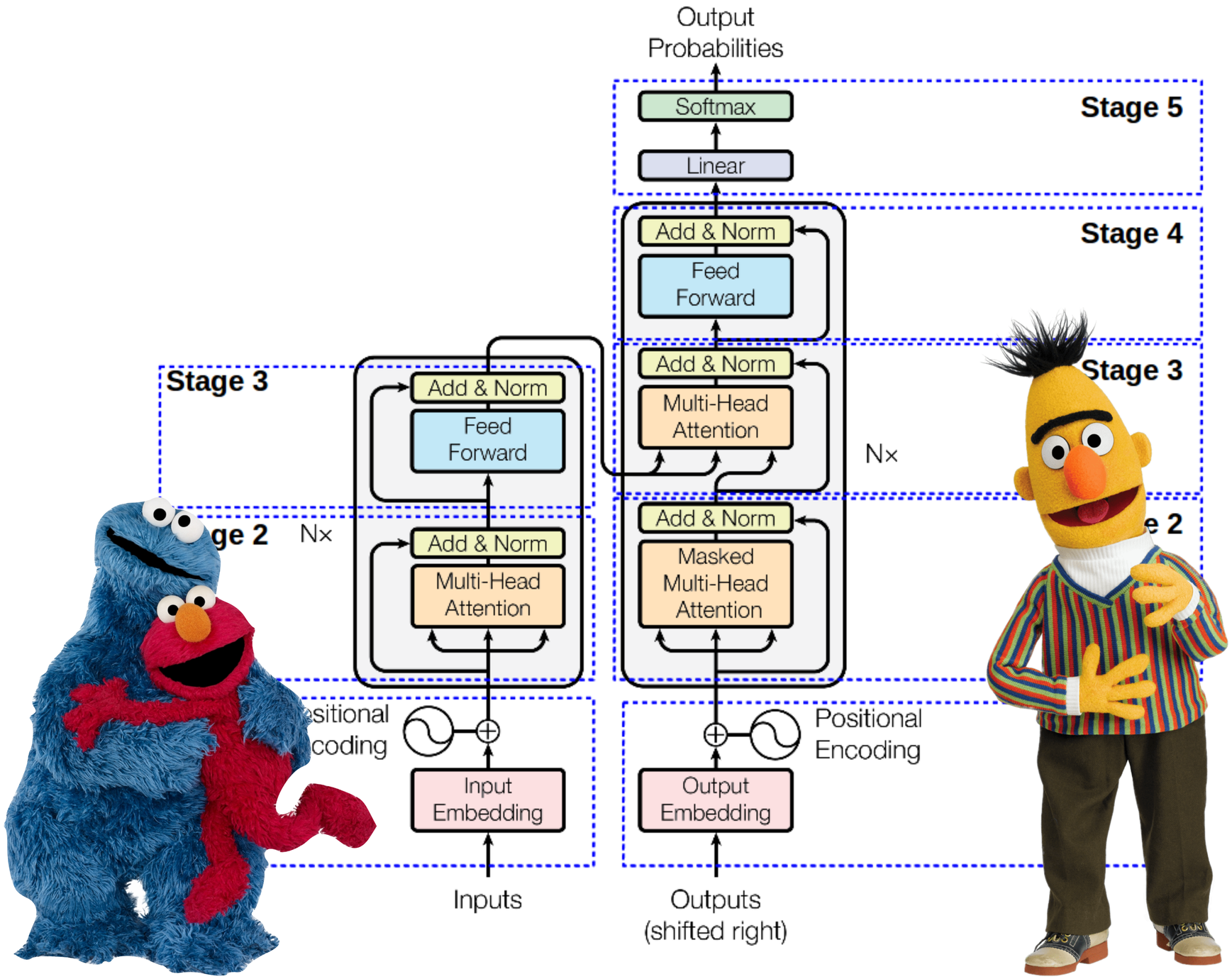
The Illustrated Transformer

Transformer: Decoder Side

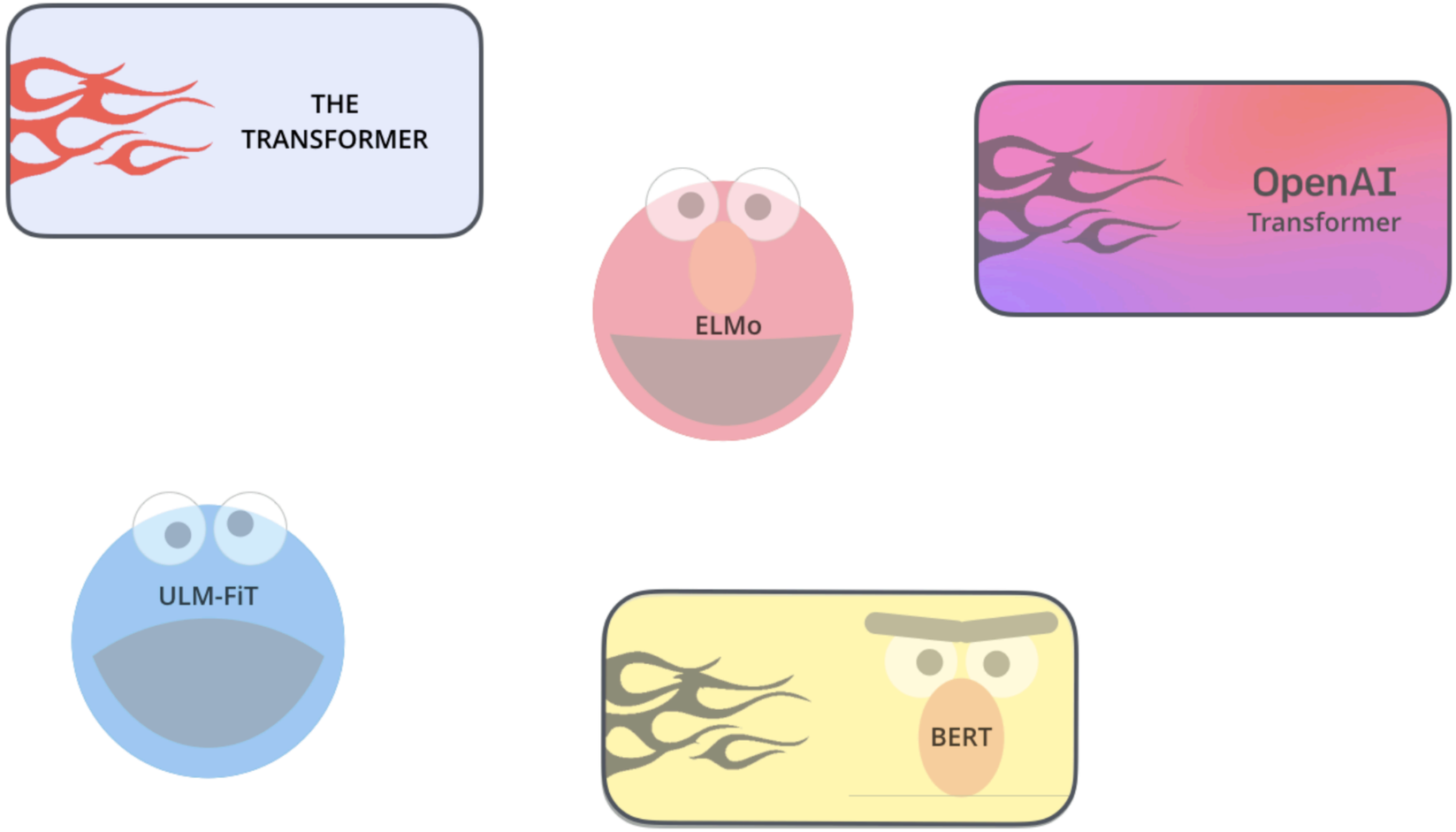


The Illustrated Transformer


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Input Embedding
Positional Encoding
Output Embedding
Positional Encoding
Stage 5
Stage 4
Stage 3
Stage 2
Stage 1
Stage 3
Stage 2
Nx
Stage 1
Inputs
Outputs (shifted right)

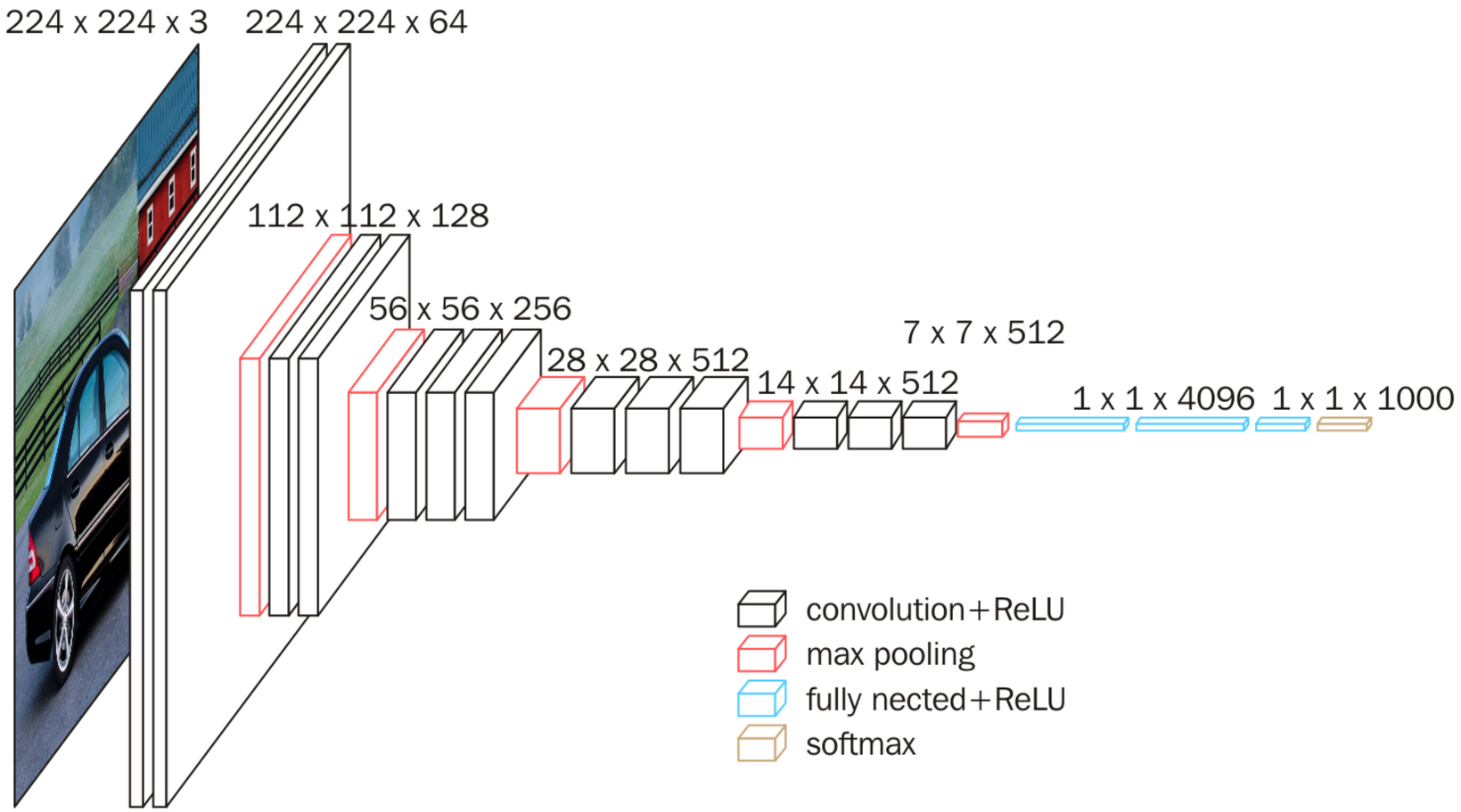

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)
Stage 5
Stage 4
Stage 3
Stage 2
Stage 3

Другие модели



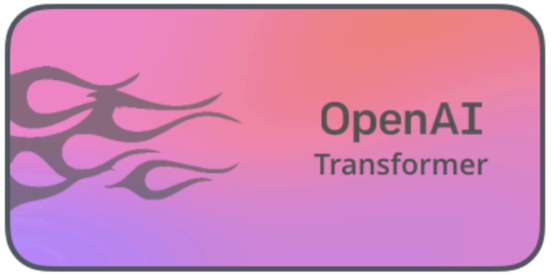
The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)

Recap: Applying transfer learning in CV



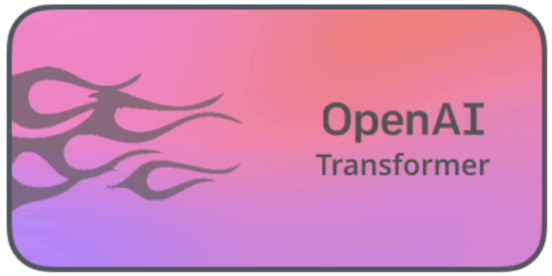
Transfer learning in NLP?

Другие модели

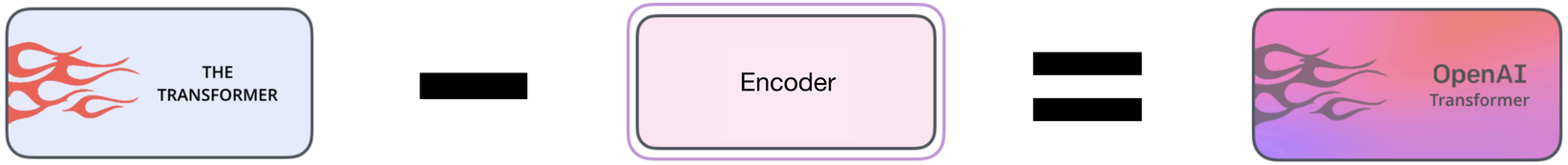


The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)

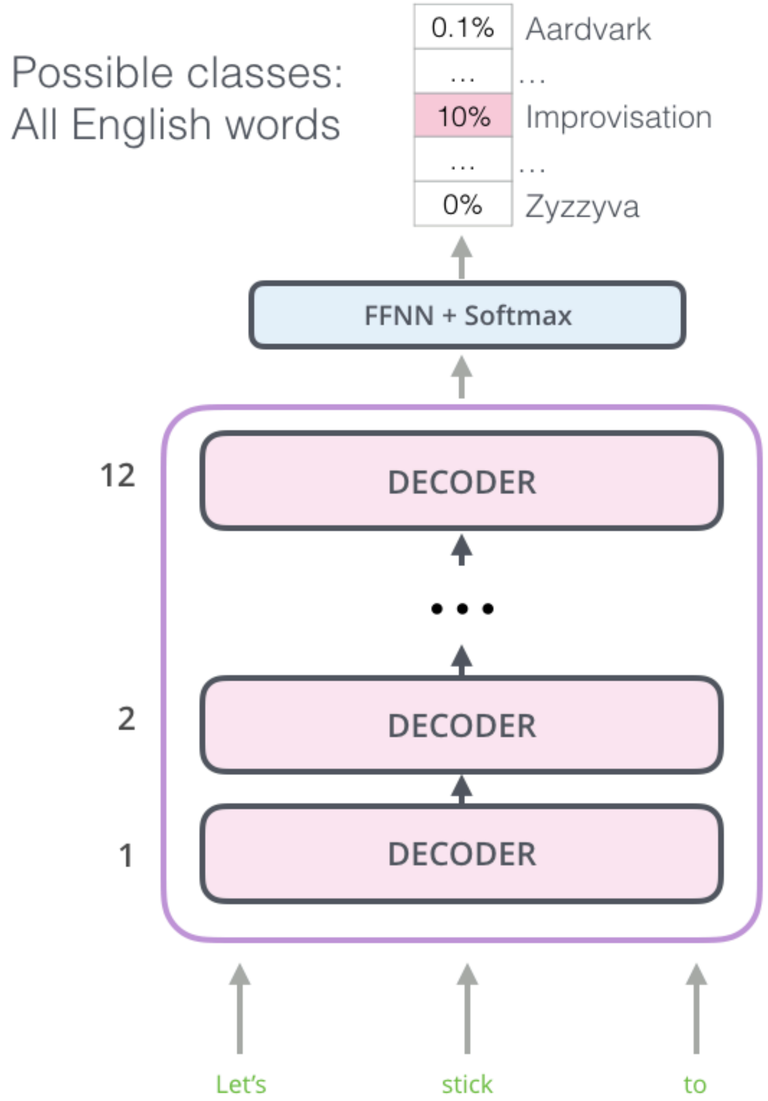
OpenAI Transformer



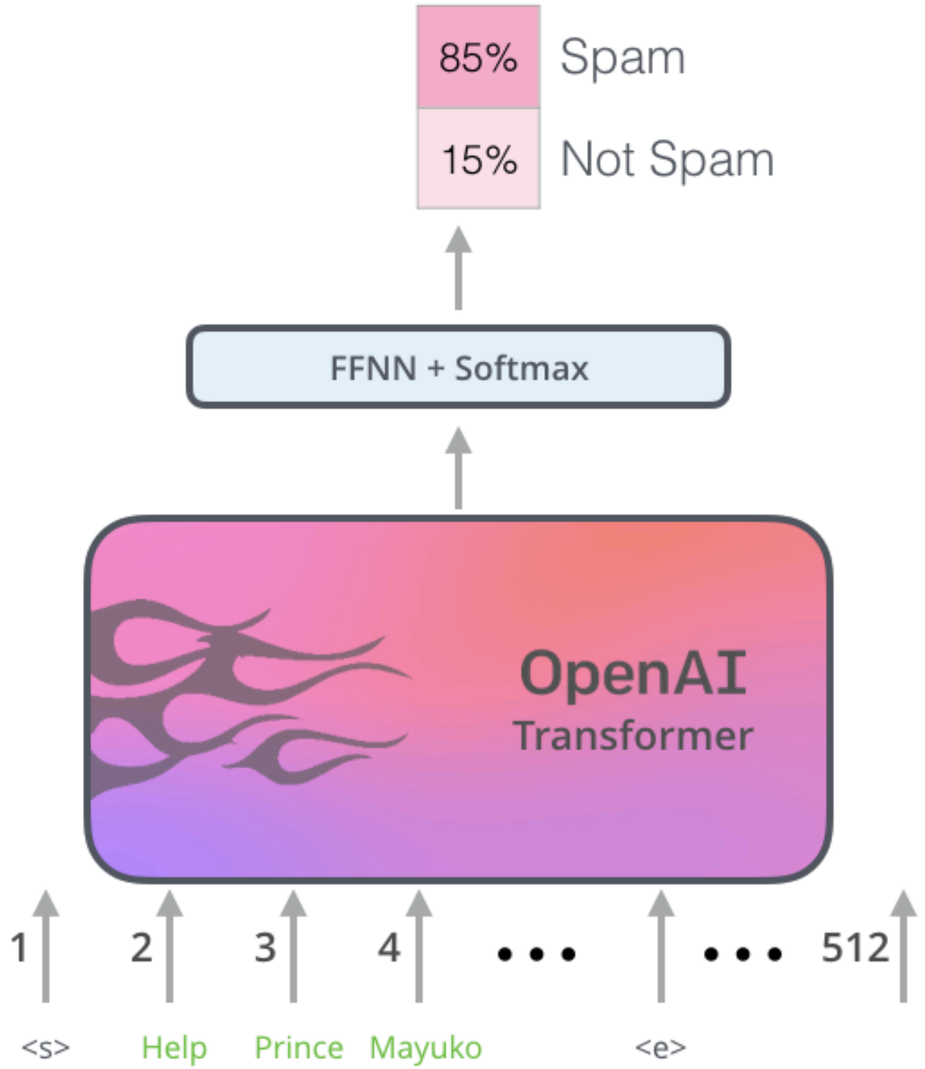
Pre-training a Transformer Decoder for Language Modeling



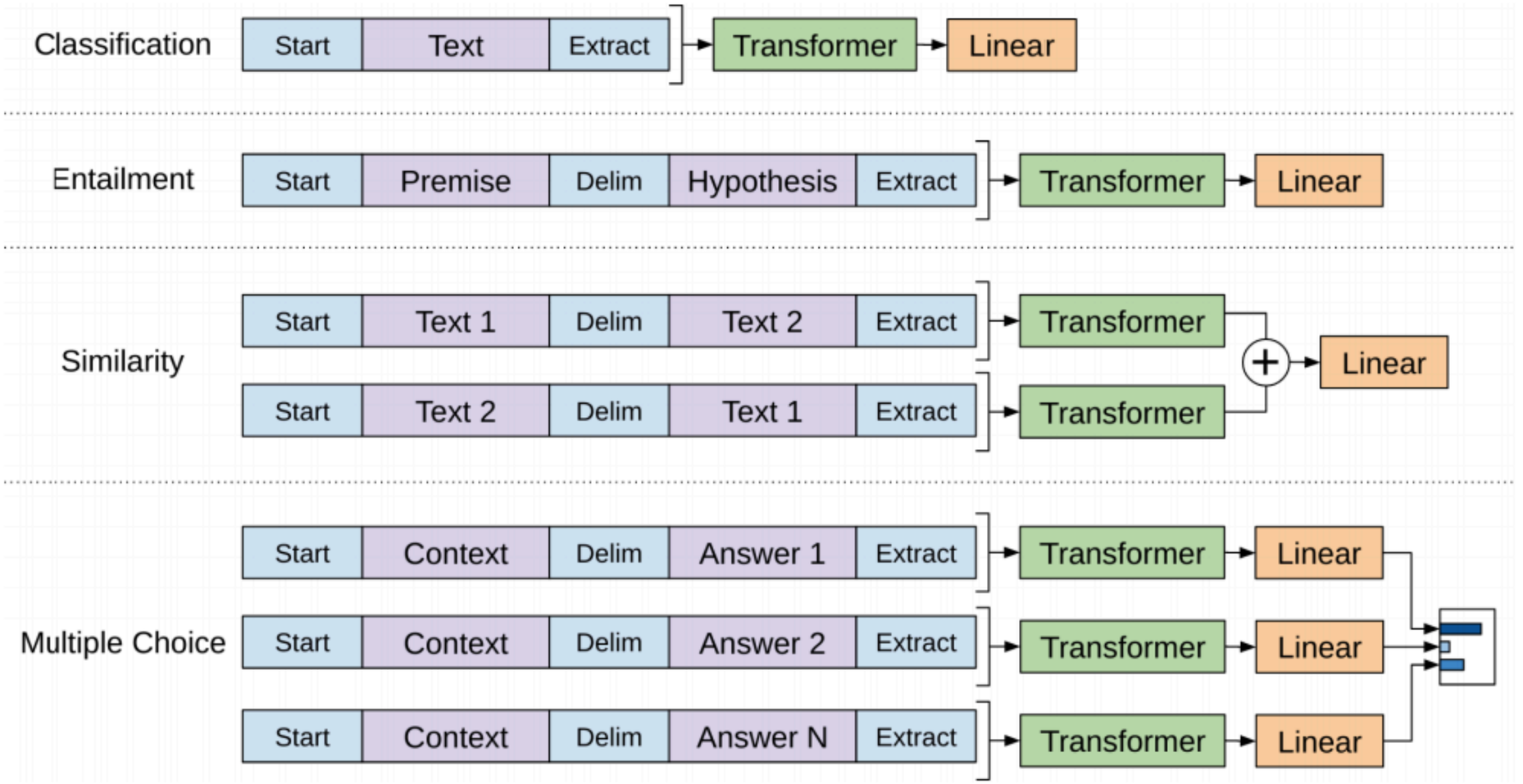
OpenAI Transformer



OpenAI Transformer



OpenAI Transformer



План лекции

The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)

ELMo

1. Expedited Labour Market Opinion
2. Electric Light Machine Organization
3. Enough Let’s Move On

Special thx for @Anastasia Yanina:

https://github.com/ml-mipt/ml-mipt/blob/advanced/week05_BERT_and_LDA/Lecture_BERT_DIHT.pdf

ELMo

1. Expedited Labour Market Opinion
2. Electric Light Machine Organization
3. Enough Let’s Move On

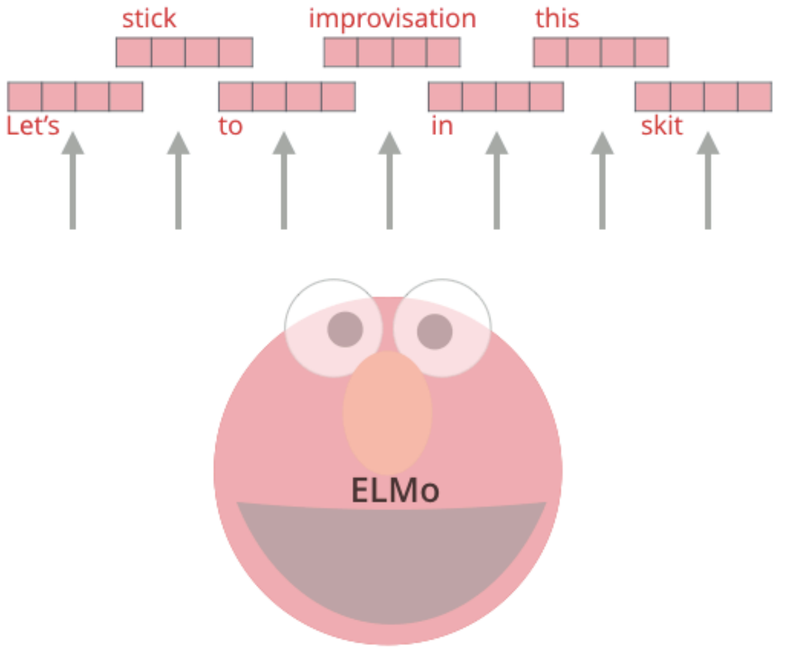
4. Embeddings from Language Models

Special thx for @Anastasia Yanina:

https://github.com/ml-mipt/ml-mipt/blob/advanced/week05_BERT_and_LDA/Lecture_BERT_DIHT.pdf

ELMo

ELMo Embeddings



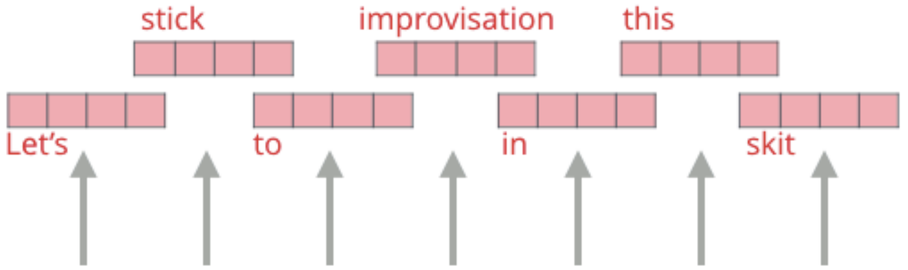
Words to embed

Let's stick to improvisation in this skit


Hey ELMo, what's the embedding of the word "stick"?
There are multiple possible embeddings! Use it in a sentence.
Oh, okay. Here: "Let's stick to improvisation in this skit"
Oh in that case, the embedding is: -0.02, -0.16, 0.12, -0.1 ….etc

ELMo

ELMo Embeddings



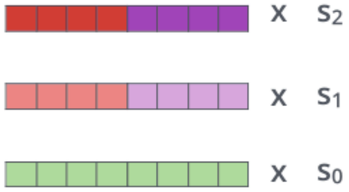
Words to embed

ELMo

Embedding of “stick” in “Let’s stick to” - Step #2

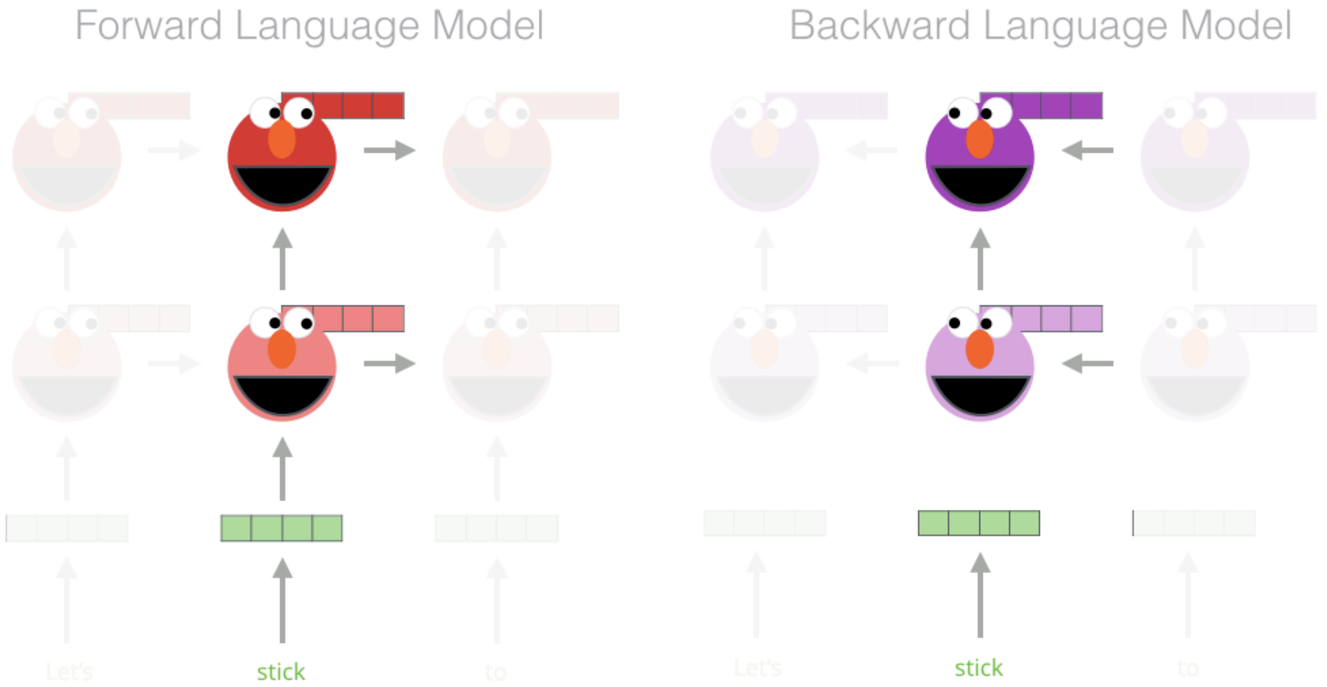
1- Concatenate hidden layers

2- Multiply each vector by a weight based on the task

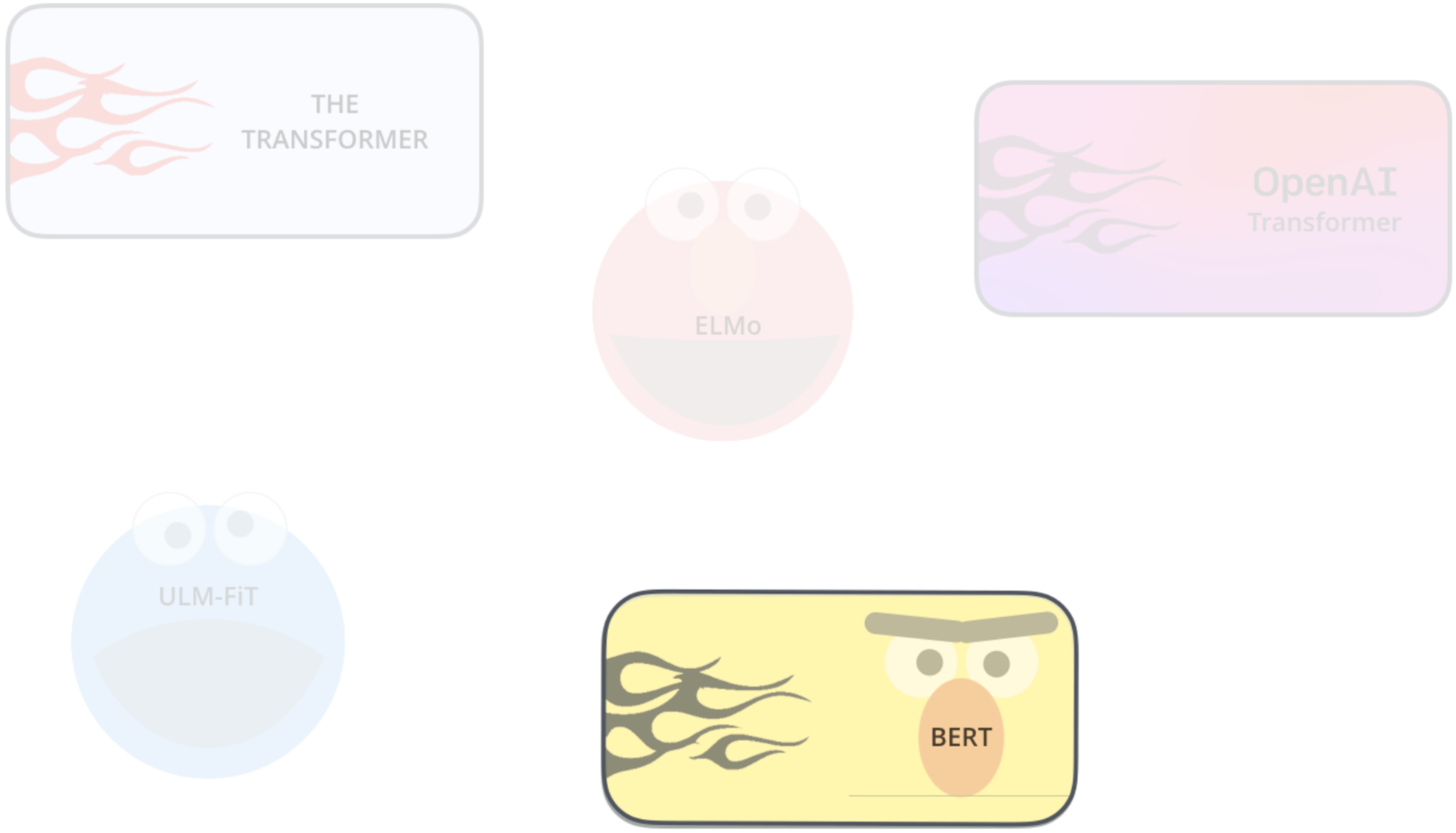


3- Sum the (now weighted) vectors

ELMo embedding of “stick” for this task in this context



Другие модели

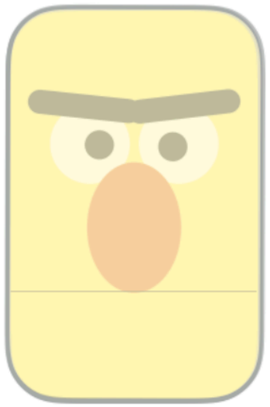


The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)

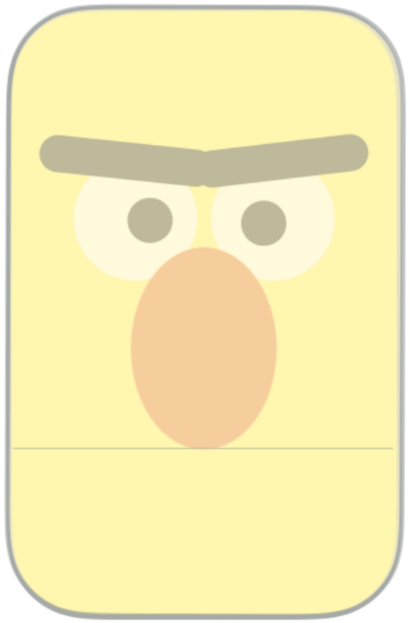
BERT

[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT



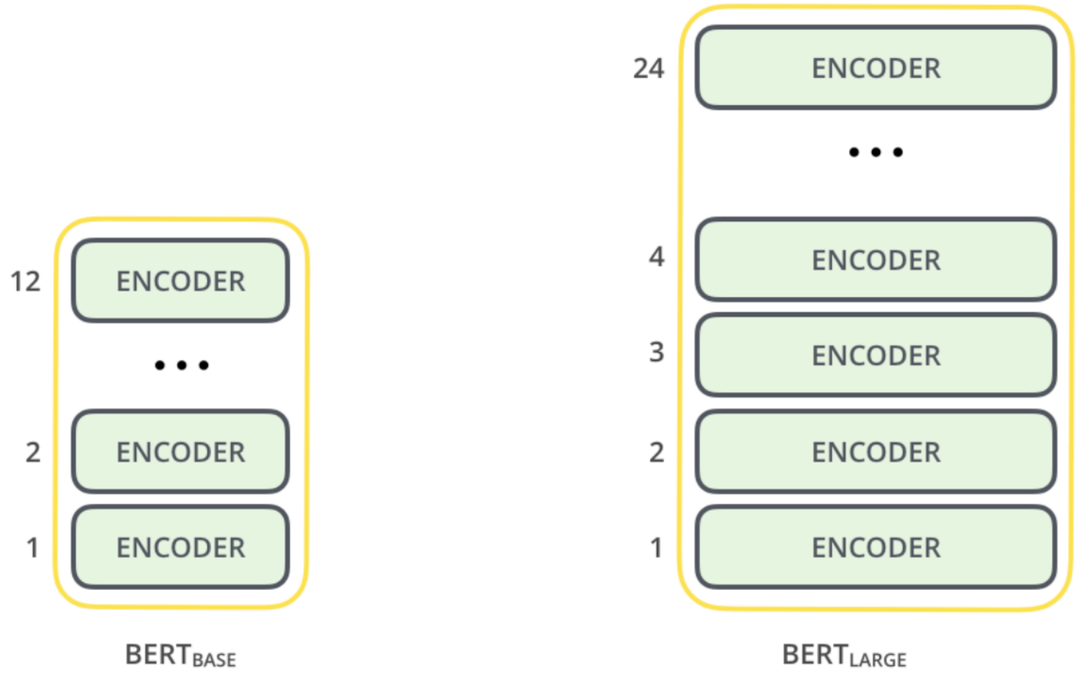
BERT$_{BASE}$



BERT$_{LARGE}$

[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

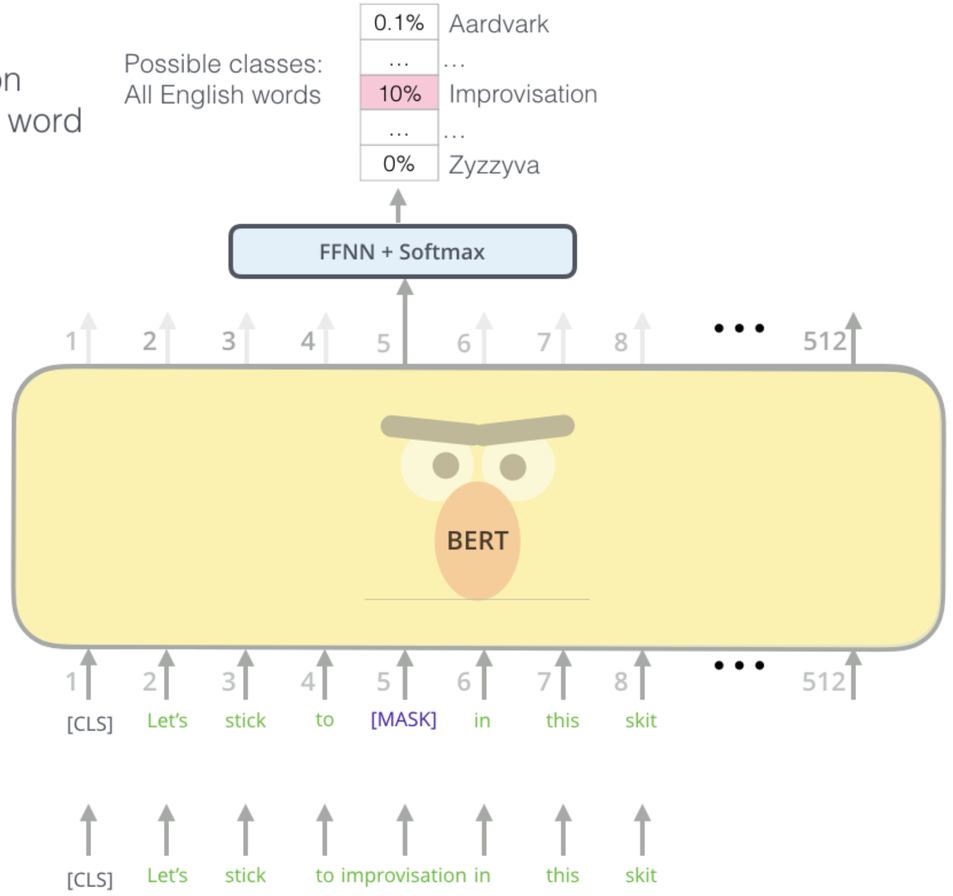
BERT



[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

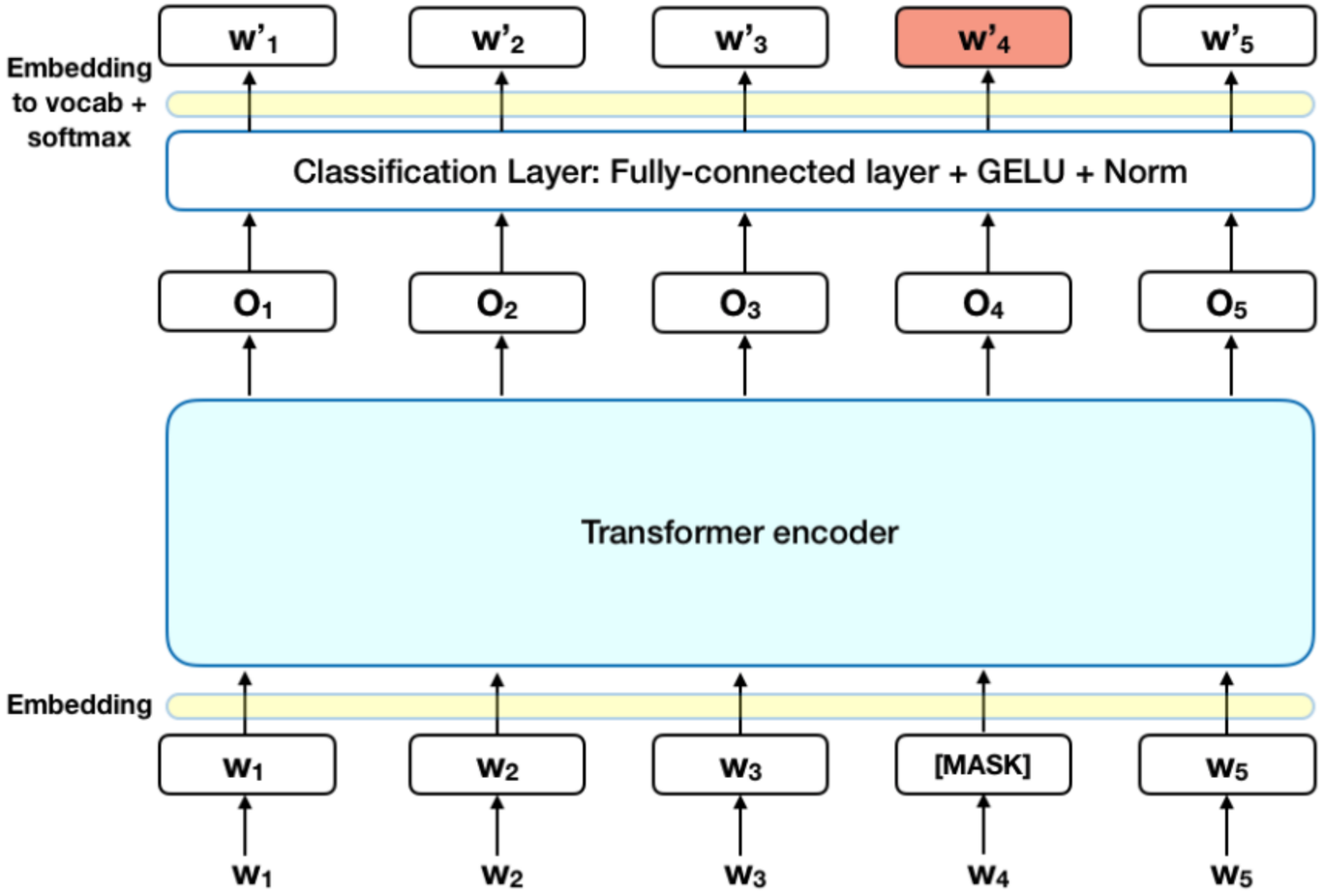
BERT

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



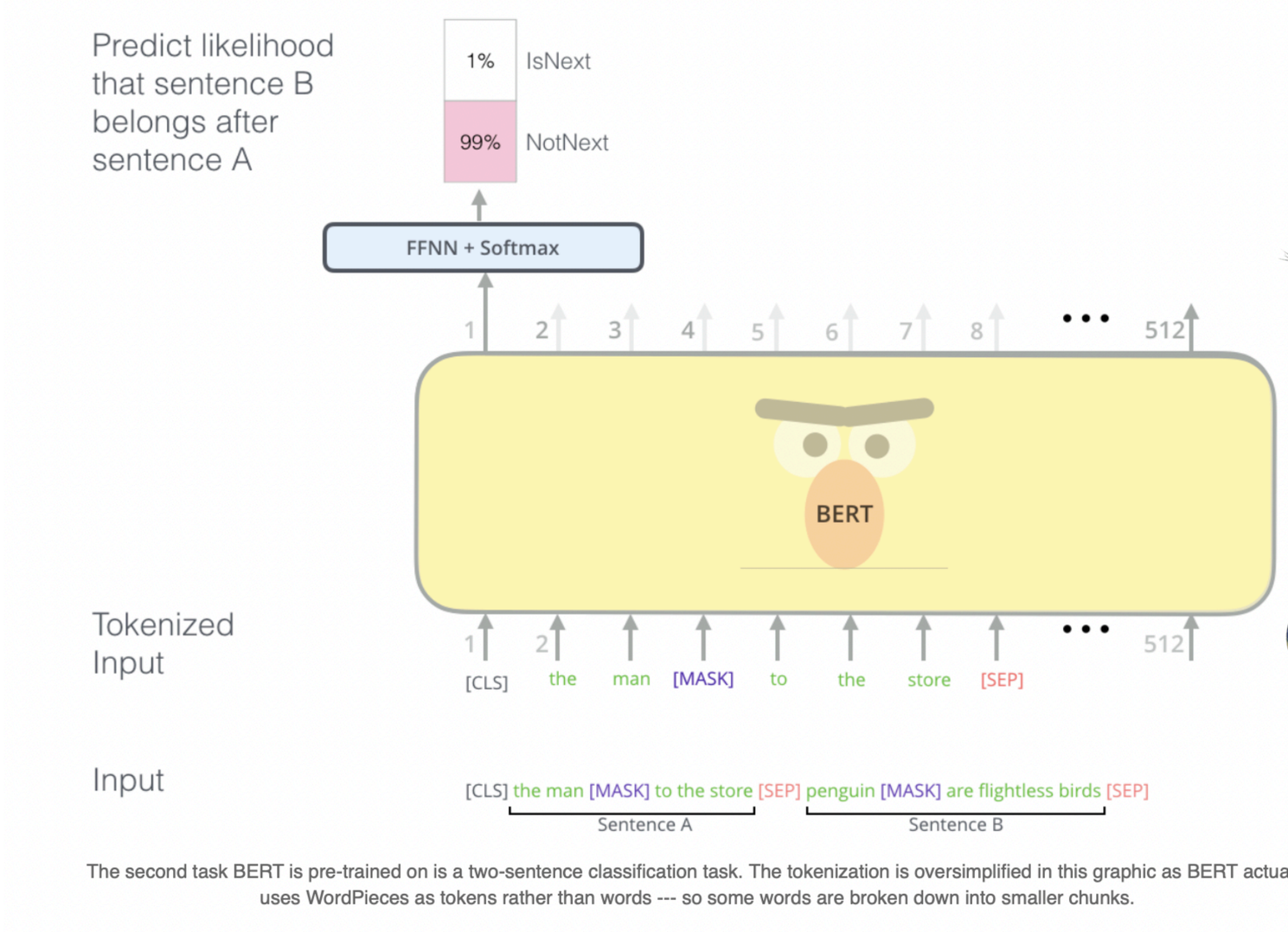
[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT



[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT



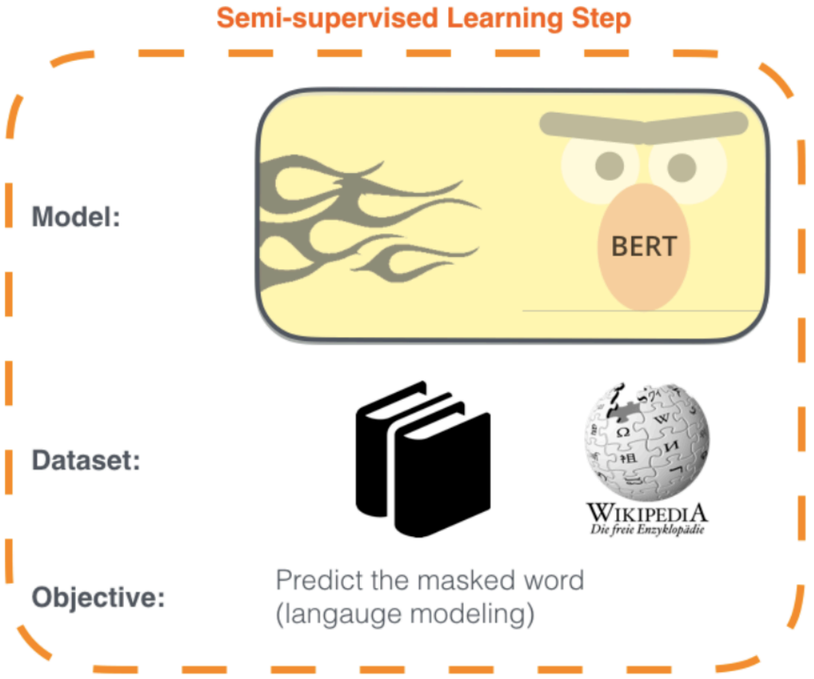
The second task BERT is pre-trained on is a two-sentence classification task. The tokenization is oversimplified in this graphic as BERT actually uses WordPieces as tokens rather than words --- so some words are broken down into smaller chunks.

[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

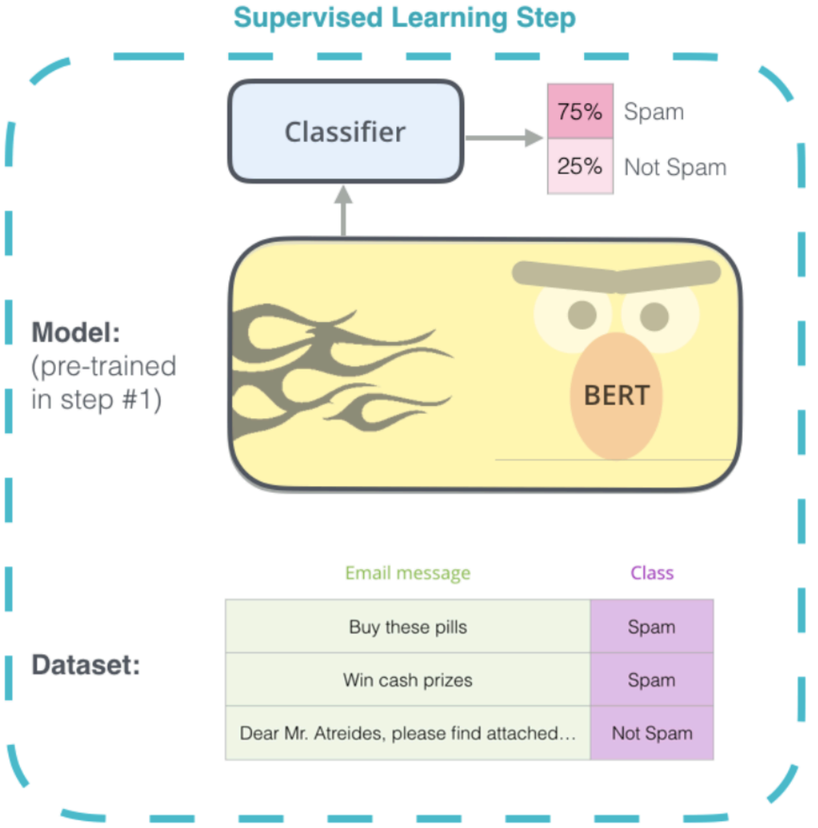
BERT

1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

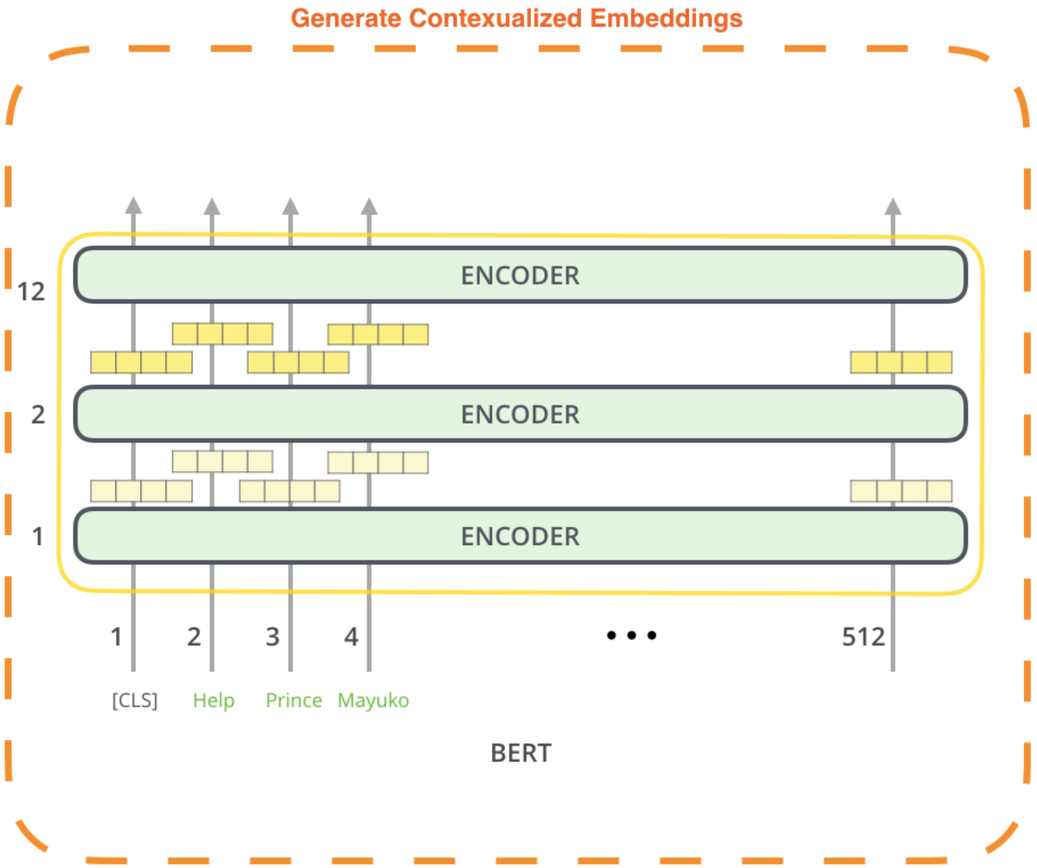


2 - Supervised training on a specific task with a labeled dataset.

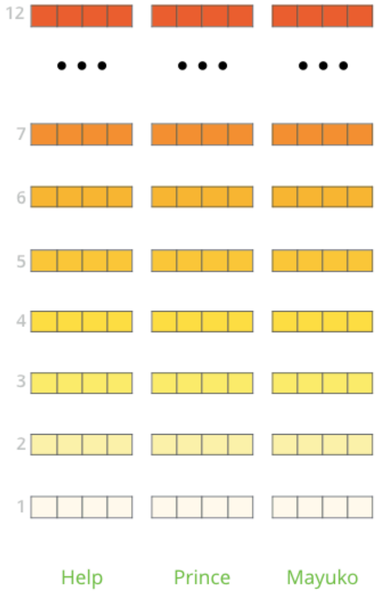


[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT for feature extraction



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

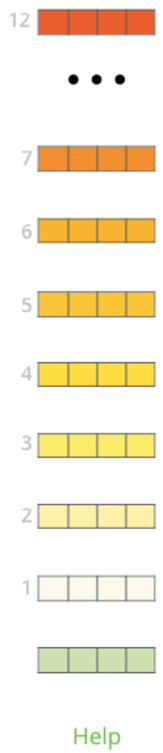


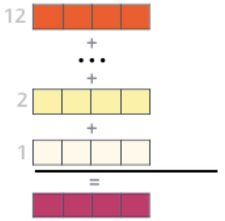
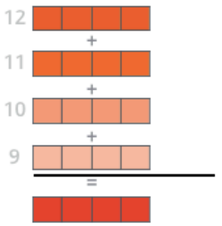
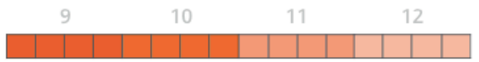
But which one should we use?

[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT for feature extraction

What is the best contextualized embedding for "Help" in that context?
For named-entity recognition task CoNLL-2003 NER

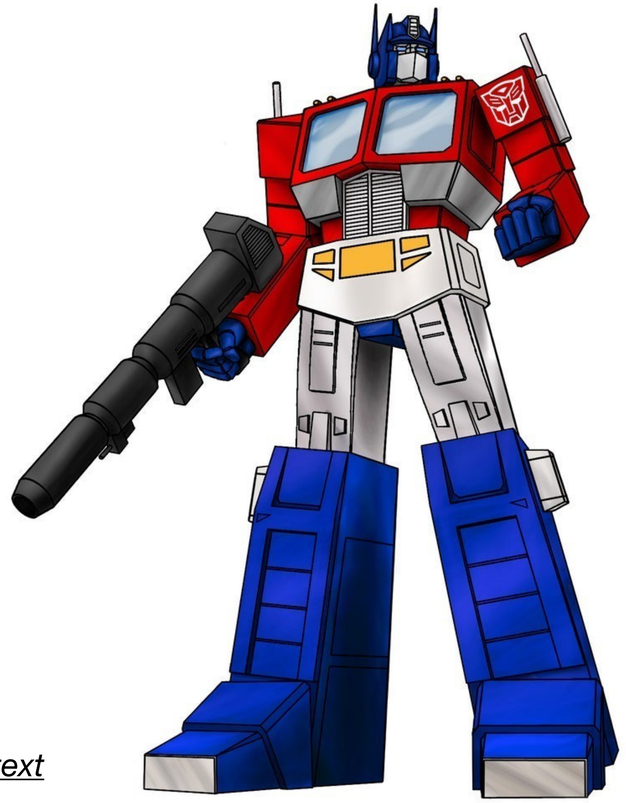


		Dev F1 Score
First Layer		91.0
Last Hidden Layer		94.9
Sum All 12 Layers		95.5
Second-to-Last Hidden Layer		95.6
Sum Last Four Hidden		95.9
Concat Last Four Hidden		96.1

[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Feature of NLP

Transformer XL



[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

Чем плох Open AI Transformer?

[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

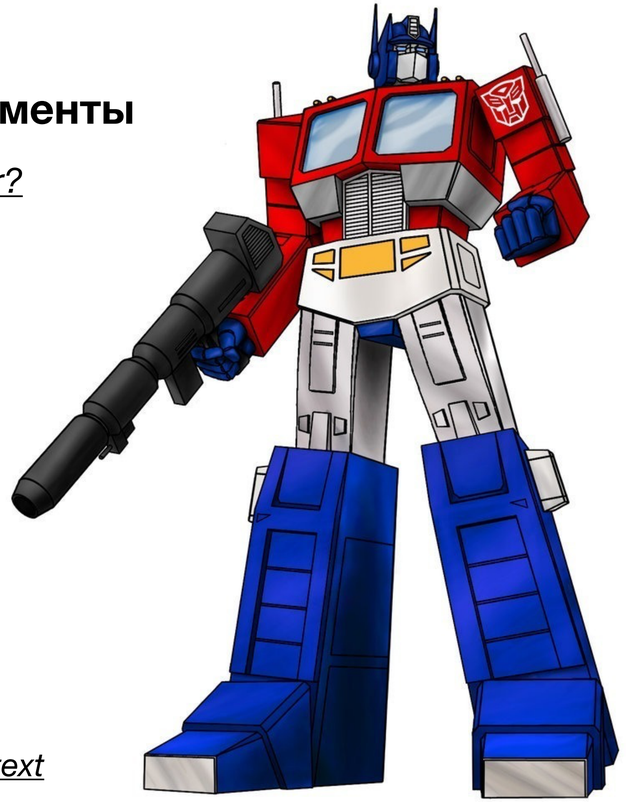
Давайте разобьем наш текст на сегменты

[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

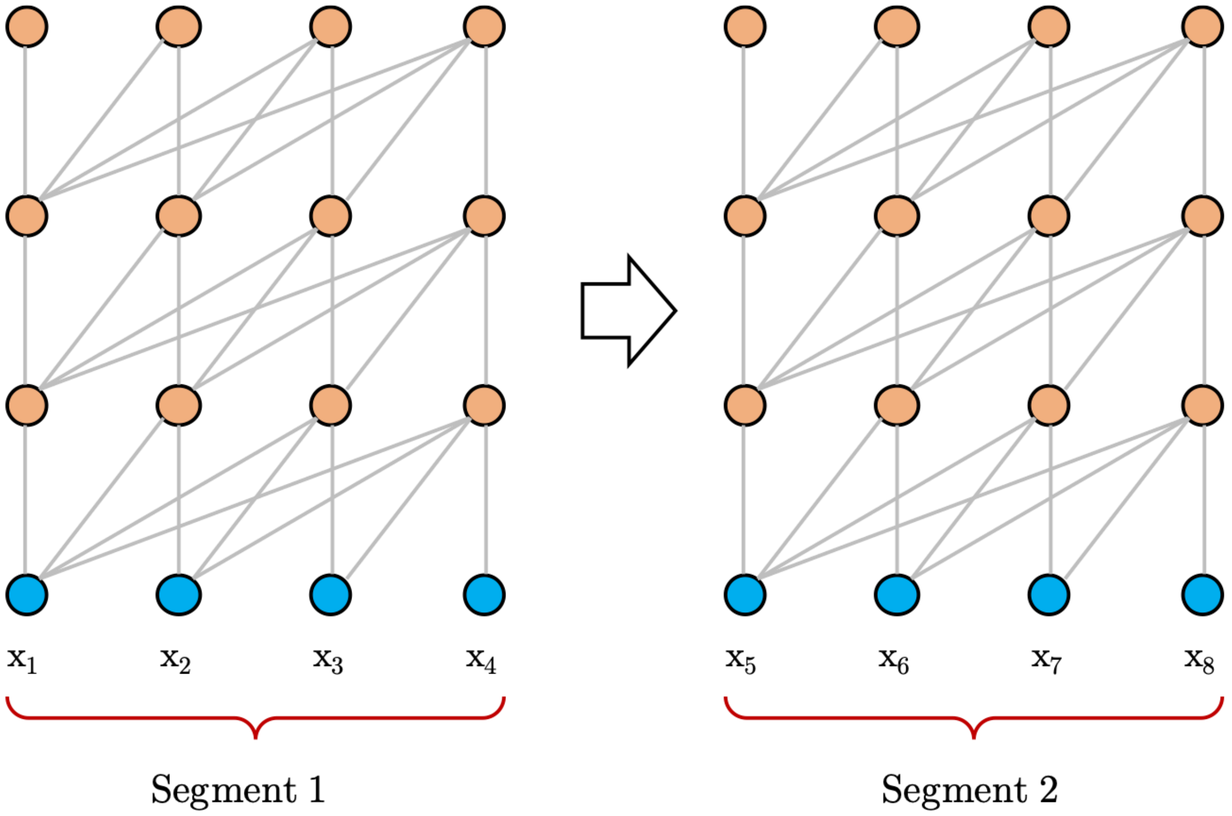
Давайте разобьем наш текст на сегменты

Как будет работать обычный transformer?



[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

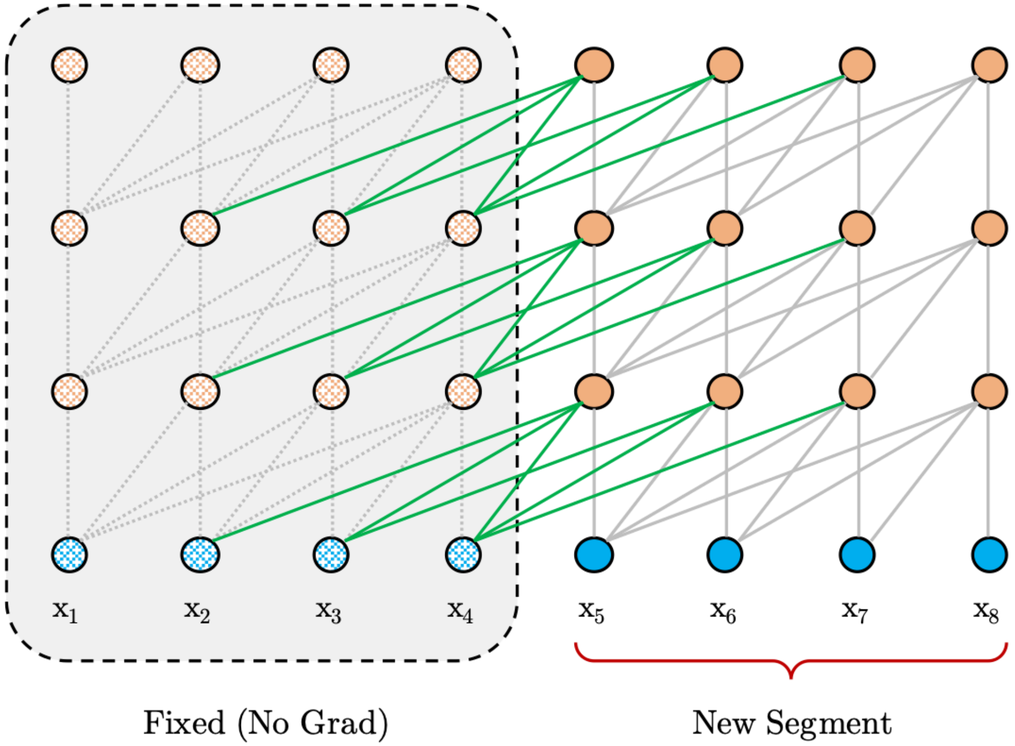
Transformer XL



[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

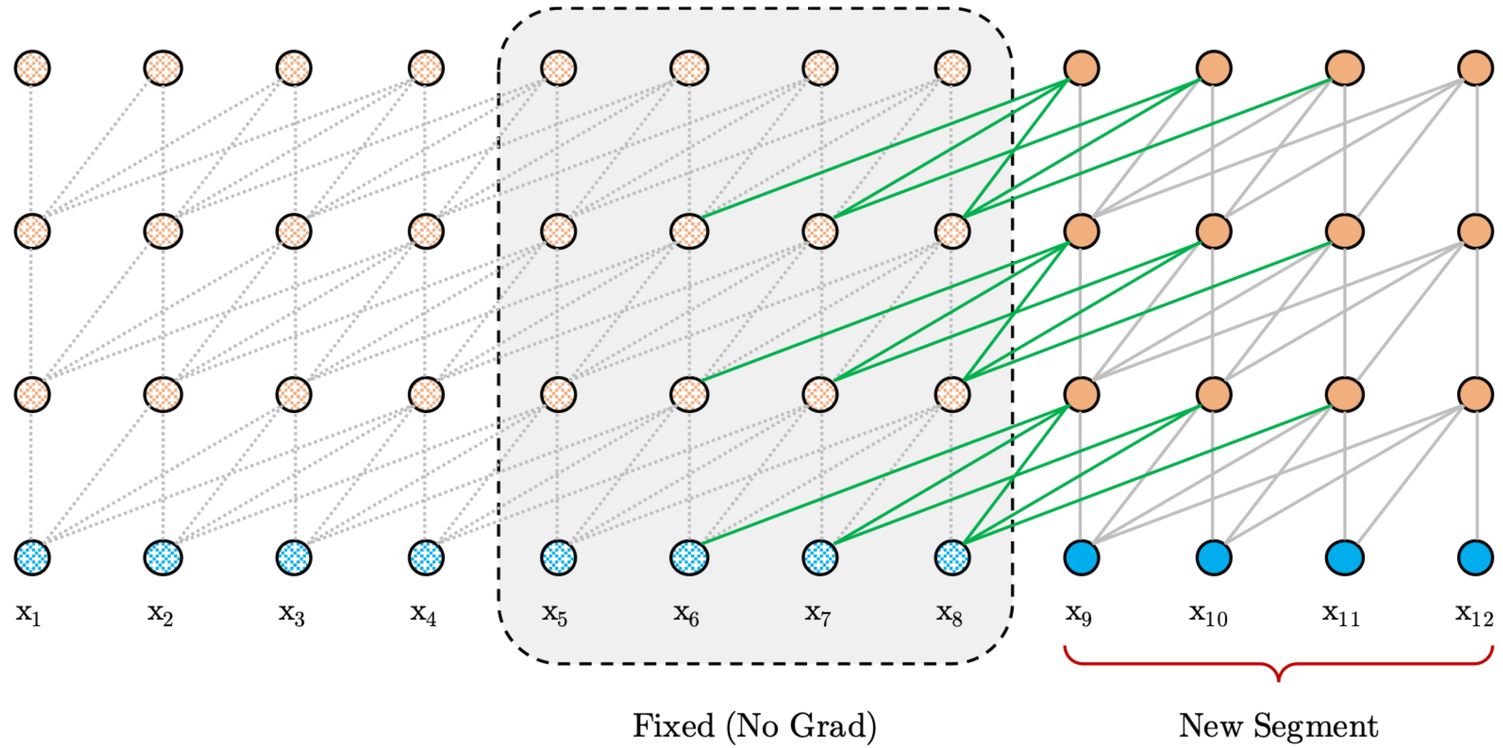
Segment-level recurrence:



[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

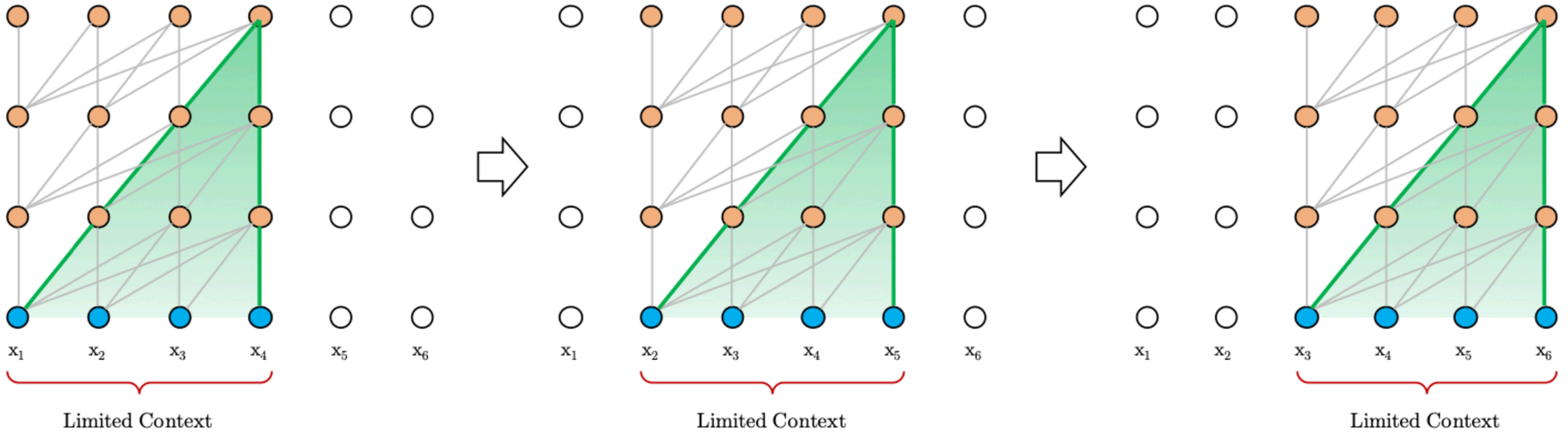
Segment-level recurrence:



[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

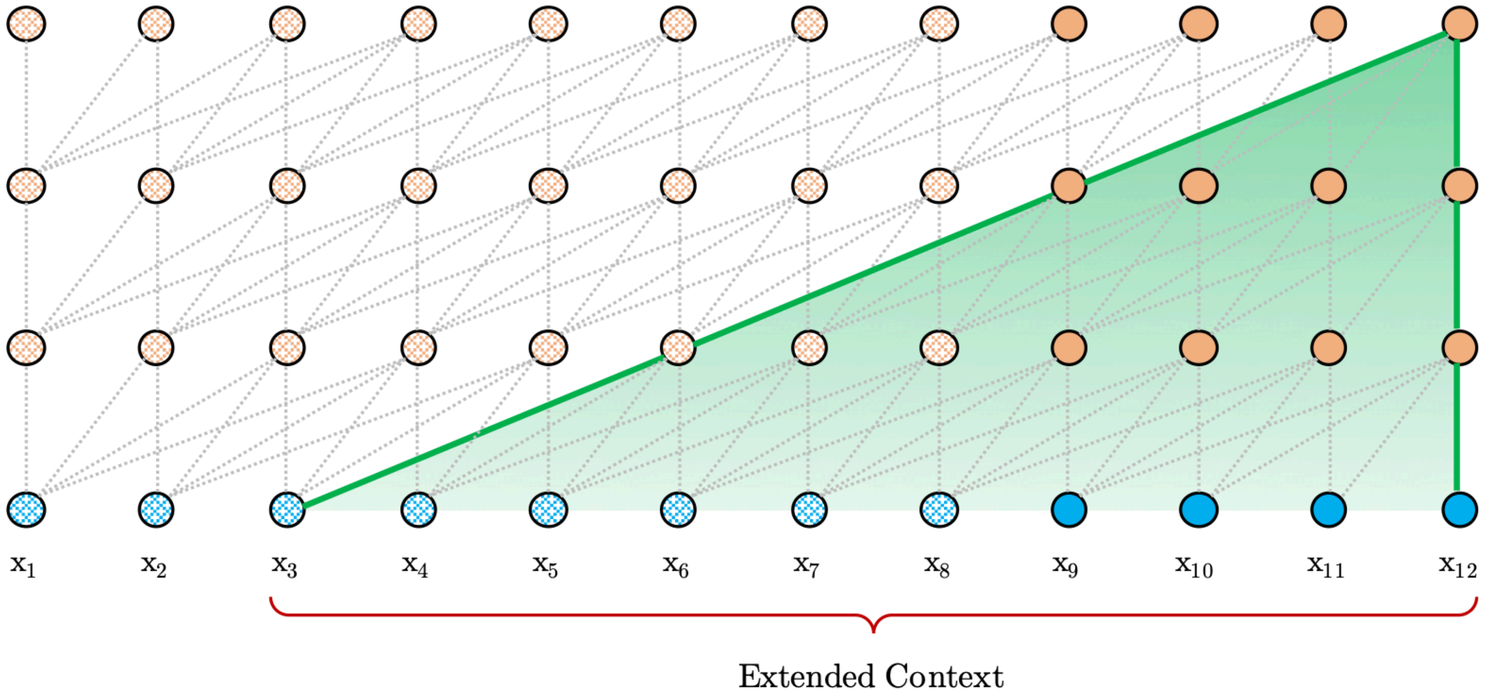
Evaluation phase of Vanilla transformer:



[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Transformer XL

Evaluation phase of Transformer XL:

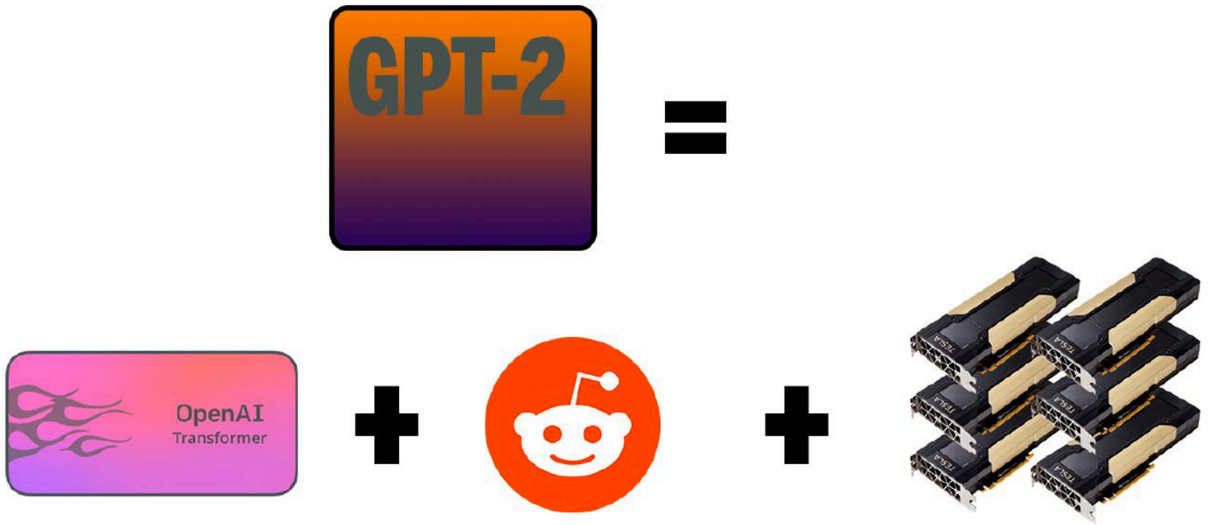


[1901.02860] Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

SOTA GPT-2

SOTA GPT-2

- Transformer-based architecture
- trained to predict the **next** word
- 1.5 billion parameters
- Trained on 8 million web-pages



https://openai.com/blog/gpt-2-6-month-follow-up/

https://github.com/ml-mipt/ml-mipt/blob/advanced/week05_BERT_and_LDA/Lecture_BERT_DIHT.pdf

GPT-2: fake news and hype

Top stories

OpenAI built a text generator so good, it's considered too dangerous to release

TechCrunch

11 hours ago

Elon Musk's AI company created a fake news generator it's too scared to make public

BGR.com

9 hours ago

The AI That Can Write A Fake News Story From A Handful Of Words

NDTV.com

2 hours ago