

Language Models

Extended Learning

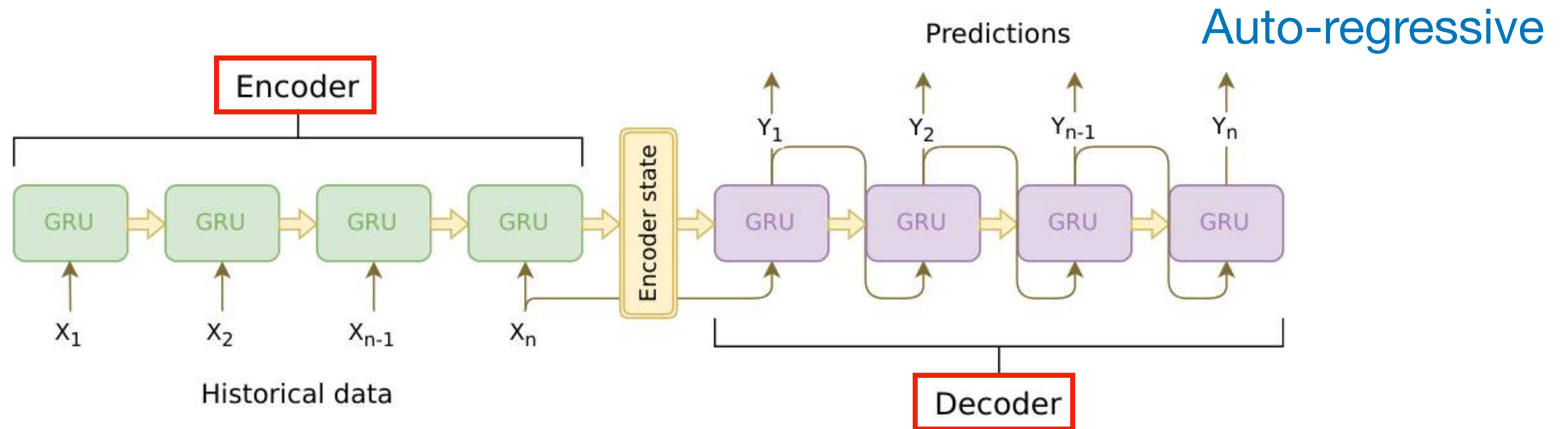
Fang-Yi Su
Nov. 26, 2024

Getting Ready for the Lesson

- I suppose you have already understood
 - Basics of deep learning
 - The structure and working of deep neural networks
 - Layers, activation functions, ...
 - The mechanism of loss functions
 - Concepts of w2v
 - CBOW, skip-gram, GloVe

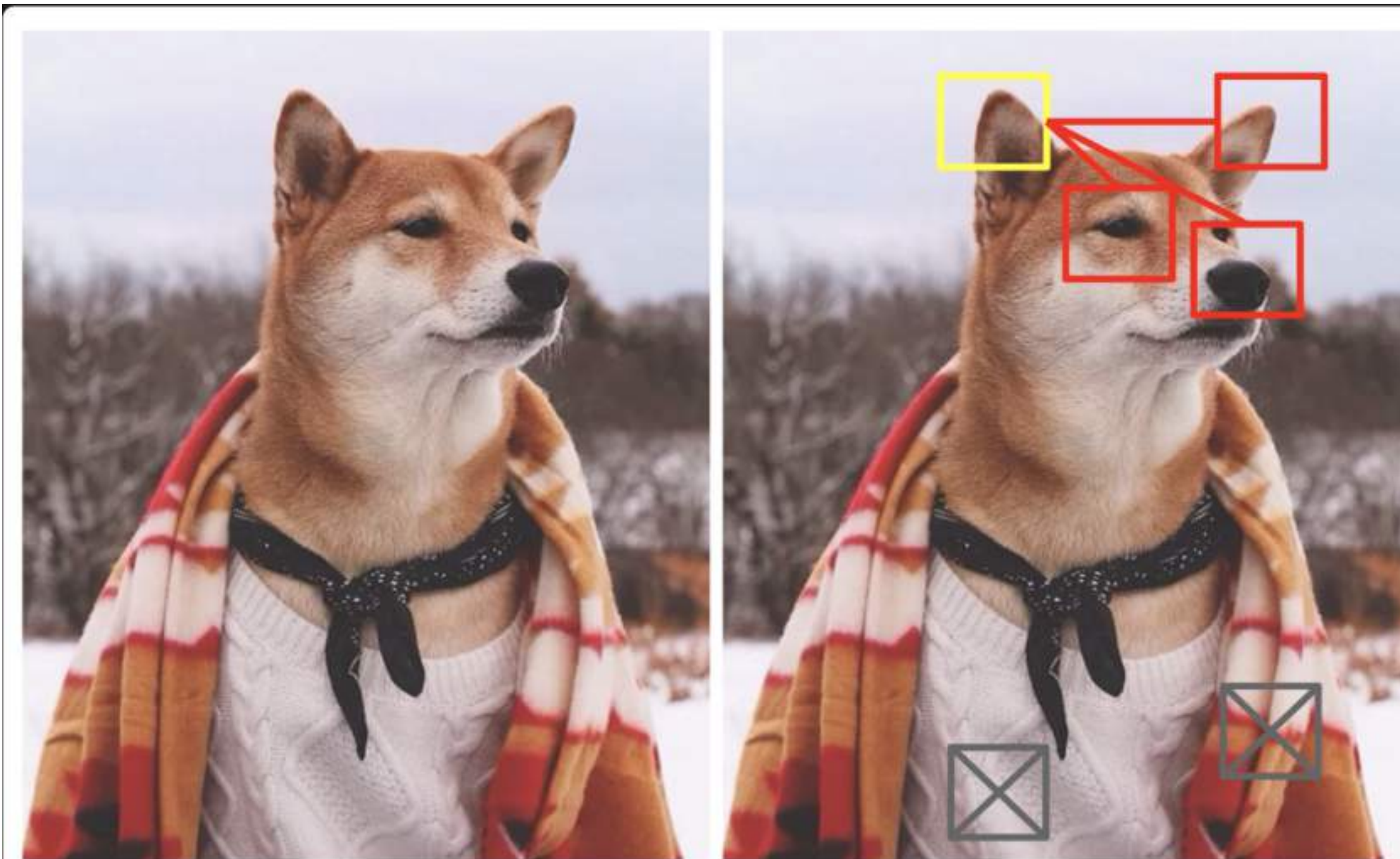
Seq2Seq Models

- It aims to transform an input sequence (source) to a new one (target) and both sequences can be of **arbitrary** lengths.



Attention

- Attention is motivated by how we pay visual attention to different regions of an image or correlate words in one sentence.



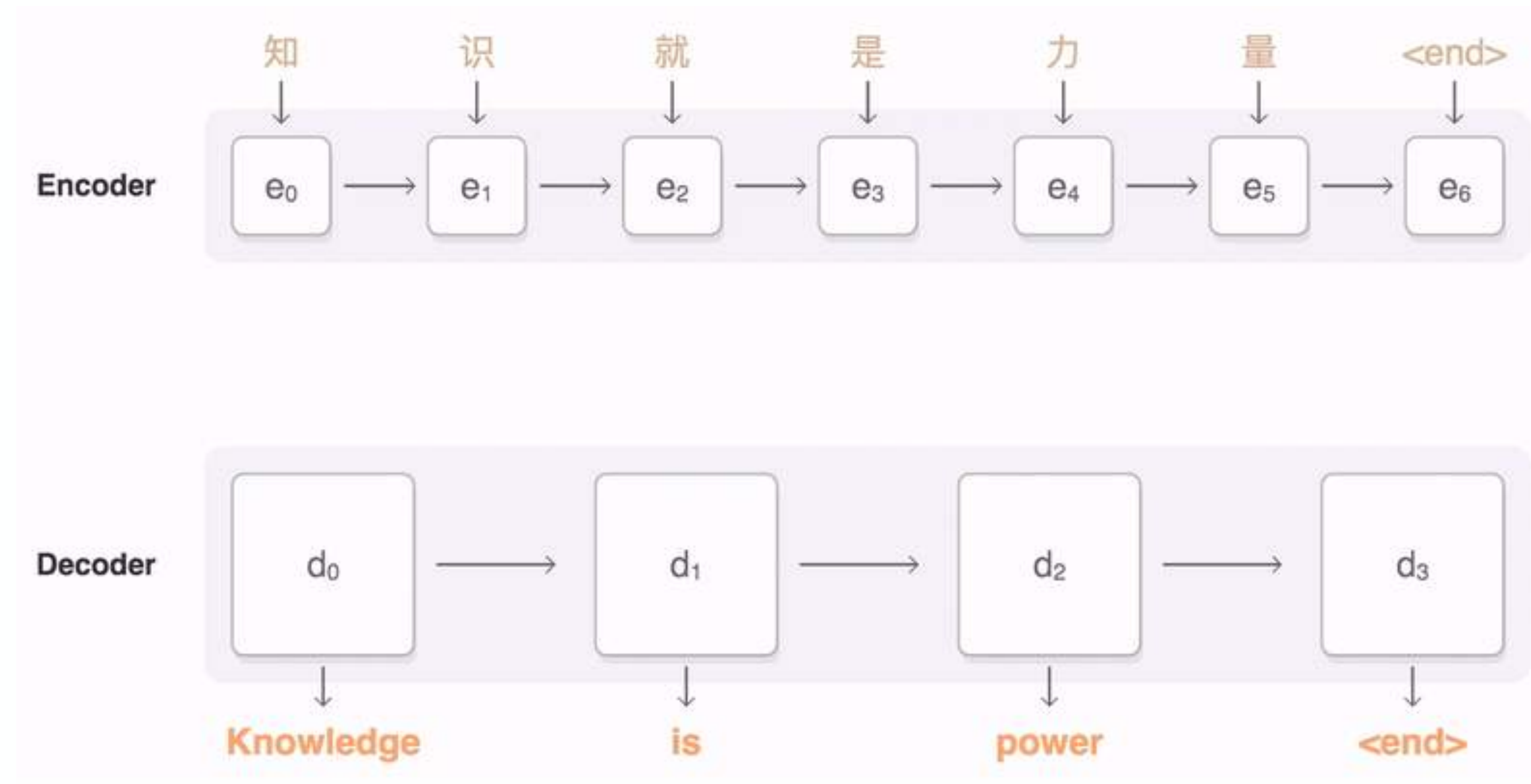
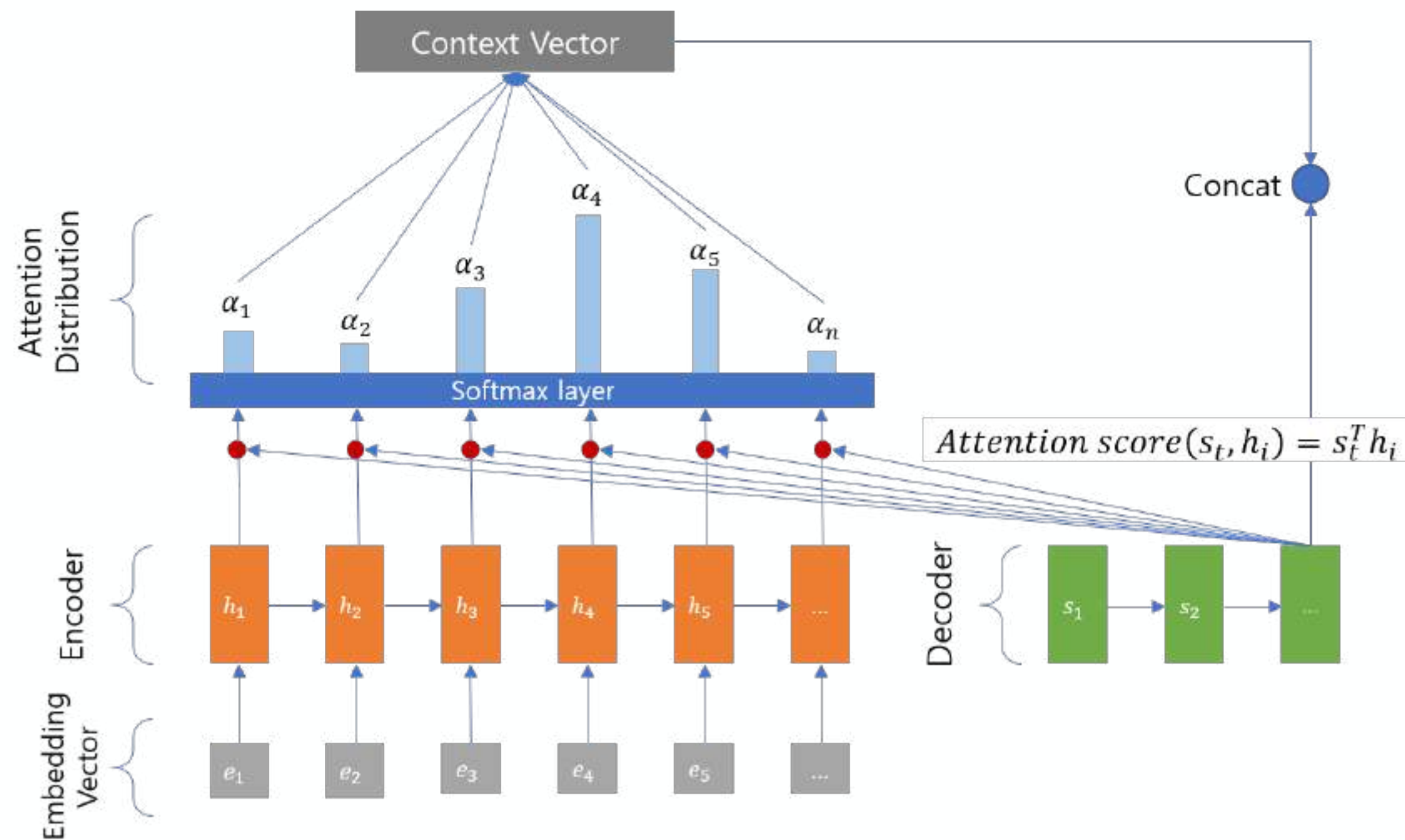
She is eating a green apple.

high attention

low attention

Attention - cont.

Neural Machine Translation (NMT)



https://en.wikipedia.org/wiki/Seq2seq#/media/File:Seq2seq_with_RNN_and_attention_mechanism.gif

Bahdanau, D. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

Transformer

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

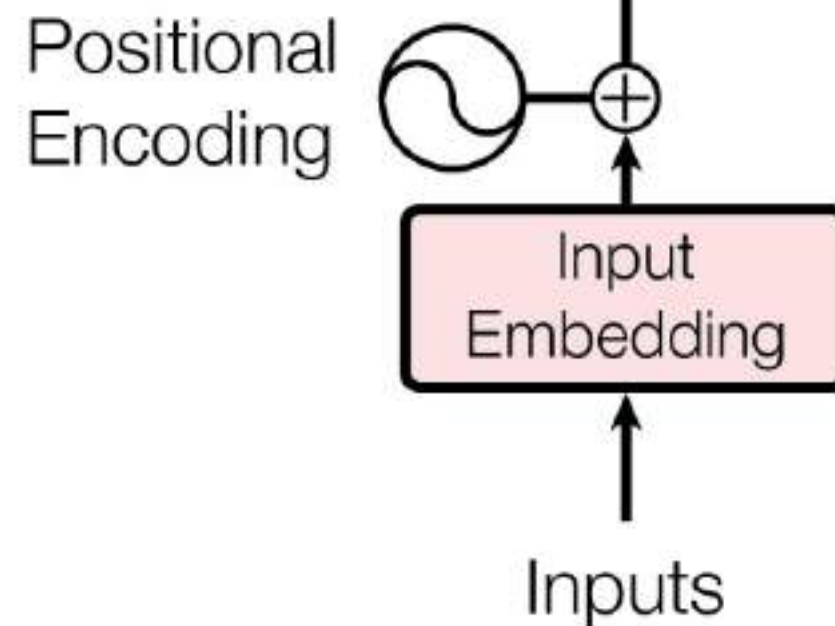
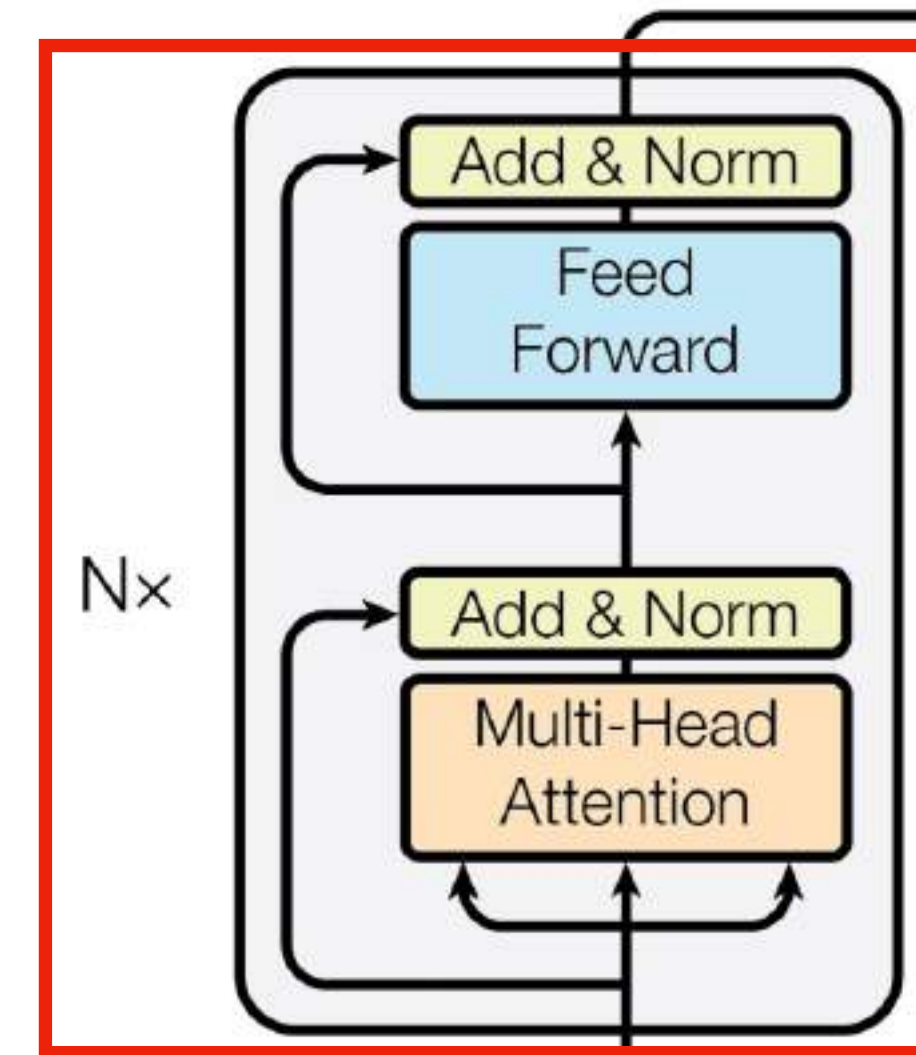
Łukasz Kaiser*
Google Brain
lukaszkaizer@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

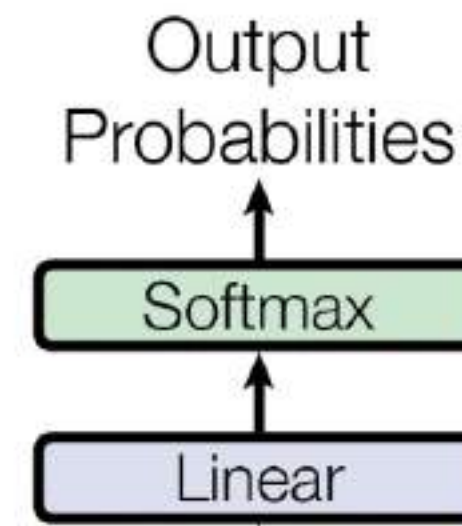
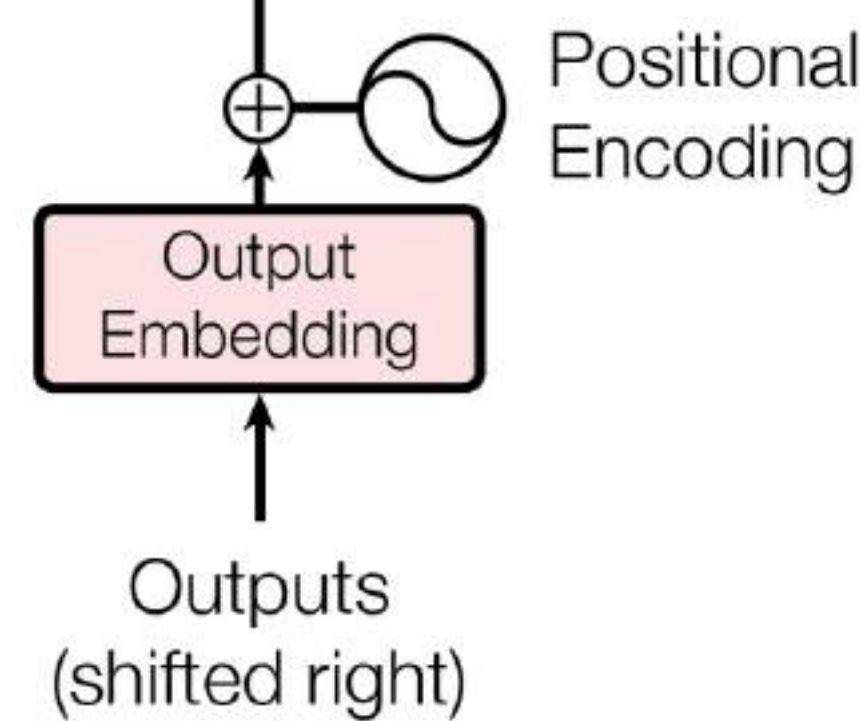
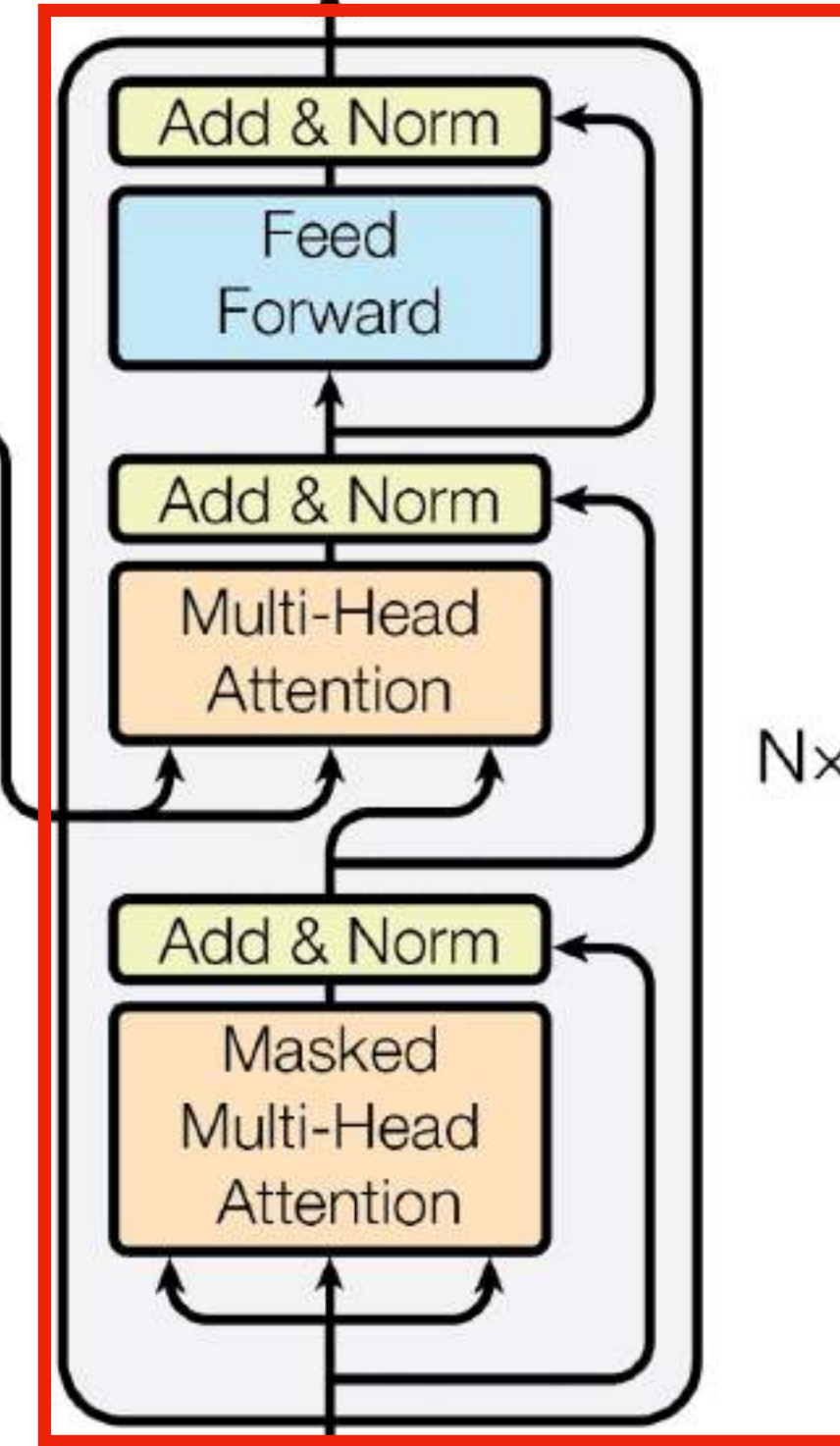
Citation: 142473 (up to Nov. 25, 2024)

Transformer Architecture

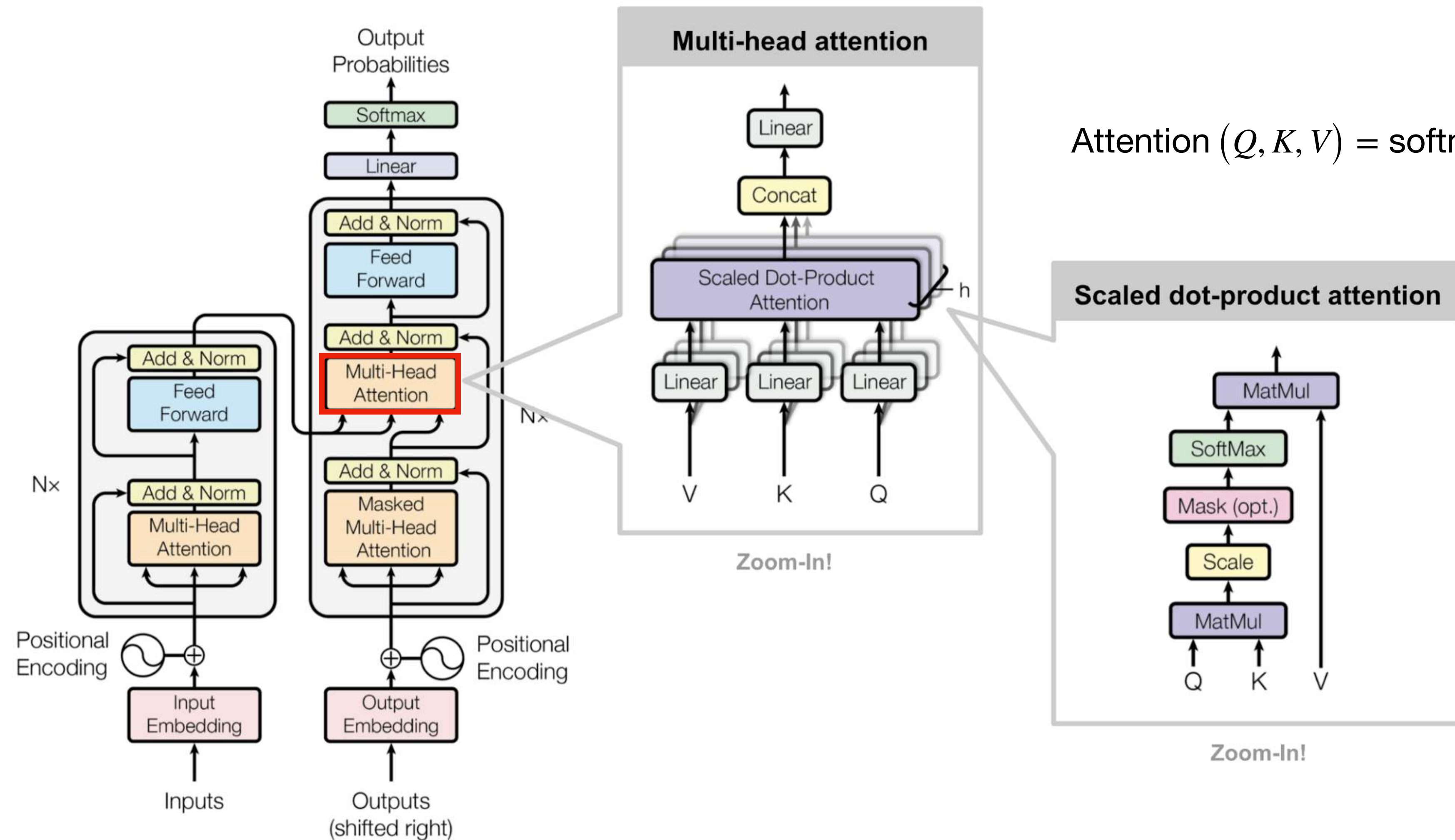
Encoder
(BERT)



Decoder
(GPT)



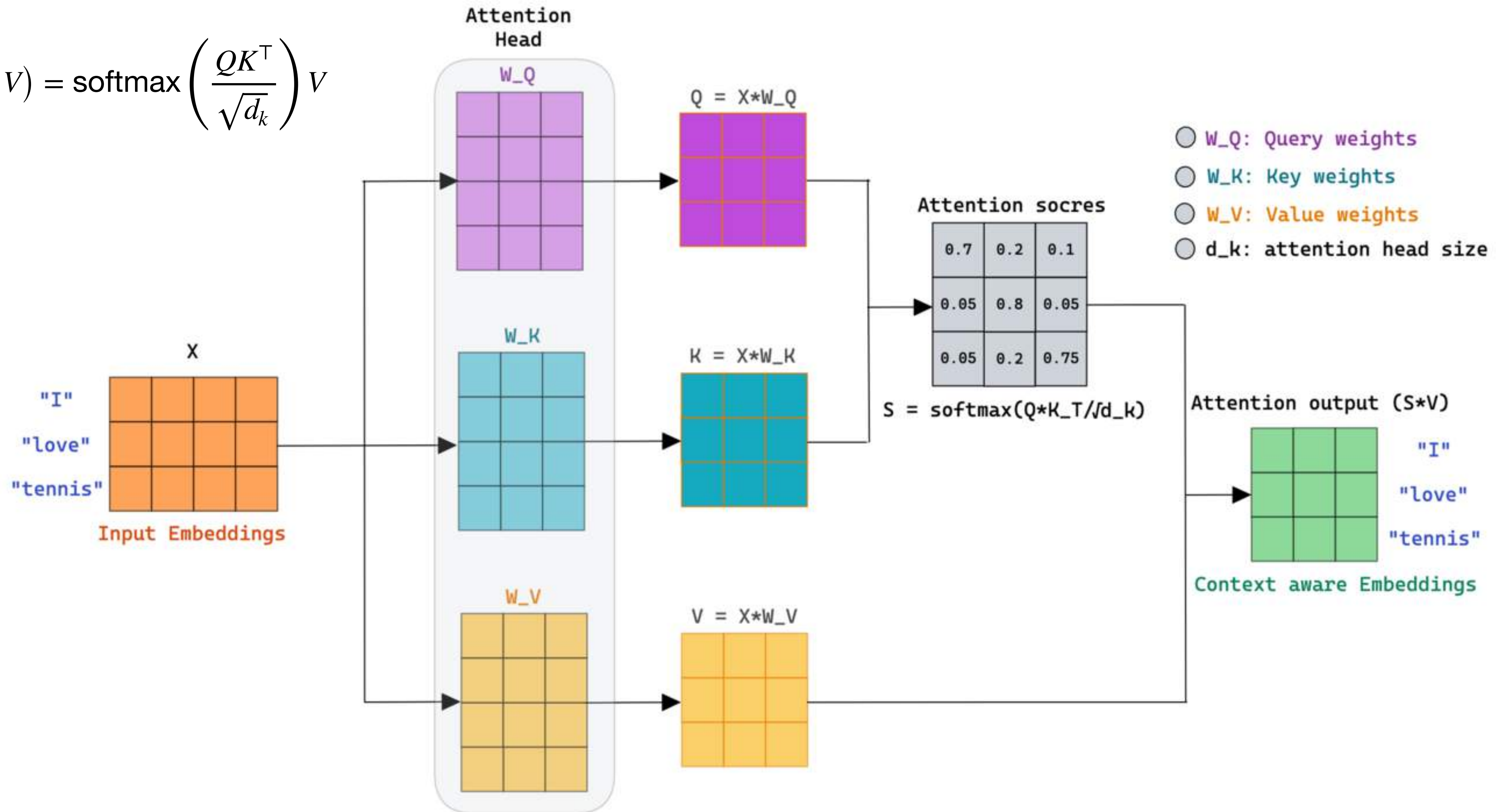
Self-Attention



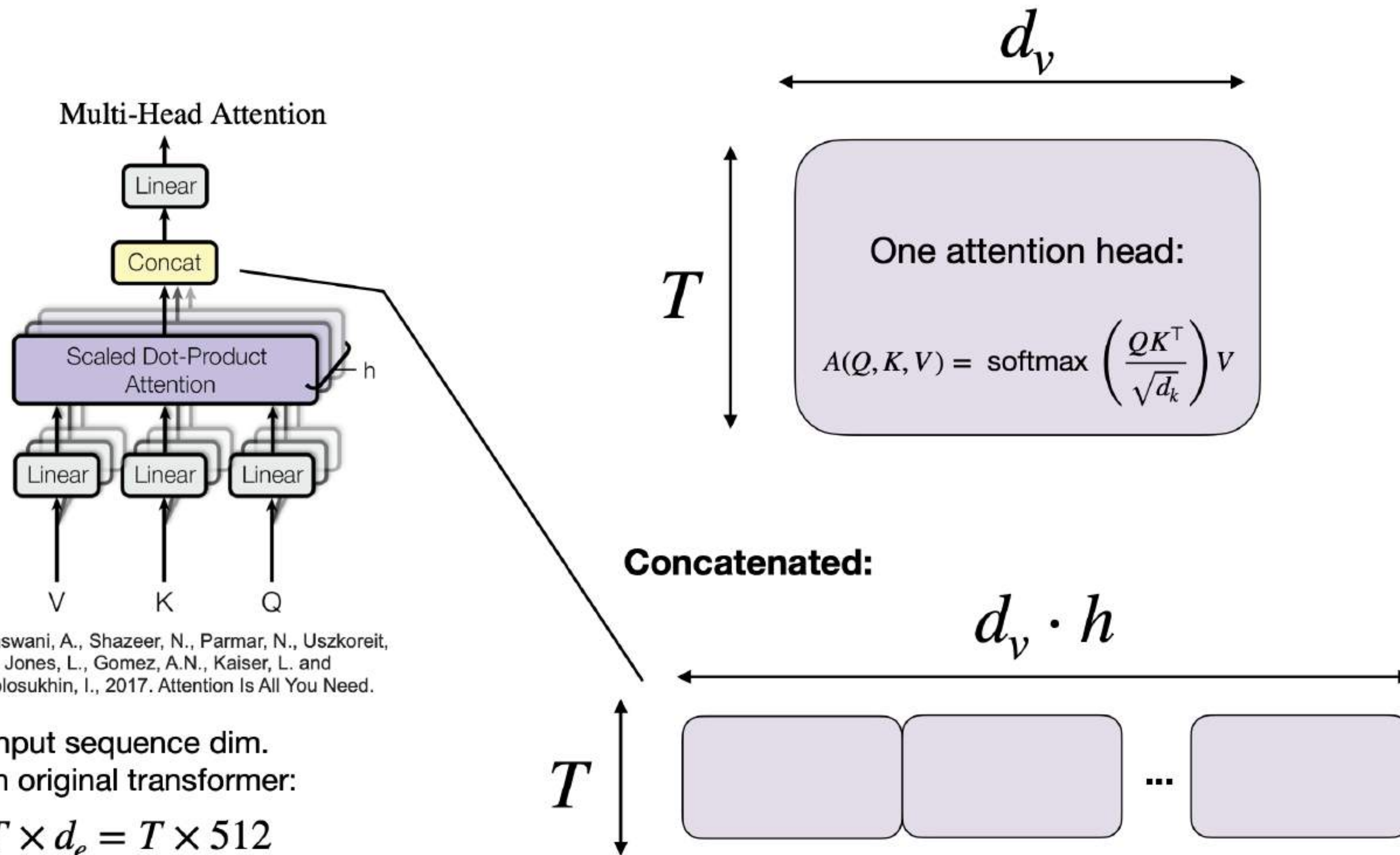
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Scaled Dot-Product Attention

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



Multi-Head Attention



Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., 2017. Attention Is All You Need.

Input sequence dim. in original transformer:

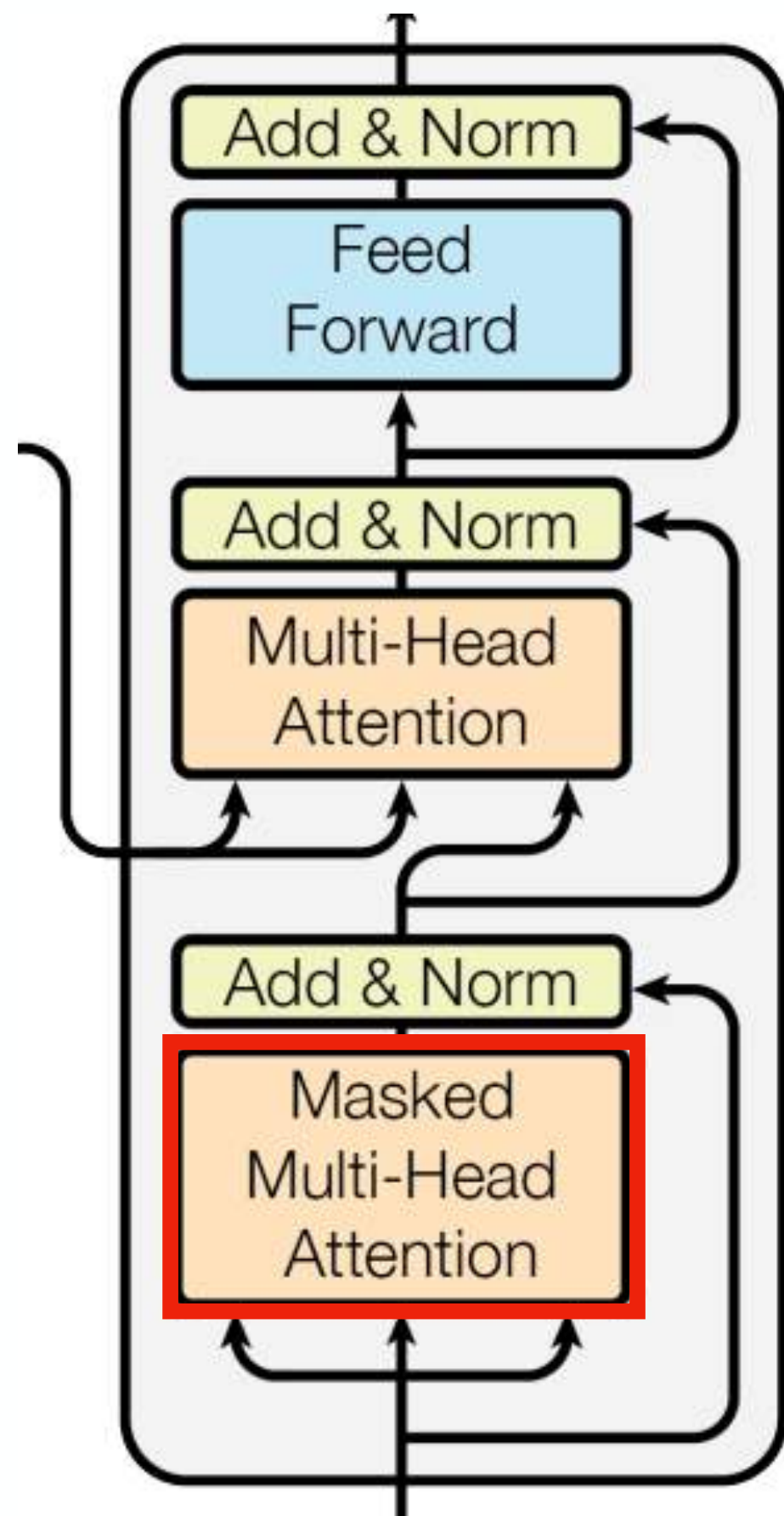
$$T \times d_e = T \times 512$$

and

$$d_v = 512/h = 64$$

Masked Multi-Head Attention

- You are not the prophet



Nx

Softmax

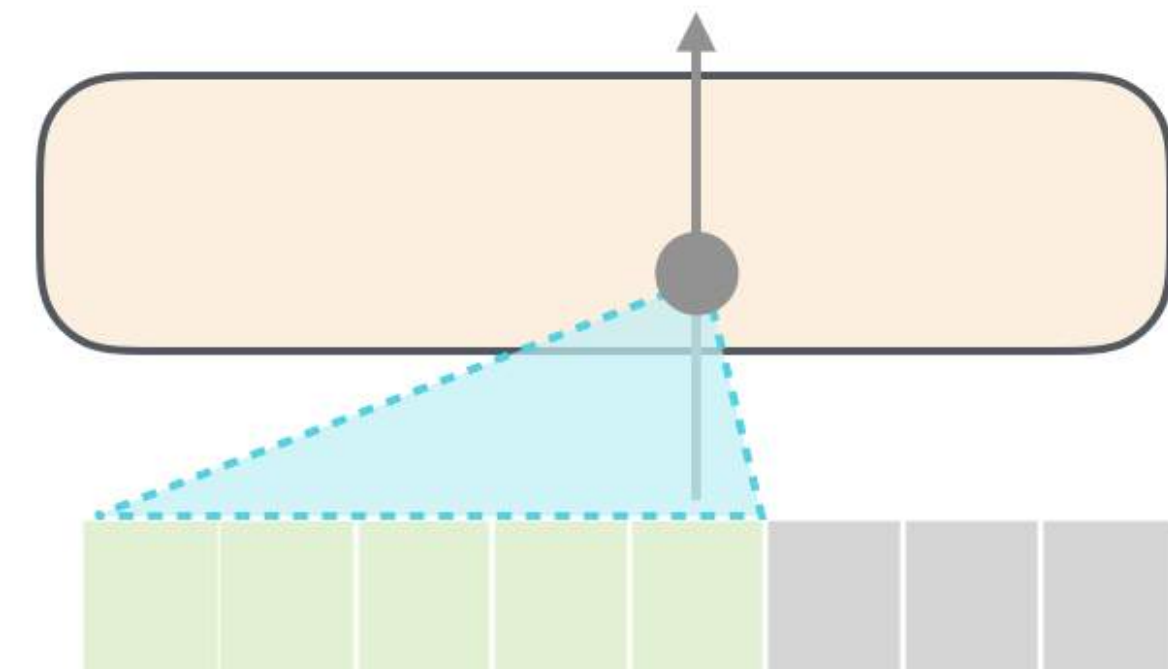
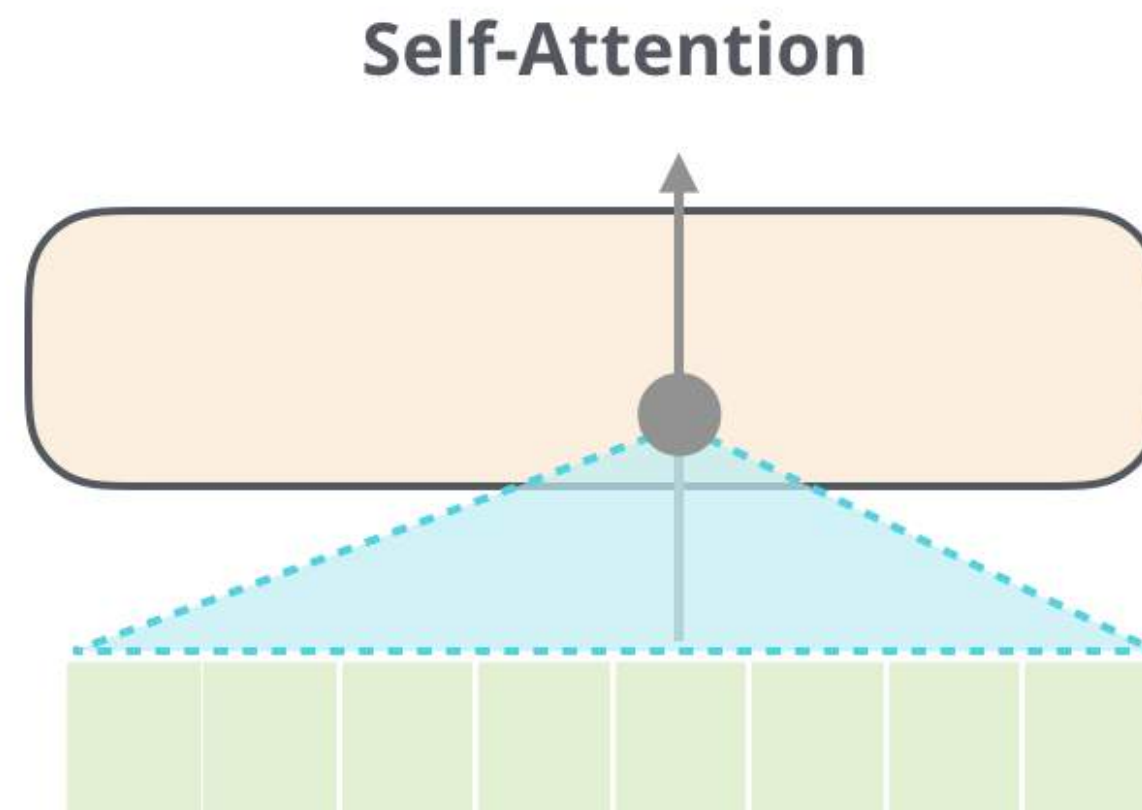
$$\frac{\text{Query Matrix} \times \text{Transposed Key Matrix}}{\sqrt{d}} = \begin{bmatrix} \text{Unnormalized Attention Scores} & \text{Masked Attention Scores} \end{bmatrix}$$

Unnormalized Attention Scores				
1.1	2.3	5.3	2.1	-1.2
-0.1	0.5	1.3	-0.1	6.1
2.3	0.2	3.3	-1.0	-0.4
0.3	1.2	-0.3	5.0	1.4
5.1	-2.9	-1.1	-4.2	0.4

Masked Attention Scores				
1.1	-∞	-∞	-∞	-∞
-0.1	0.5	-∞	-∞	-∞
2.3	0.2	3.3	-∞	-∞
0.3	1.2	-0.3	5.0	-∞
5.1	-2.9	-1.1	-4.2	0.4

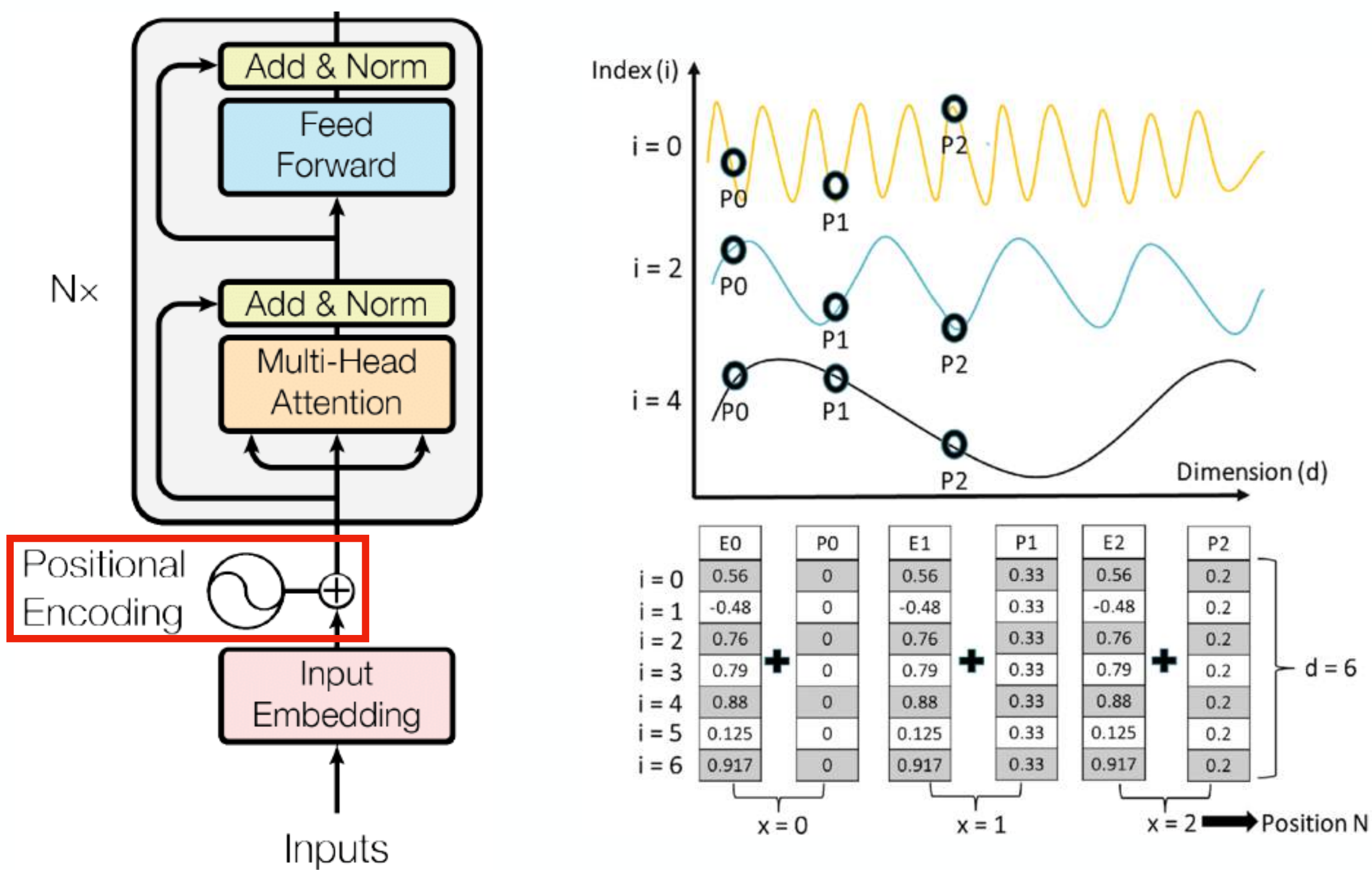
(Causal Attention)

Masked Self-Attention



Positional Encoding

- It's All About Positioning



Formulation for Positional Encoding

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

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

$\frac{e_1 + n}{10000^{2/5}}$	$\frac{e_2 + n}{10000^{4/5}}$	$\frac{e_3 + n}{10000^{6/5}}$	$\frac{e_4 + n}{10000^{8/5}}$	$\frac{e_5 + n}{10000^{10/5}}$
-------------------------------	-------------------------------	-------------------------------	-------------------------------	--------------------------------

Input Encoding for Token n to the Transformer

© AIML.com Research

$\cos(\frac{n}{10000^{2/5}})$	$\sin(\frac{n}{10000^{4/5}})$	$\cos(\frac{n}{10000^{6/5}})$	$\sin(\frac{n}{10000^{8/5}})$	$\cos(\frac{n}{10000^{10/5}}$
$i=1$	$i=2$	$i=3$	$i=4$	$i=5$

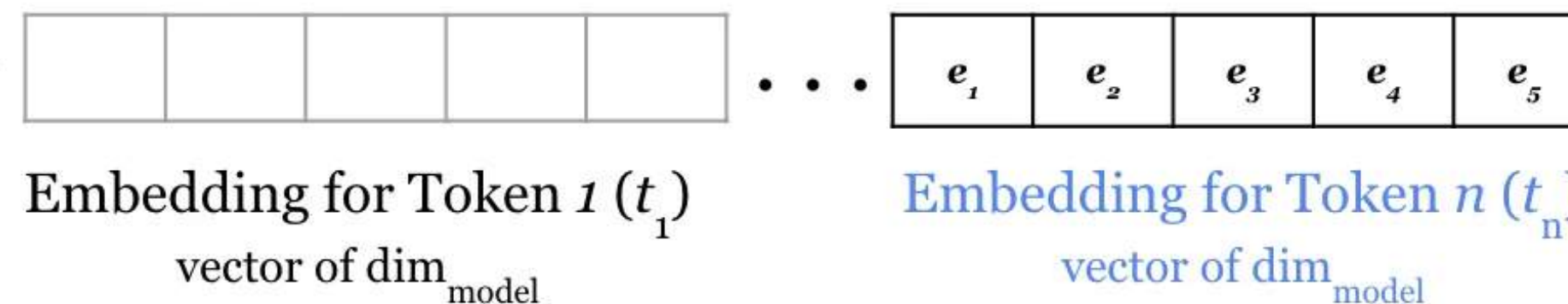
Positional Encoding for Token at position n

where, $pos=n, d_{model}=5$

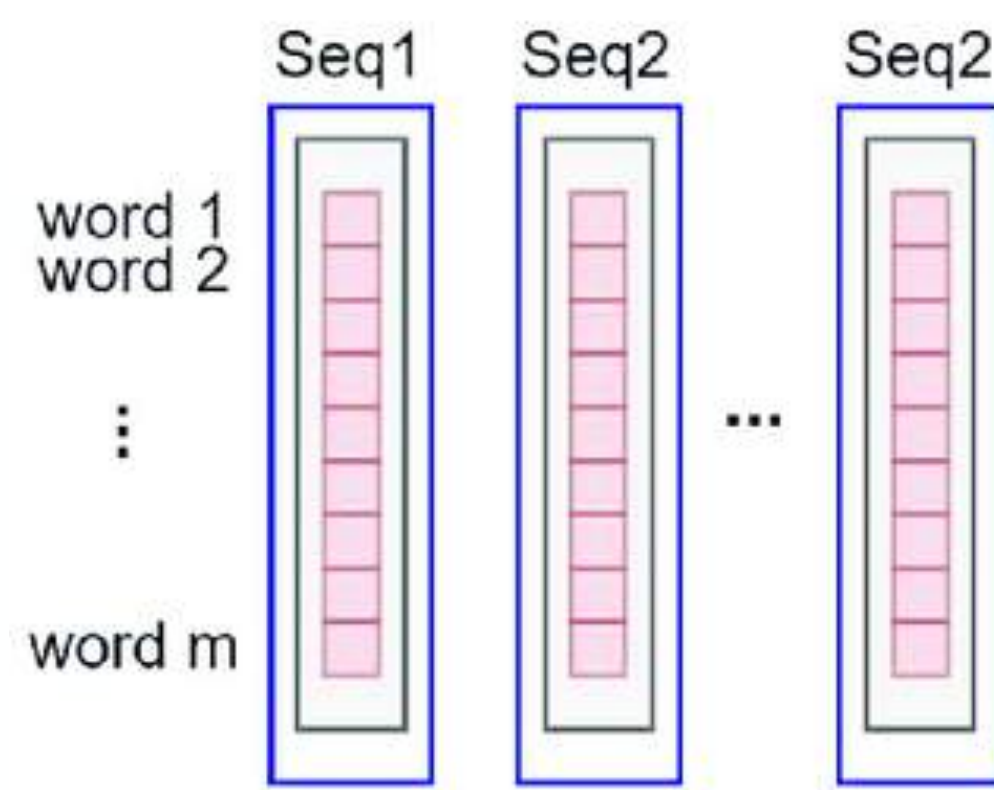
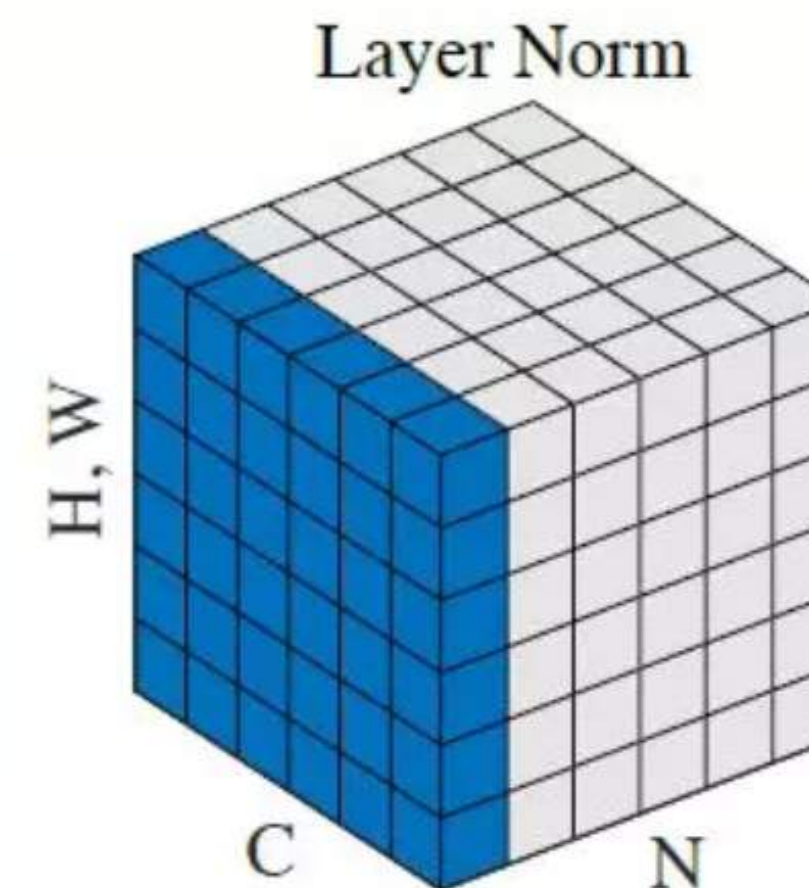
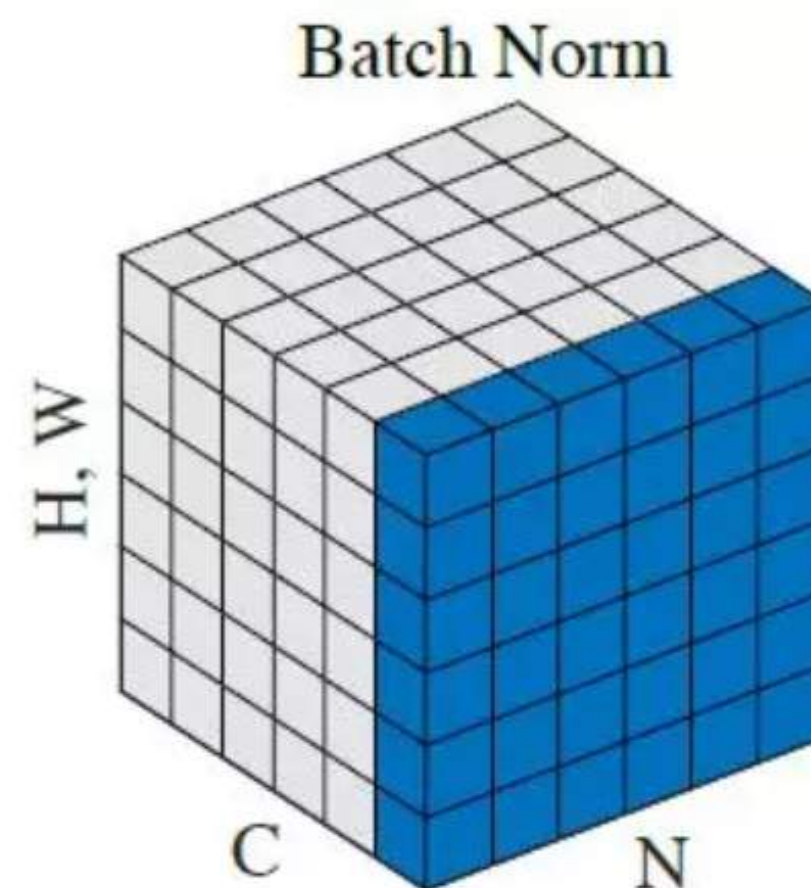
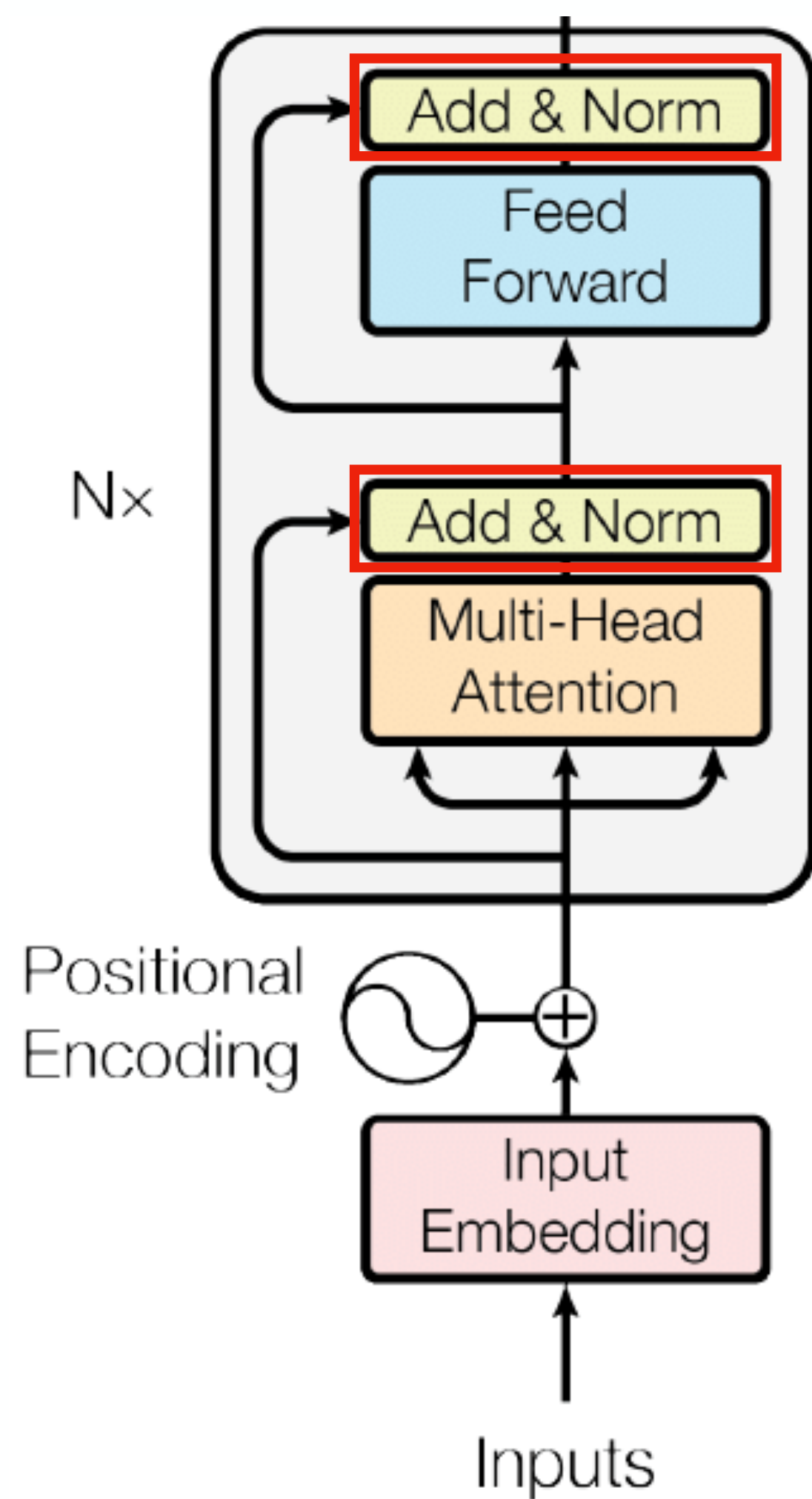


element-wise Addition

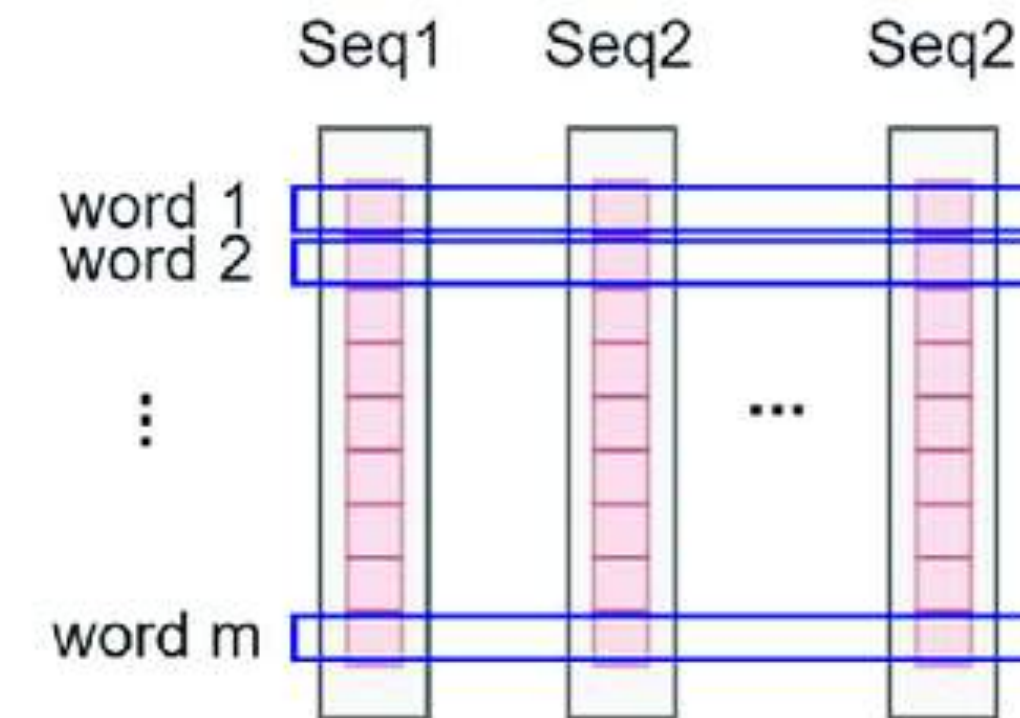
Input Token Embeddings



Layer Normalization



Layer Normalization (LN)



Batch Normalization (BN)

BERT

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

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova

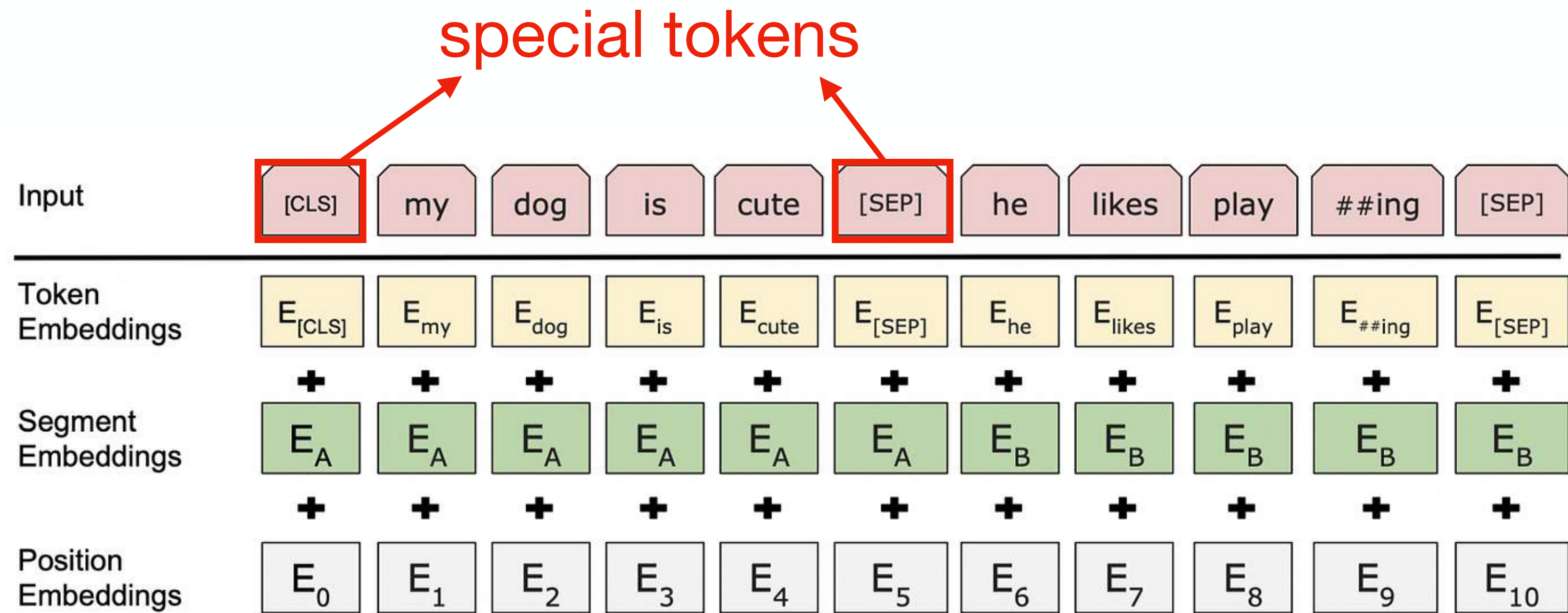
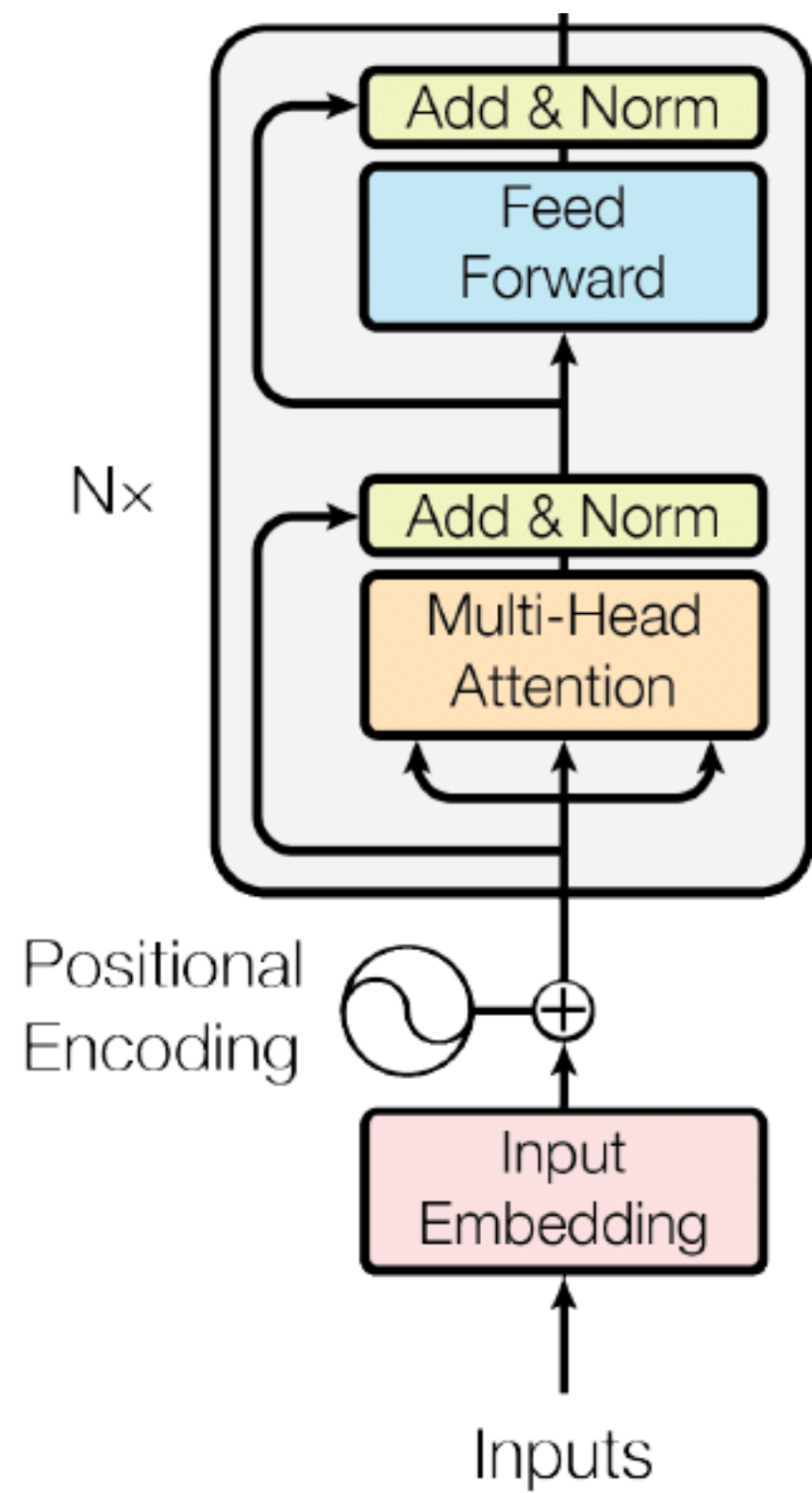
Google AI Language

`{jacobdevlin, mingweichang, kentonl, kristout}@google.com`

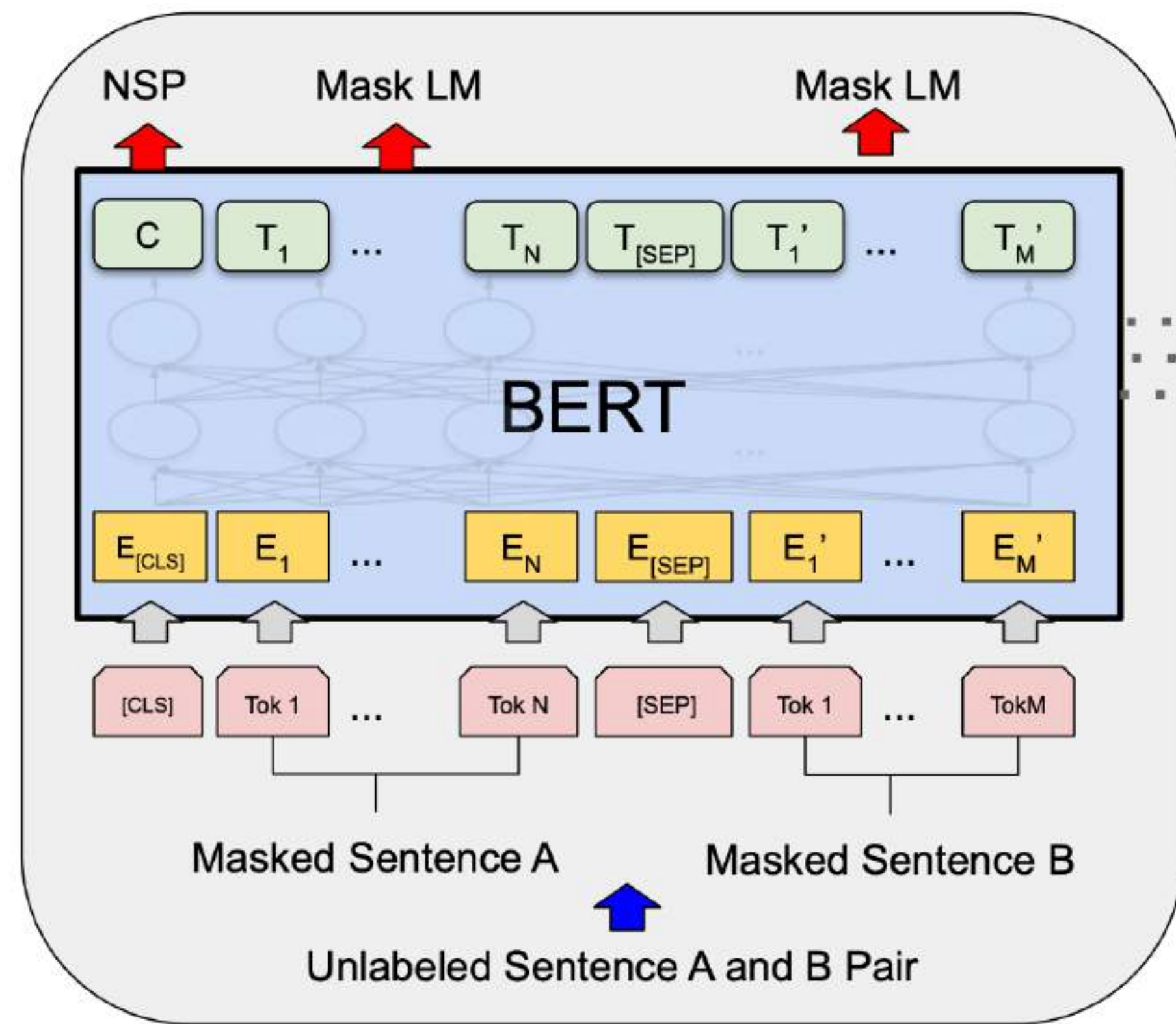
Citation: 118772 (up to Nov. 25, 2024)

BERT

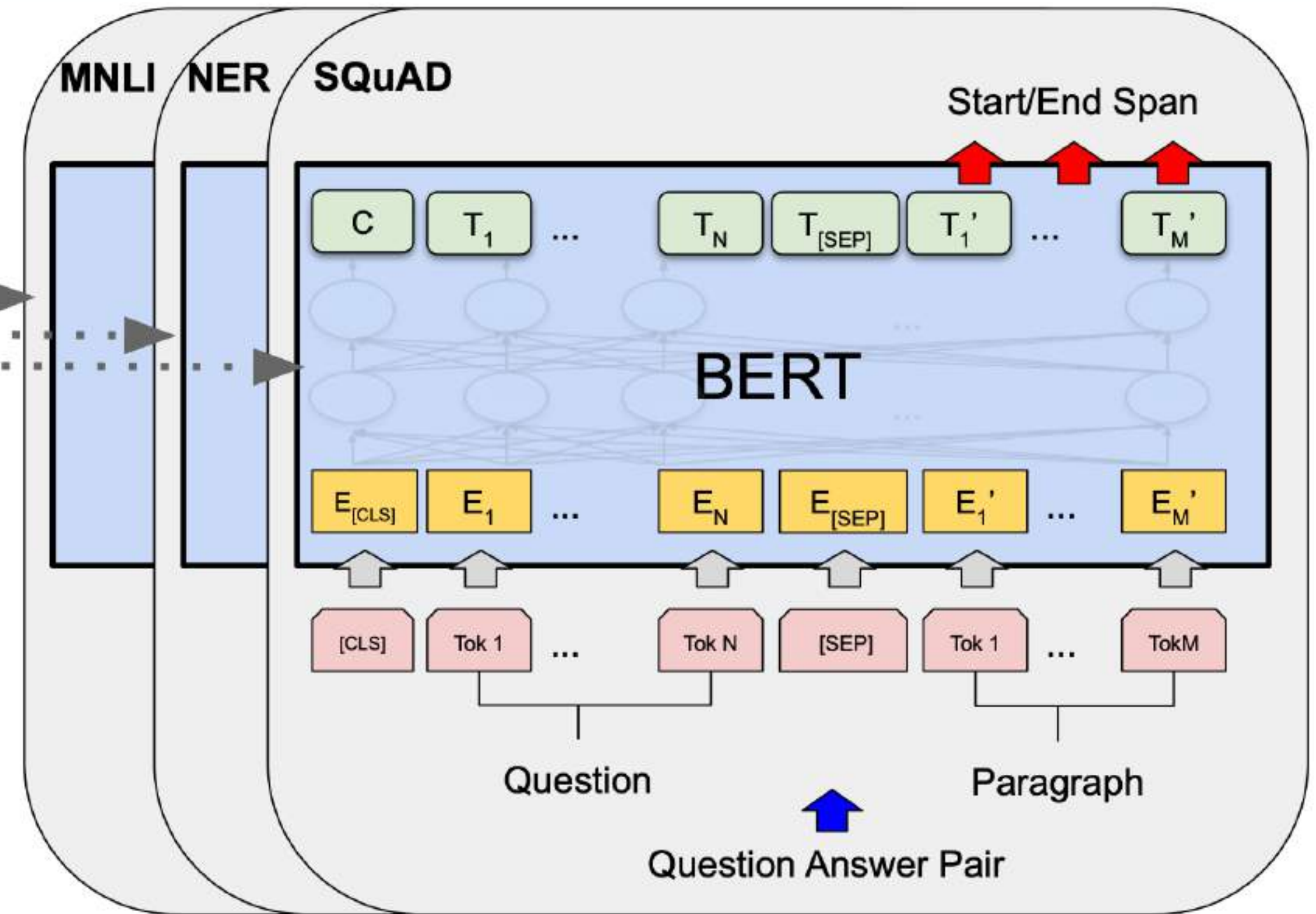
- Bidirectional Encoder Representations from Transformer



Two-Stage Training



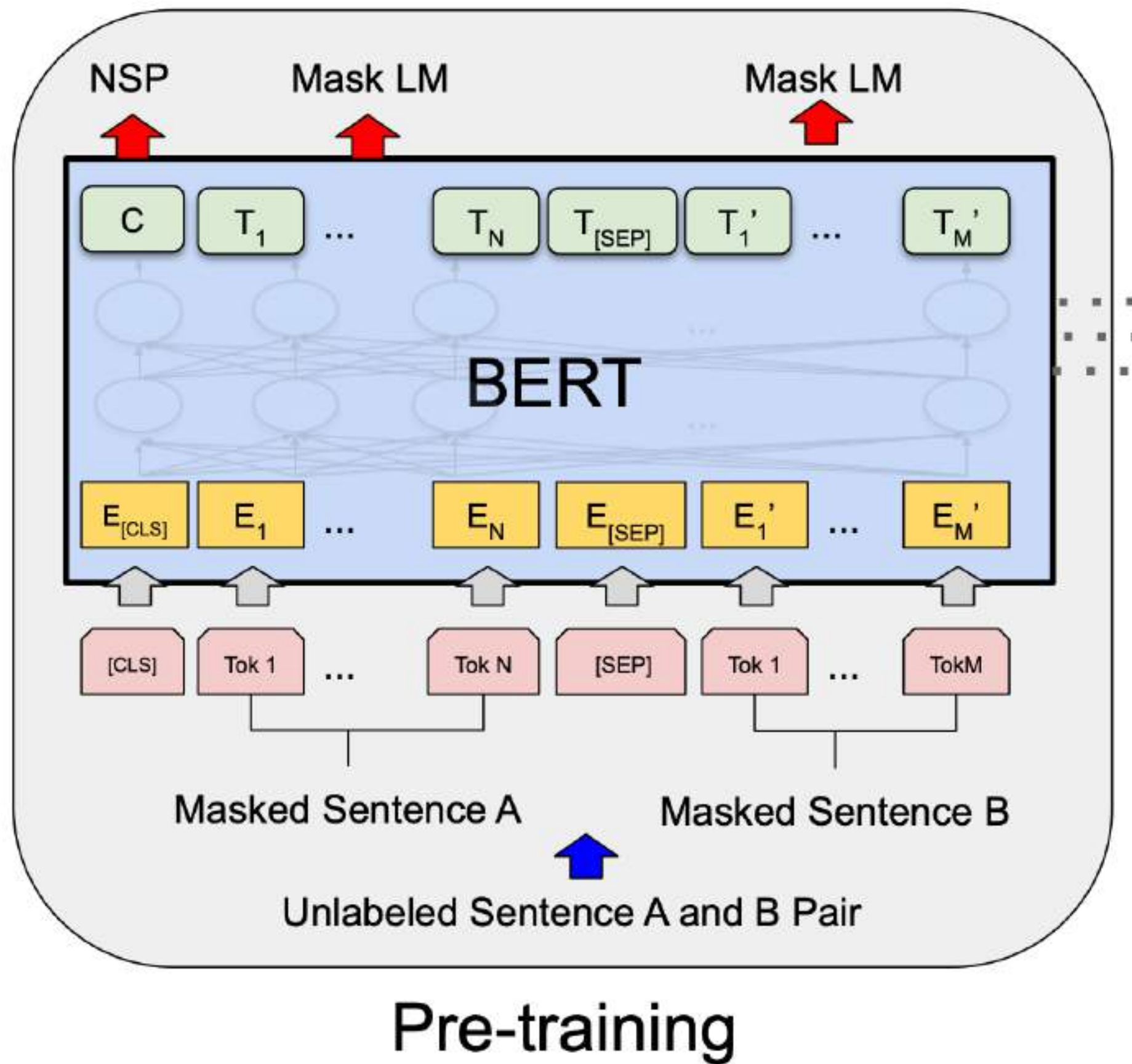
Pre-training



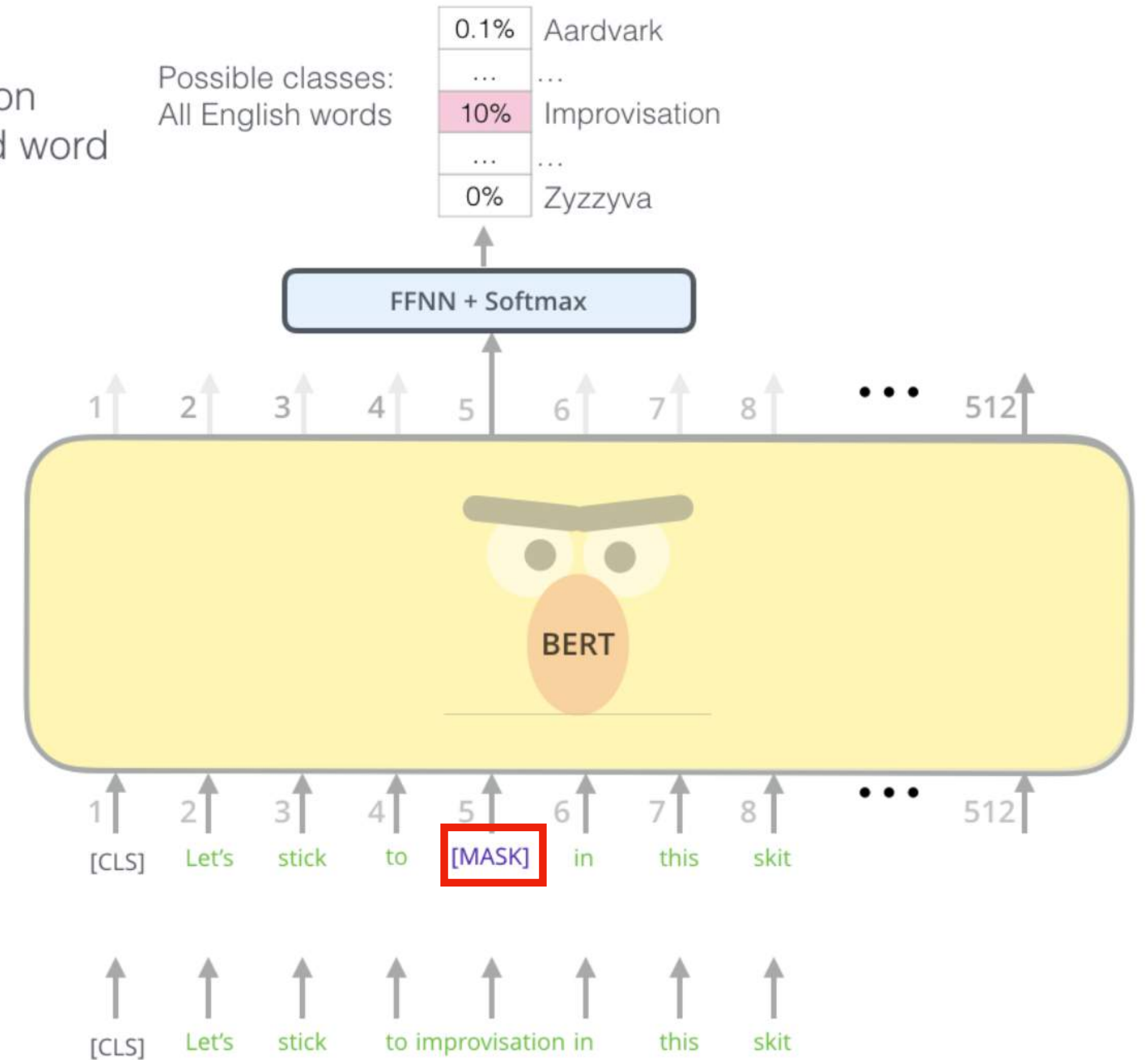
Fine-Tuning

Pre-Training Stage

- Masked Language Model (MLM)



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



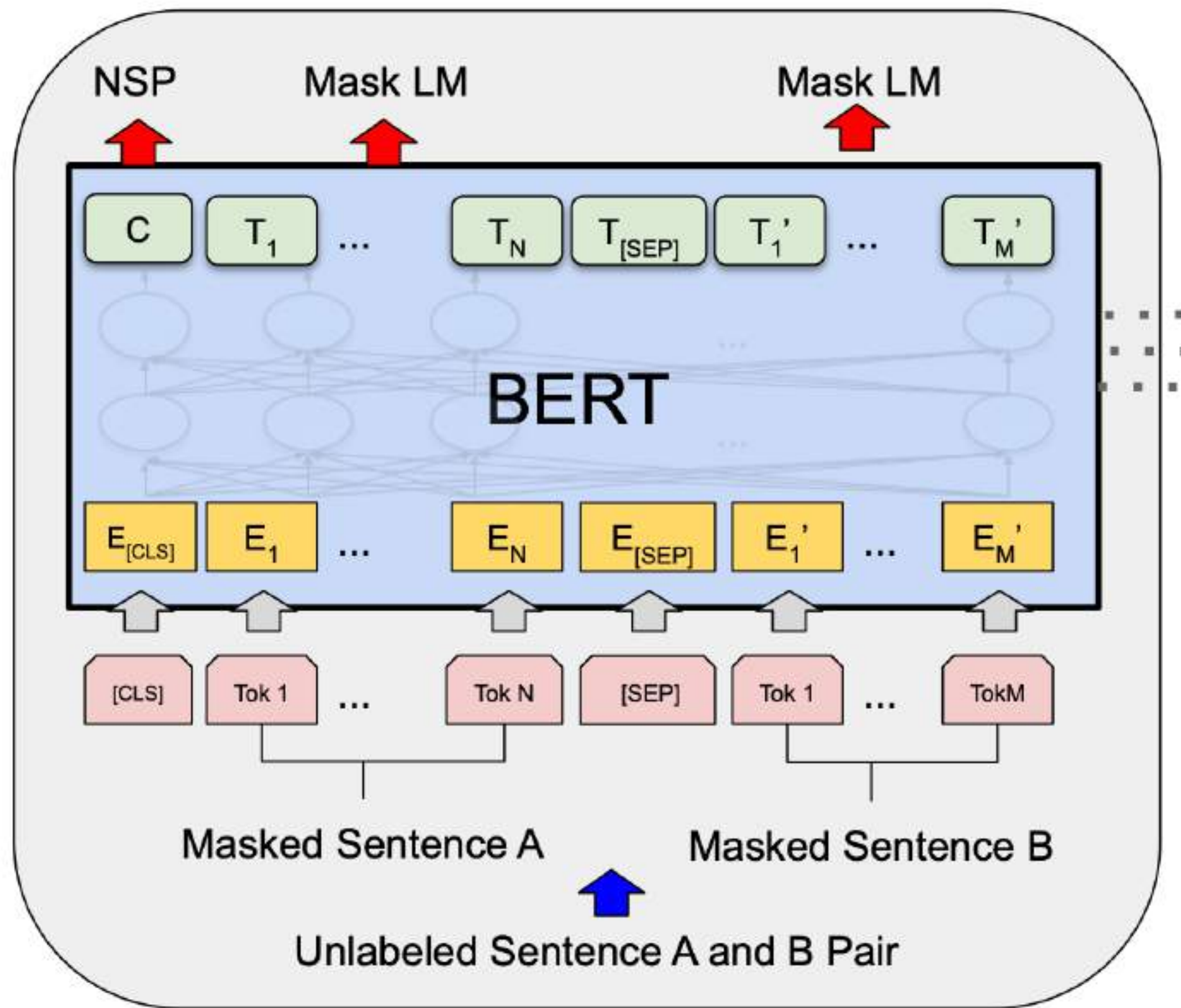
Randomly mask 15% of tokens

Input

BERT's clever language modeling task masks 15% of words in the input and asks the model to predict the missing word.

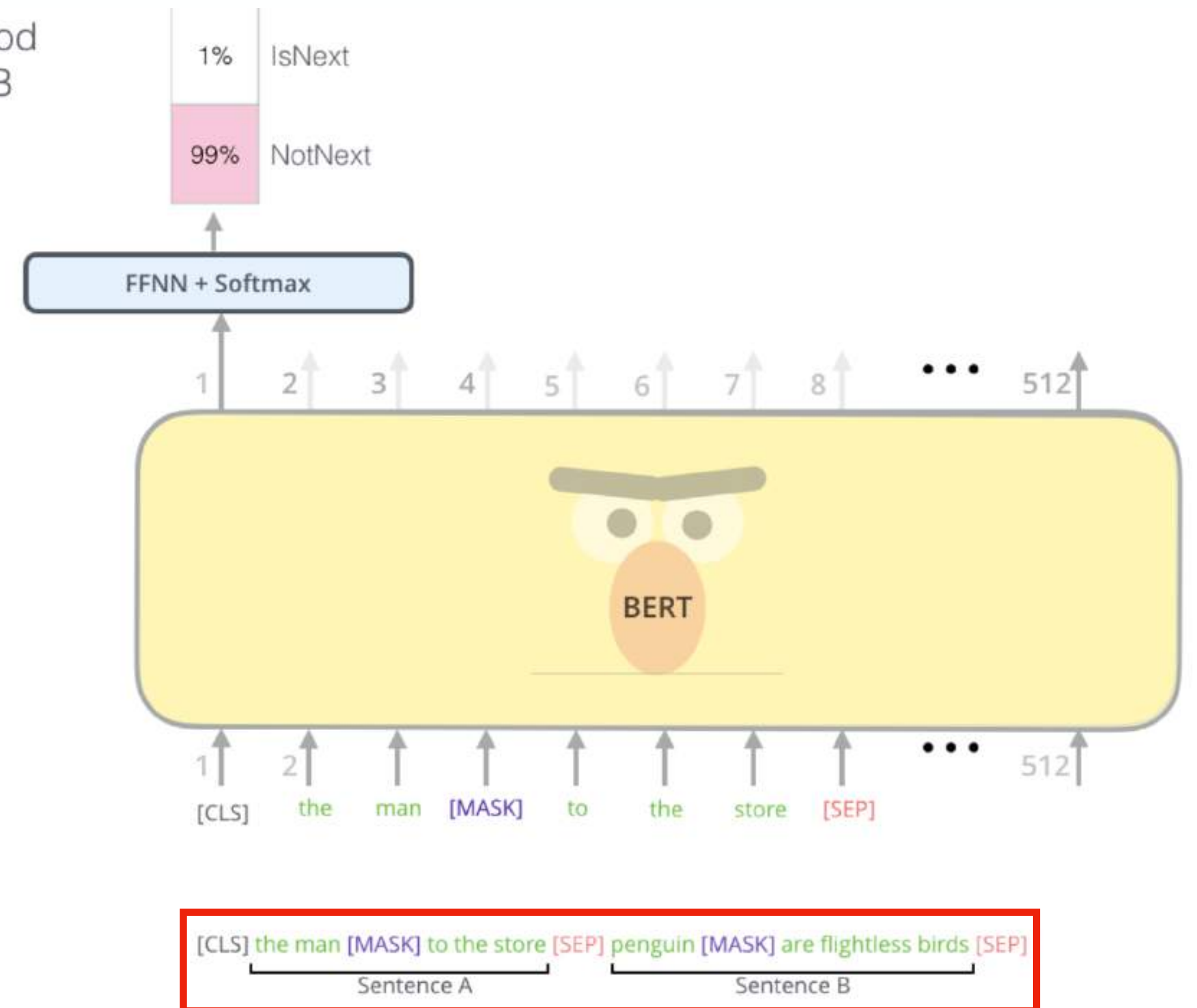
Pre-Training Stage

- Next Sentence Prediction (NSP)



Pre-training

Predict likelihood that sentence B belongs after sentence A



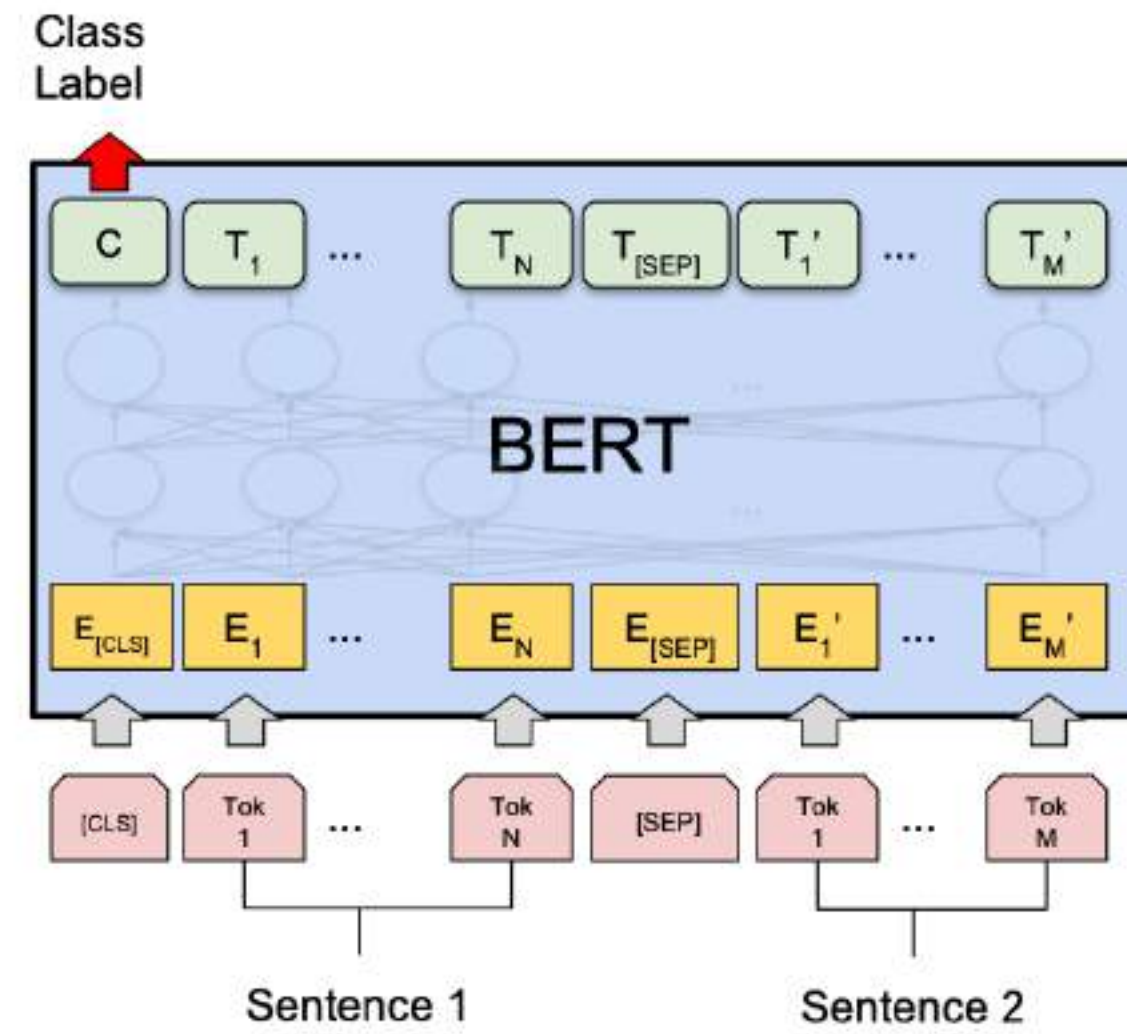
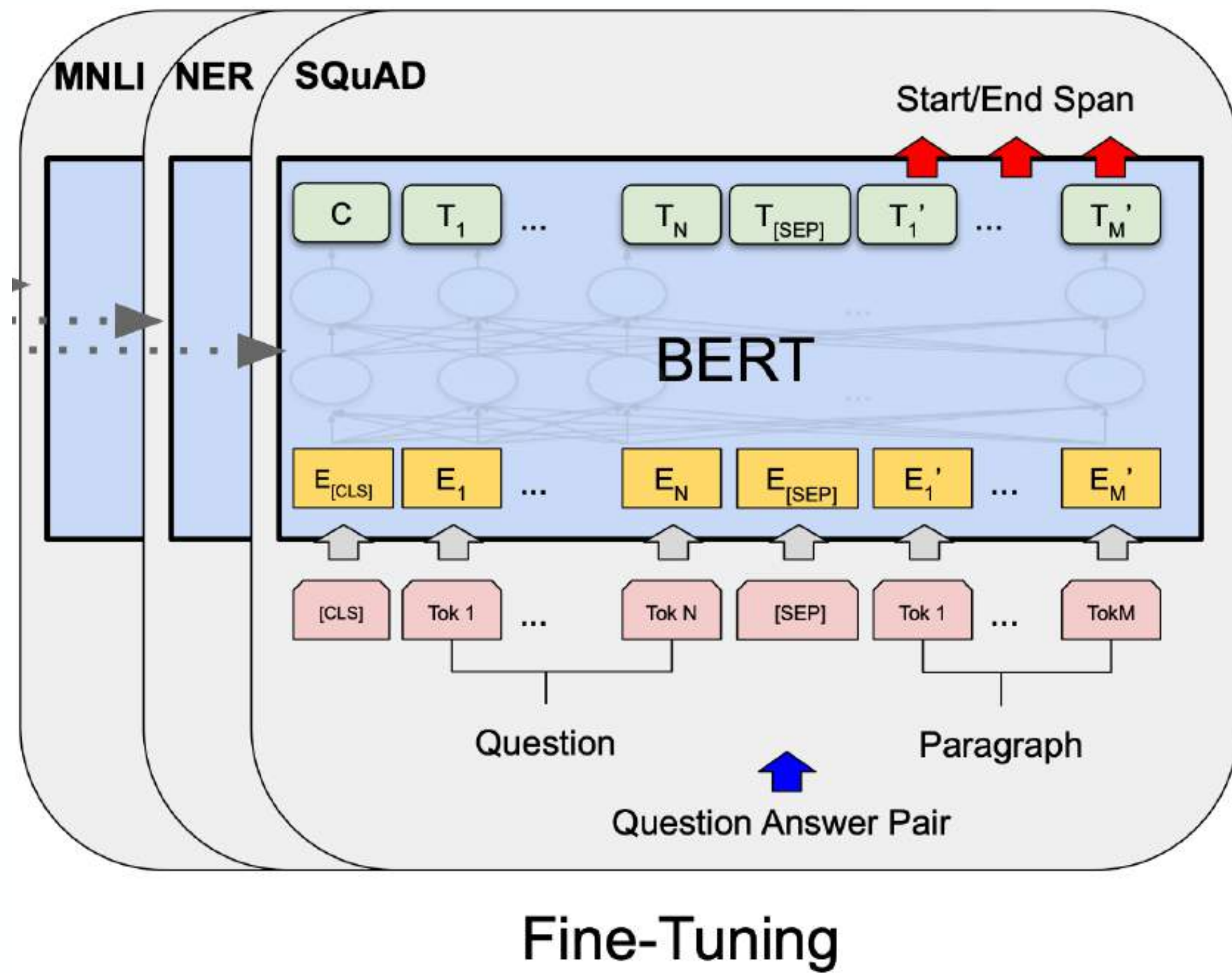
Tokenized Input

Input

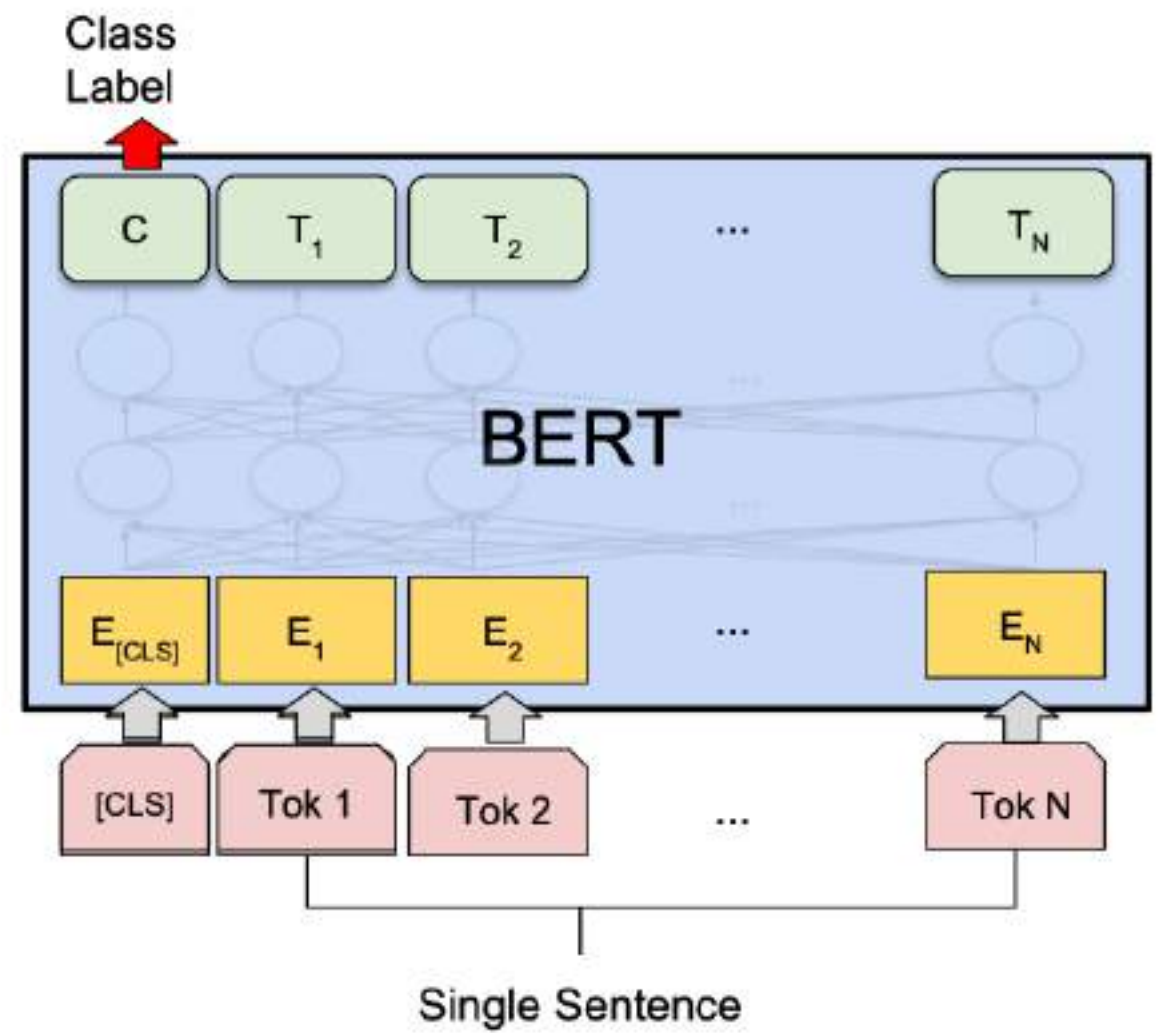
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]
Sentence A Sentence B

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.

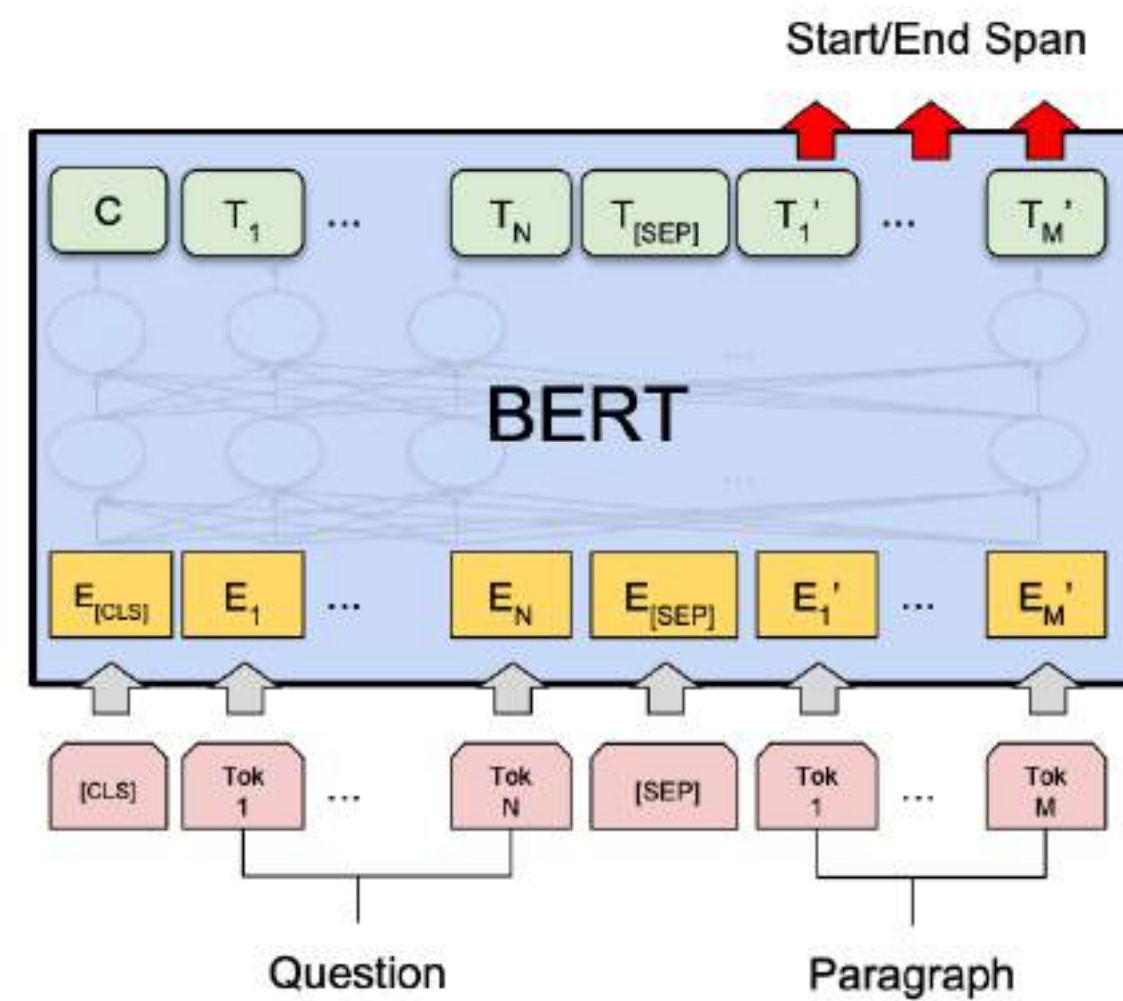
Fine-Tuning Stage



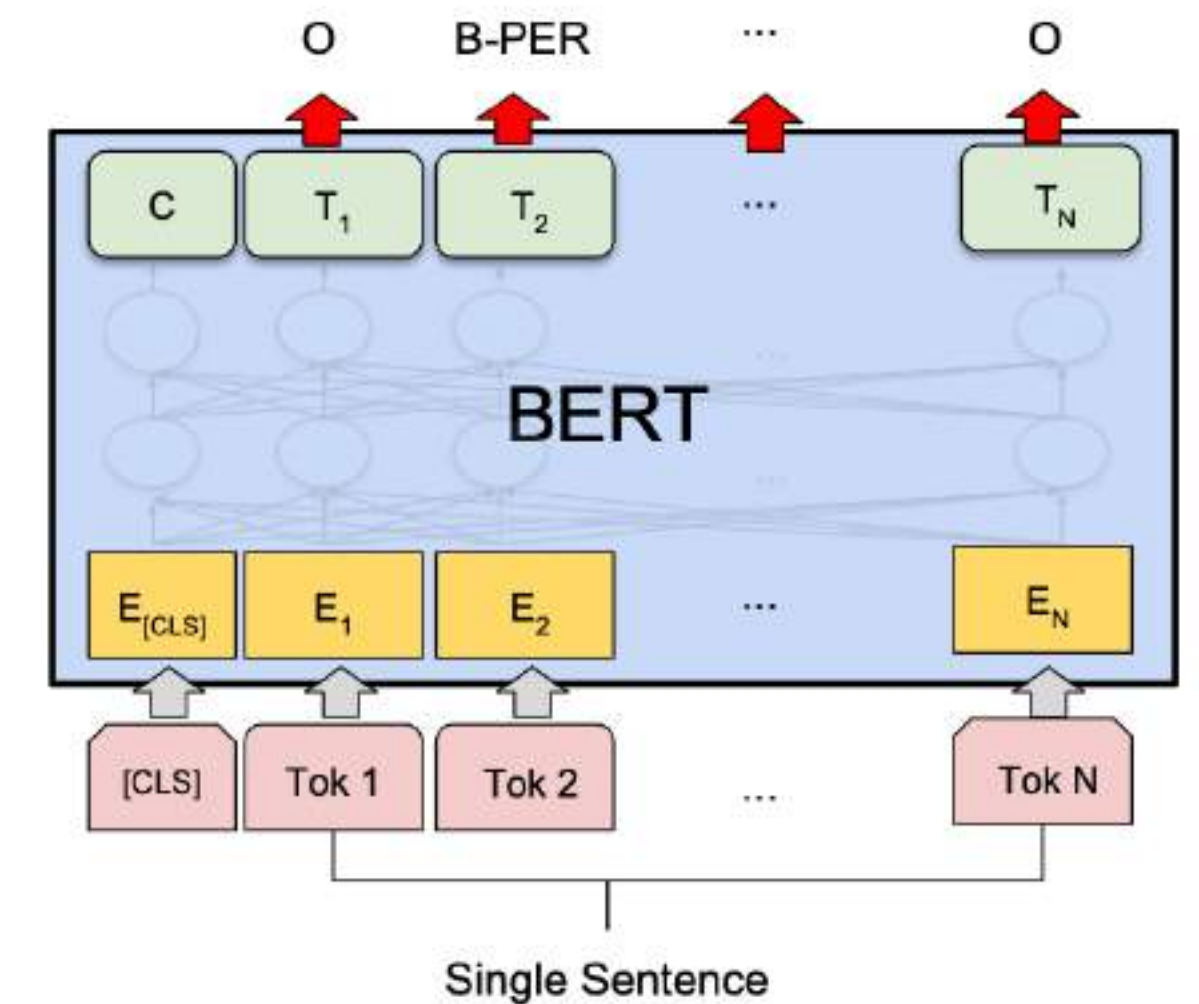
(a) Sentence Pair Classification Tasks:
MNL, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



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



(c) Question Answering Tasks:
SQuAD v1.1



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

GPT-1

Improving Language Understanding by Generative Pre-Training

Alec Radford
OpenAI
alec@openai.com

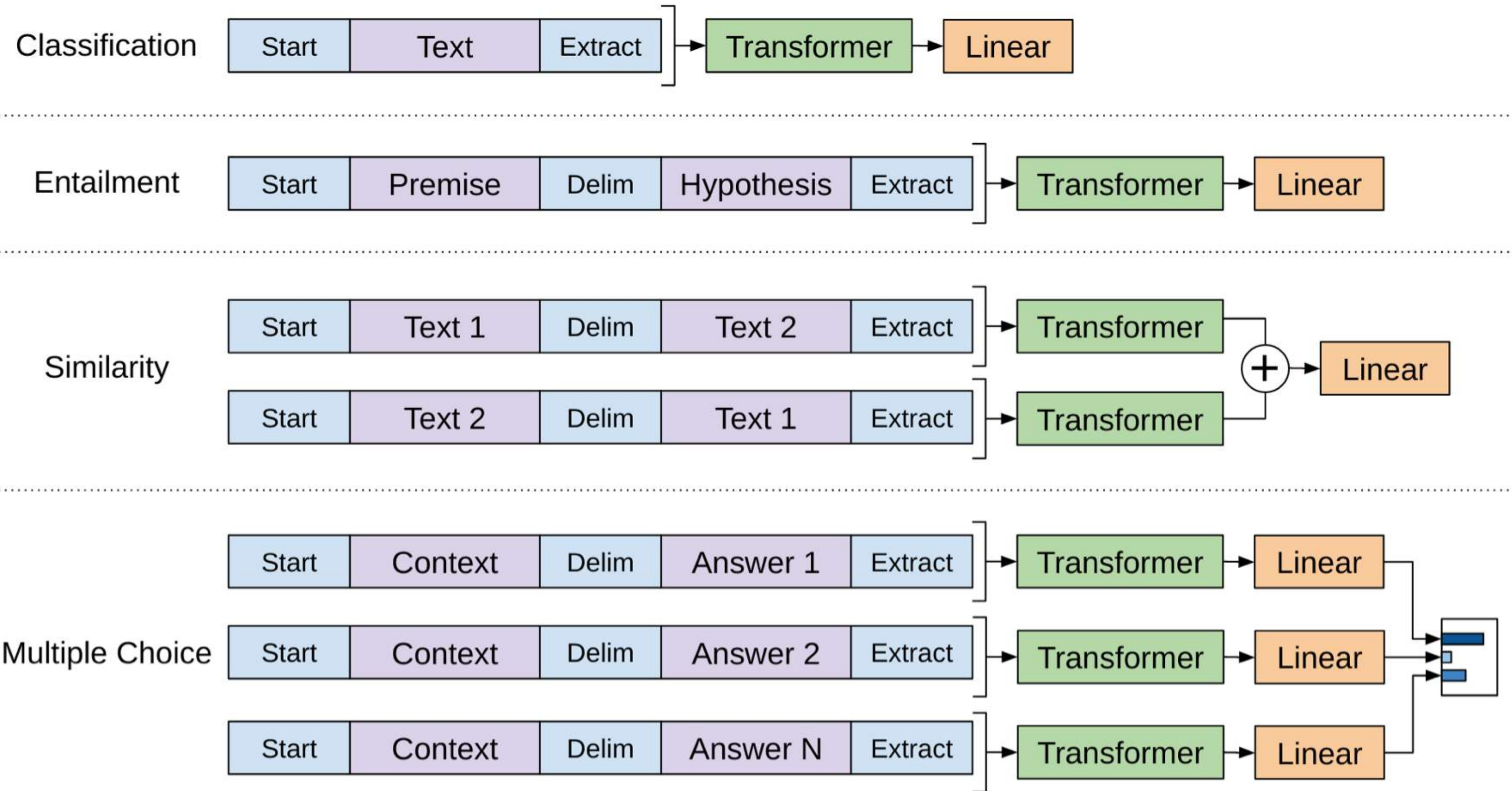
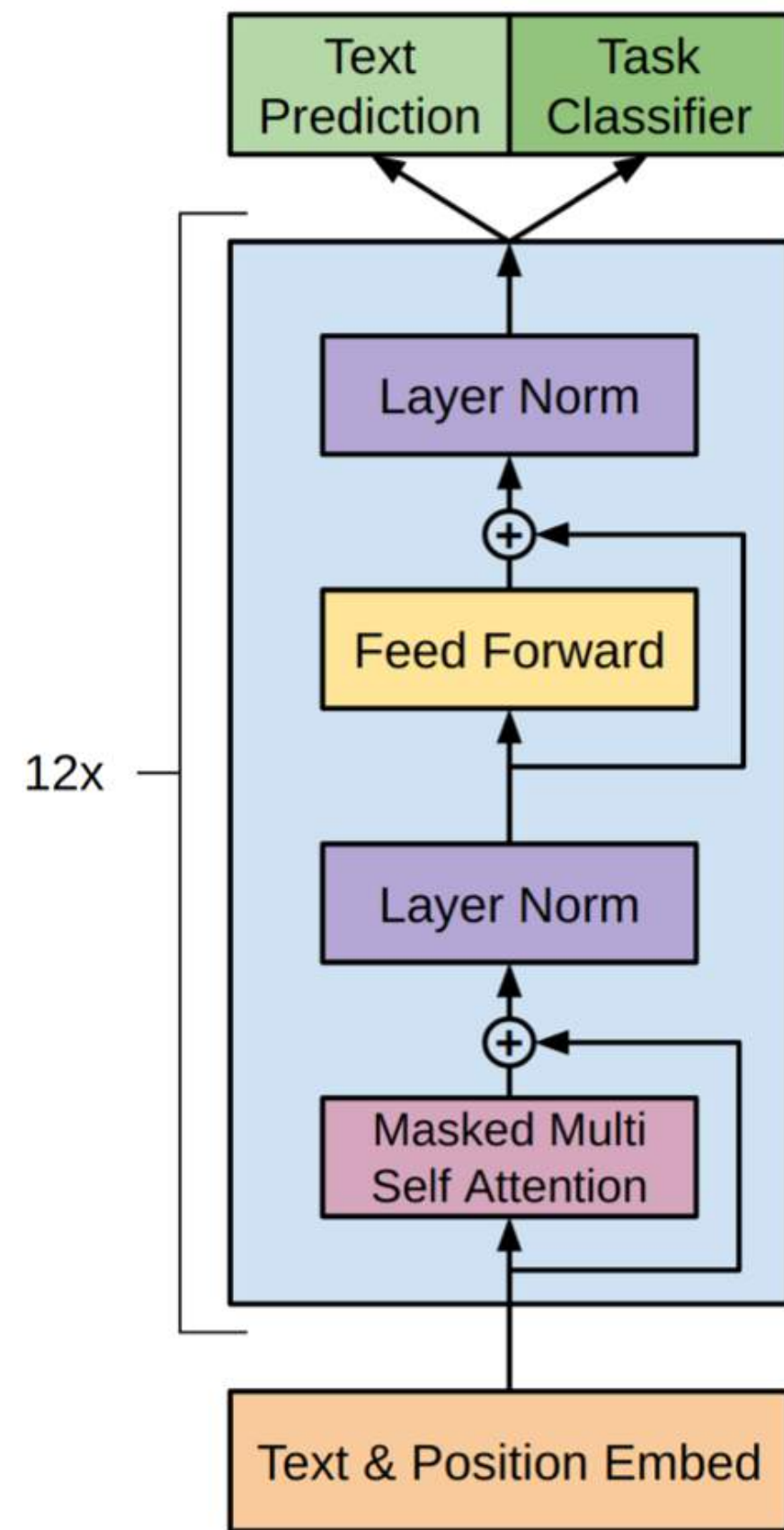
Karthik Narasimhan
OpenAI
karthikn@openai.com

Tim Salimans
OpenAI
tim@openai.com

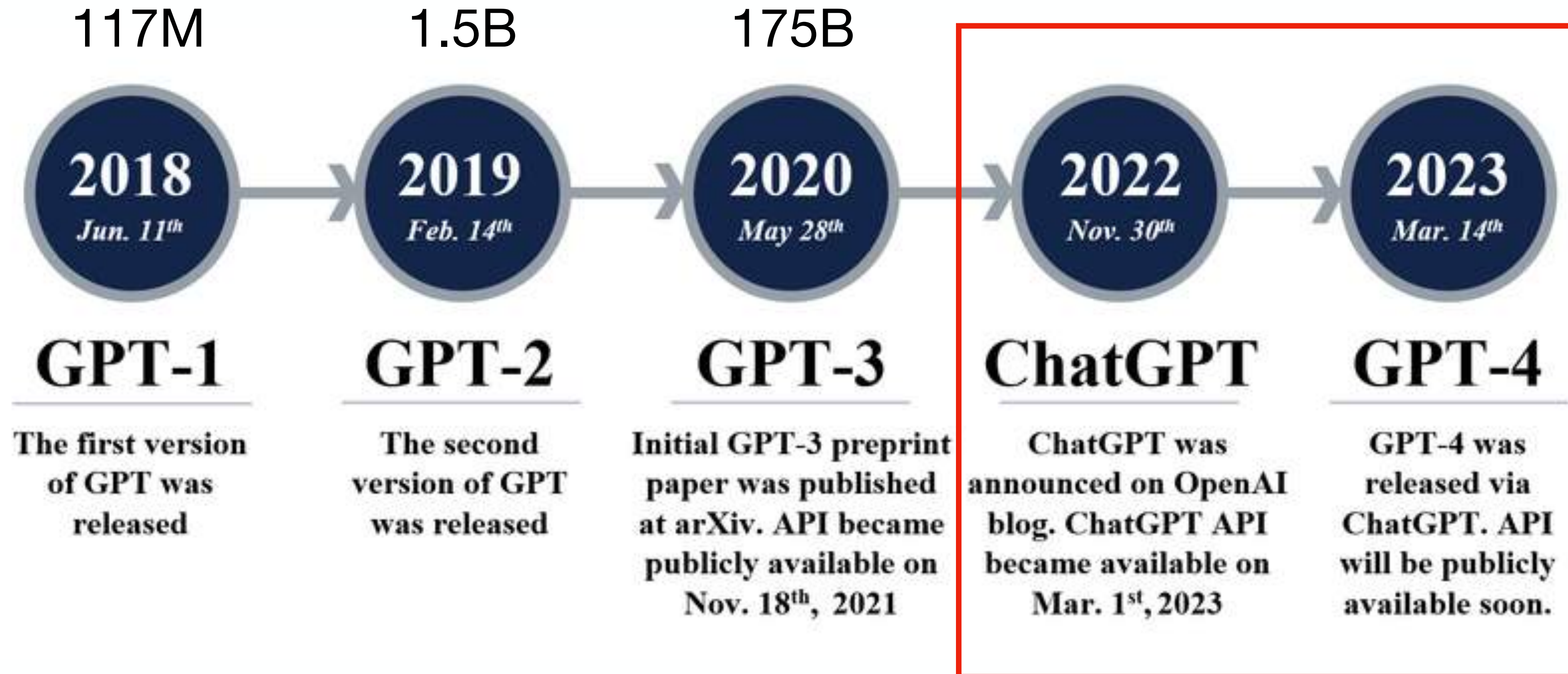
Ilya Sutskever
OpenAI
ilyasu@openai.com

Citation: 11689 (up to Nov. 25, 2024)

GPT-1

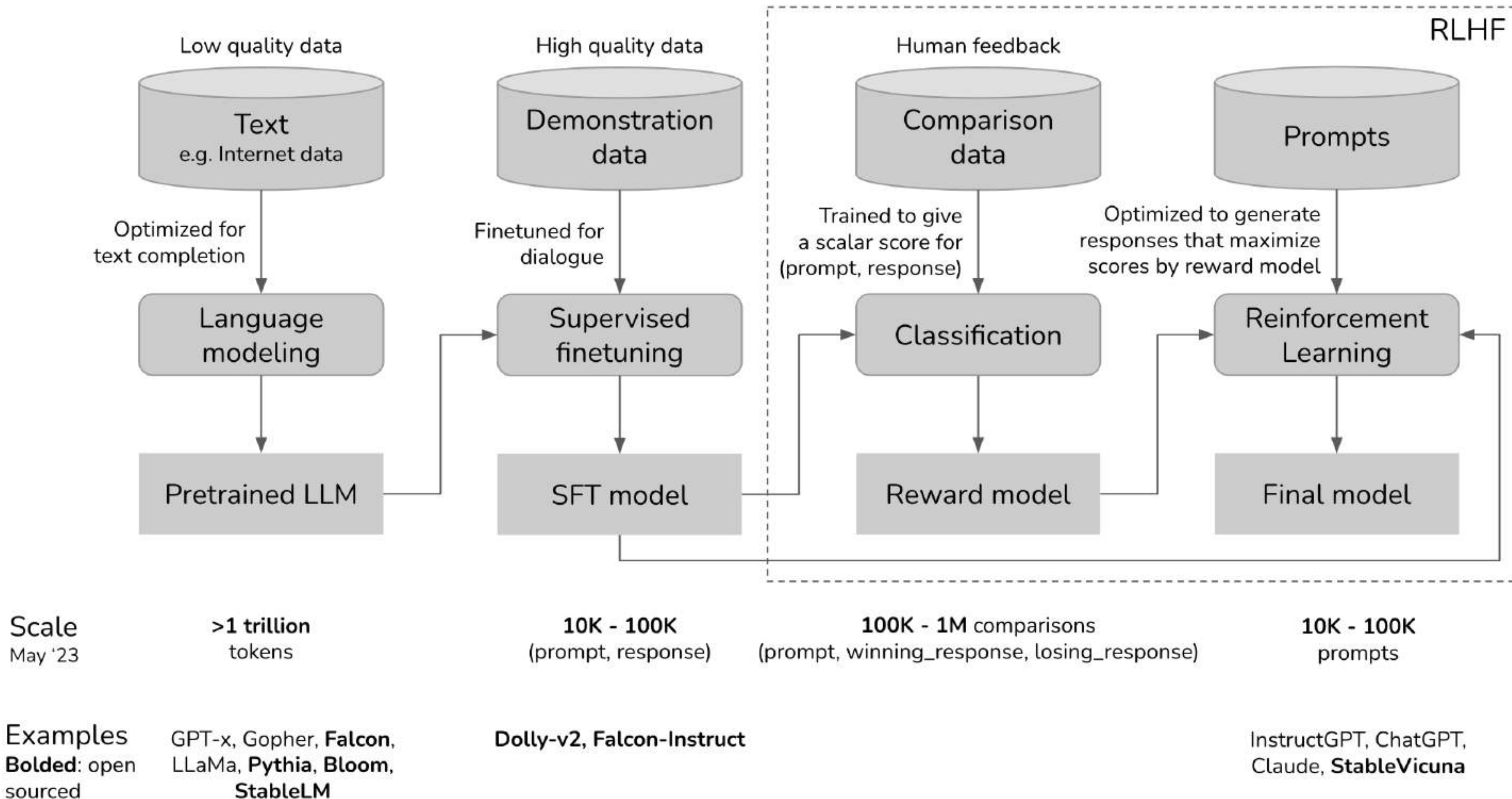


GPT Evolution

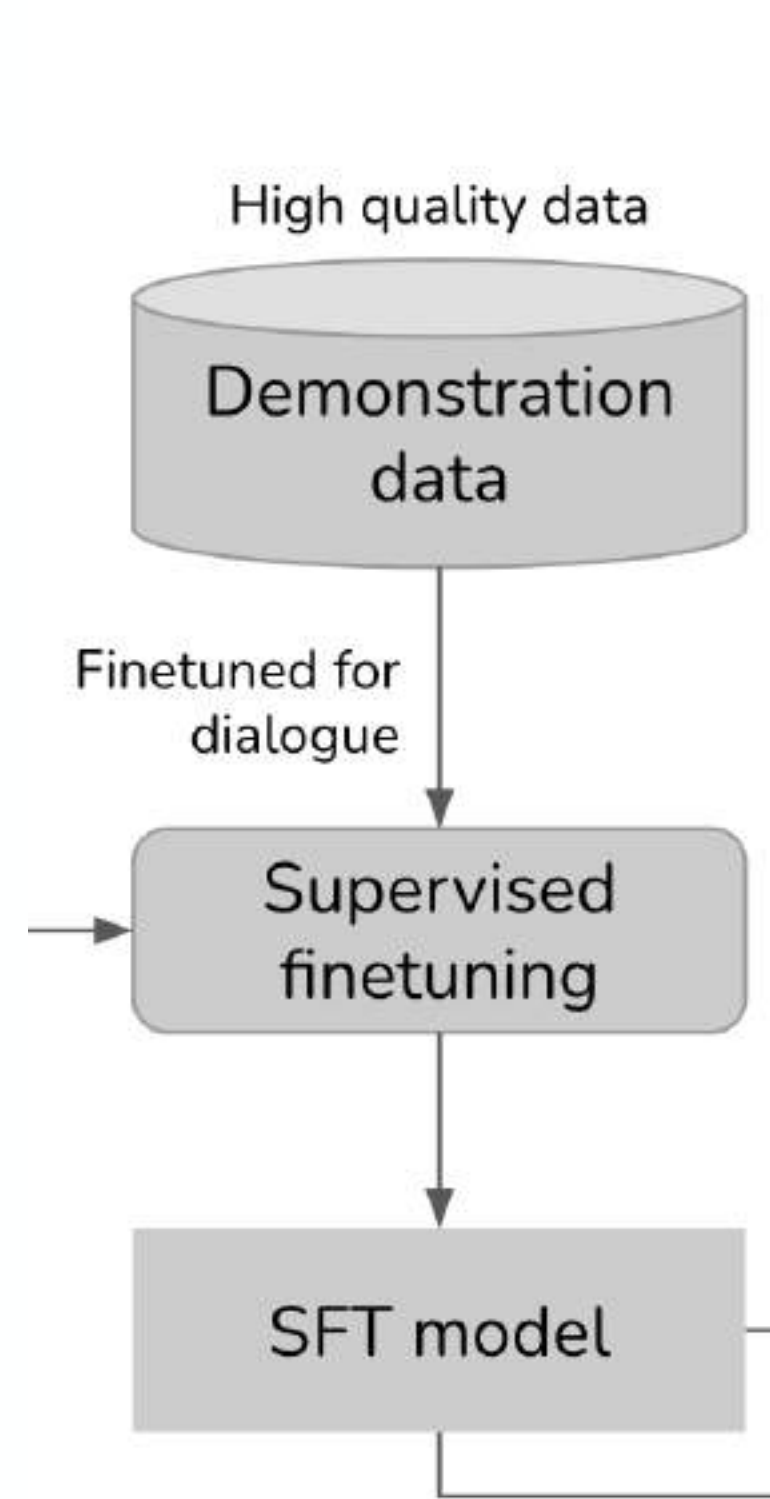


Here comes the LLMs!!!

3 Steps for Training LLMs

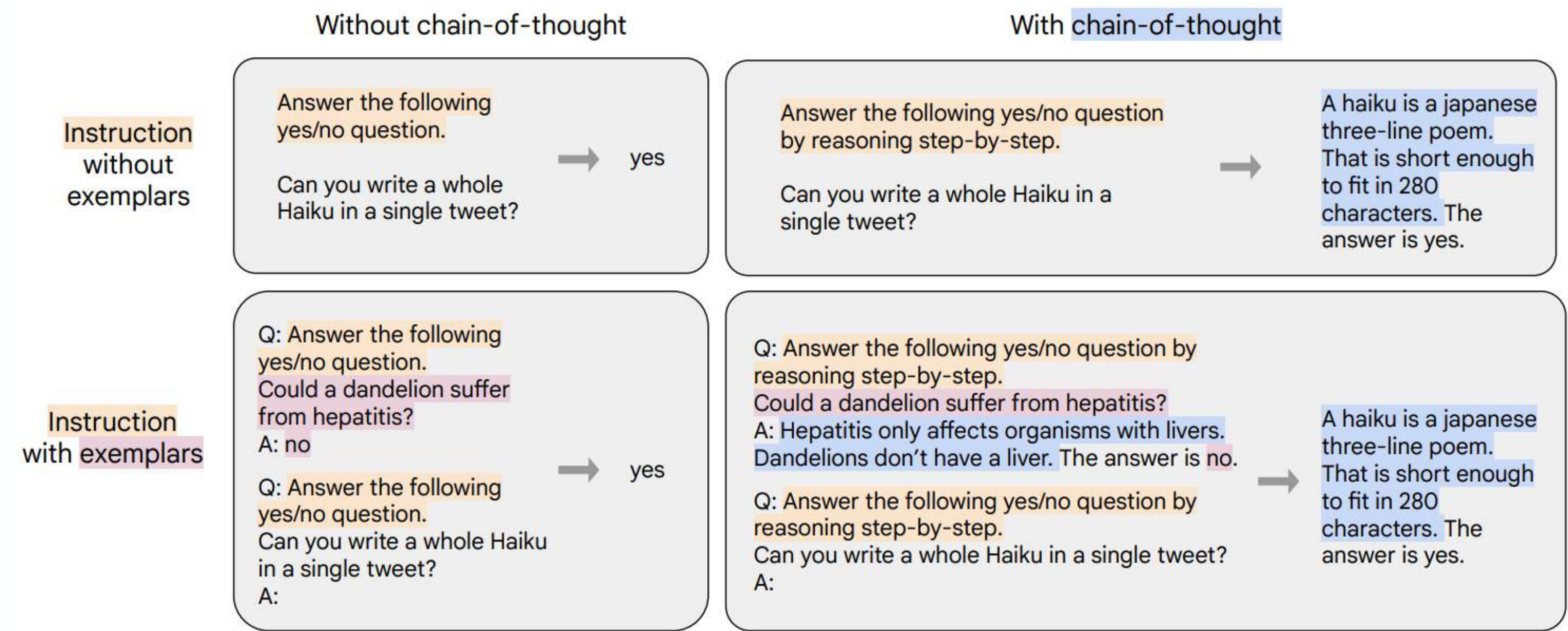
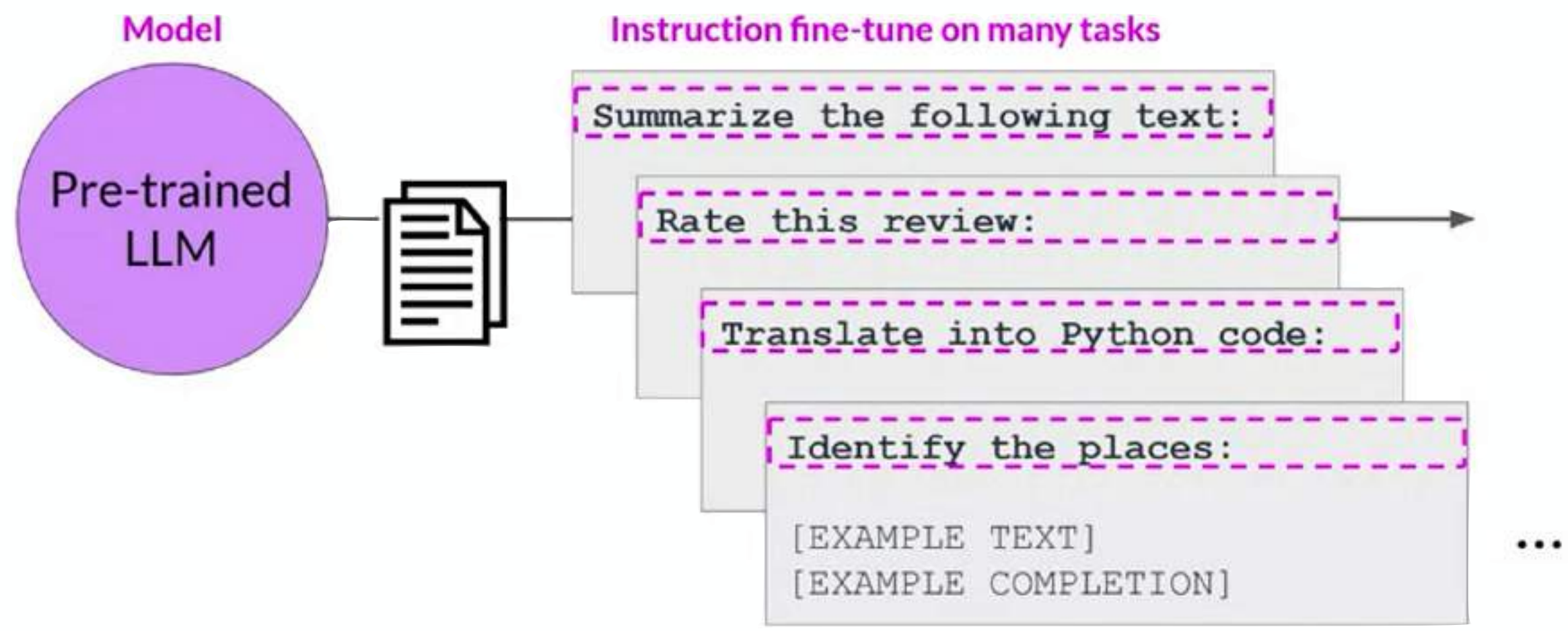


Instruction Fine-Tuning

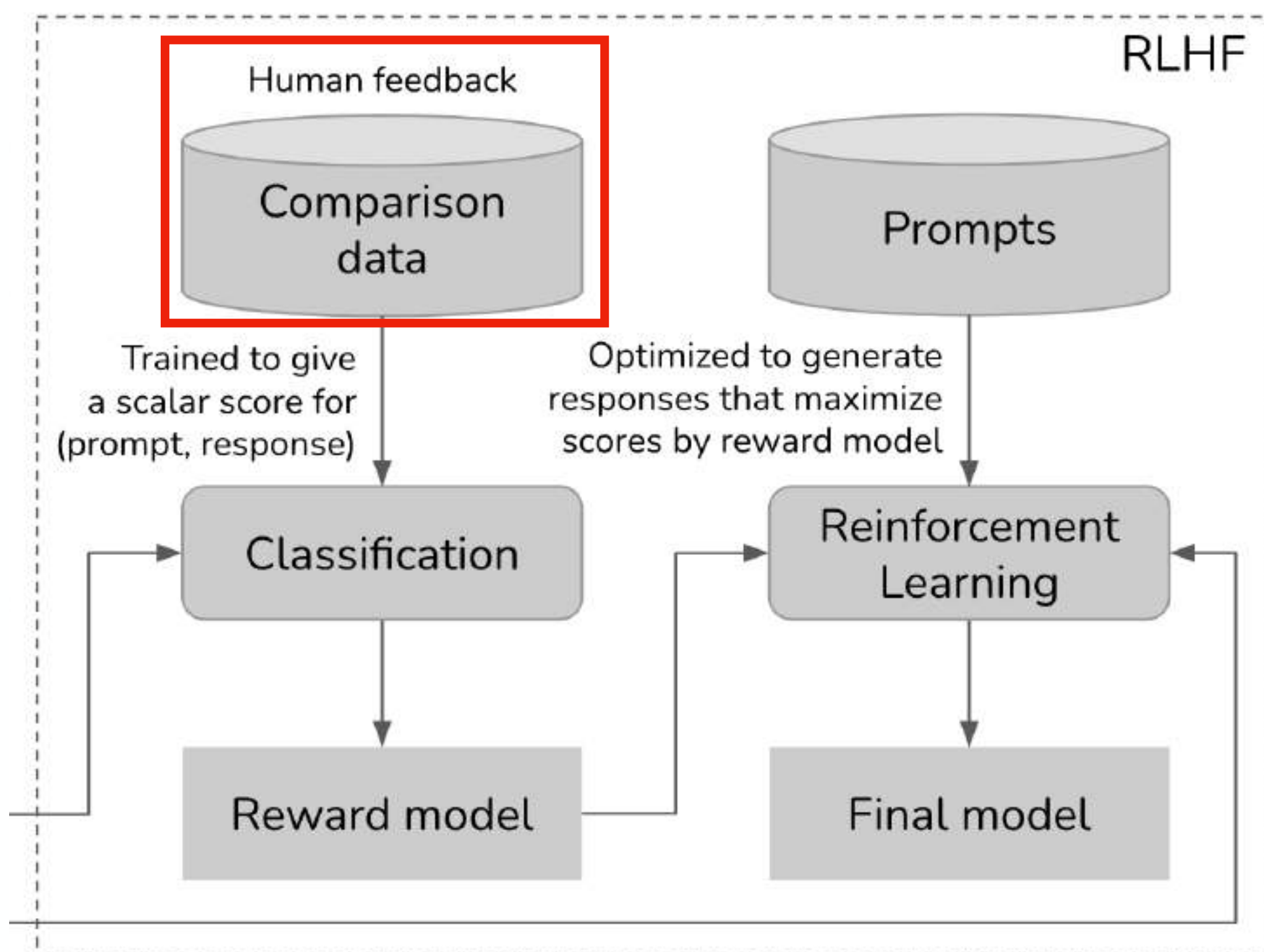


10K - 100K
(prompt, response)

Dolly-v2, Falcon-Instruct



Reinforcement Learning from Human Feedback



100K - 1M comparisons
(prompt, winning_response, losing_response)

10K - 100K
prompts

InstructGPT, ChatGPT,
Claude, **StableVicuna**

The screenshot shows a ChatGPT interface with the following content:

BO $RI = 2.44 * ((Mp / Mp')^{1/3}) * Rp$

Which response do you prefer?
Your choice will help make ChatGPT better.

Response 1

It looks like you've provided an equation, which appears to involve some variables. This equation could represent a relationship between various quantities. Let's break down the components:

- **RI**: This seems to be a variable, representing some quantity.
- **Mp**: It appears to be another variable, likely representing a mass or a quantity of some sort.
- **Mp'**: This could be another quantity of mass or a reference mass.
- **Rp**: This appears to be yet another variable, representing something else.

The equation you've given suggests that **RI** is calculated based on the ratio of **Mp** to **Mp'**, with some scaling factors. The presence of exponents (1/3) indicates a cube root operation.

If you can provide more context about the meaning of these variables and the overall purpose of this equation, I can give you a more detailed evaluation or interpretation of it. Additionally, please make sure the equation is correctly formatted with appropriate parentheses and symbols for a clear understanding.

Response 2

The expression you've provided appears to be a mathematical equation. Let's break it down:

- **RI** represents a value or variable.
- 2.44 is a constant.
- $(Mp / Mp')^{1/3}$ represents the cube root of the ratio of two values, **Mp** and **Mp'**.
- **Rp** is another variable.

The equation seems to be a formula for calculating **RI** based on these variables and constants. To evaluate it, you would need specific values for **Mp**, **Mp'**, and **Rp**. Once you have those values, you can plug them into the equation to calculate **RI**.

Keep in mind that without specific values for these variables, I can't provide you with a numerical result. If you have values for **Mp**, **Mp'**, and **Rp** that you'd like to plug into this equation, please provide them, and I can help you calculate **RI**.