Transformer model

Byte pair encoding

Sentence embeddings

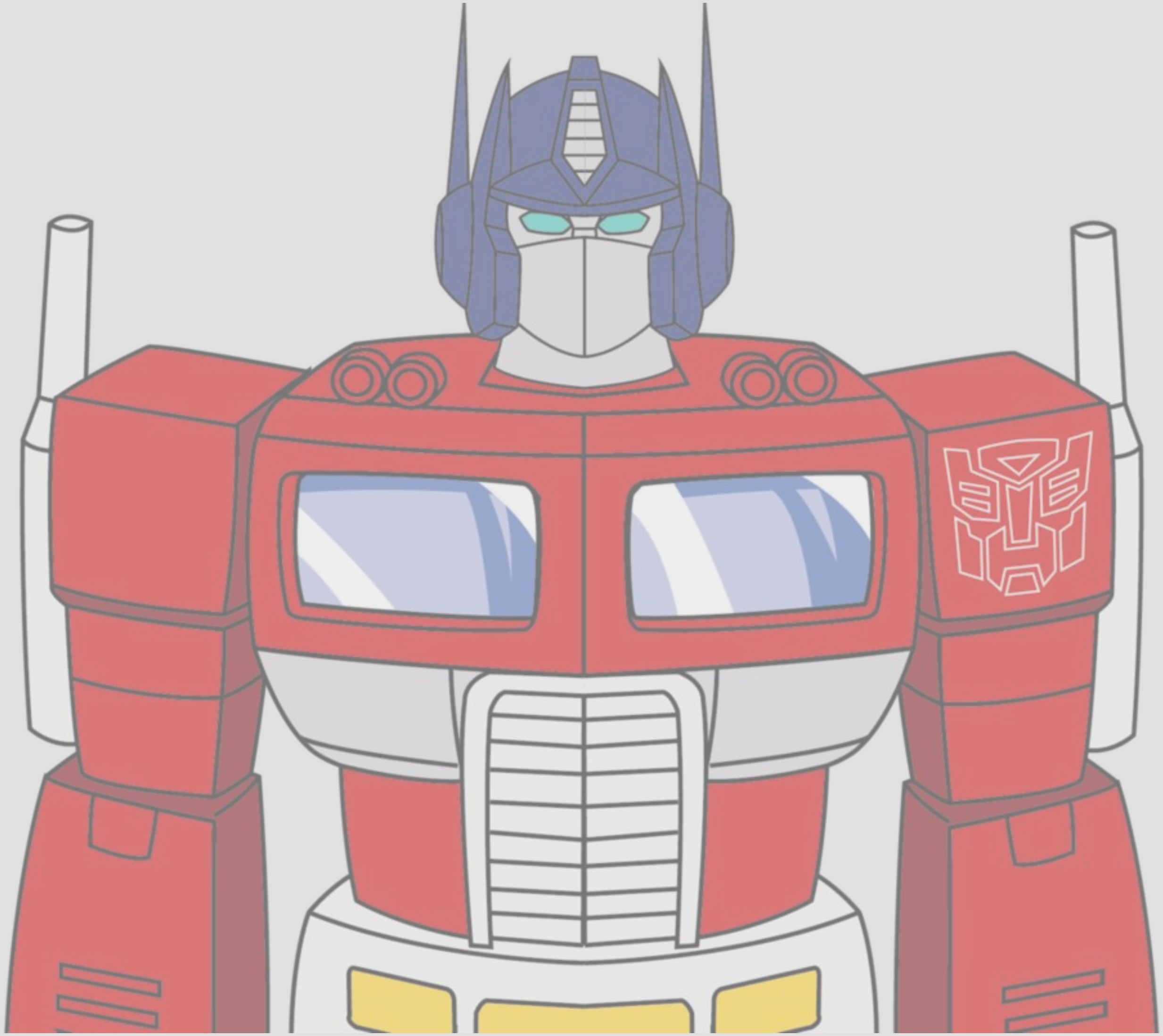
BERT

Transformer models evaluation

Based on lecture by Ekaterina Artemova

Transformer model

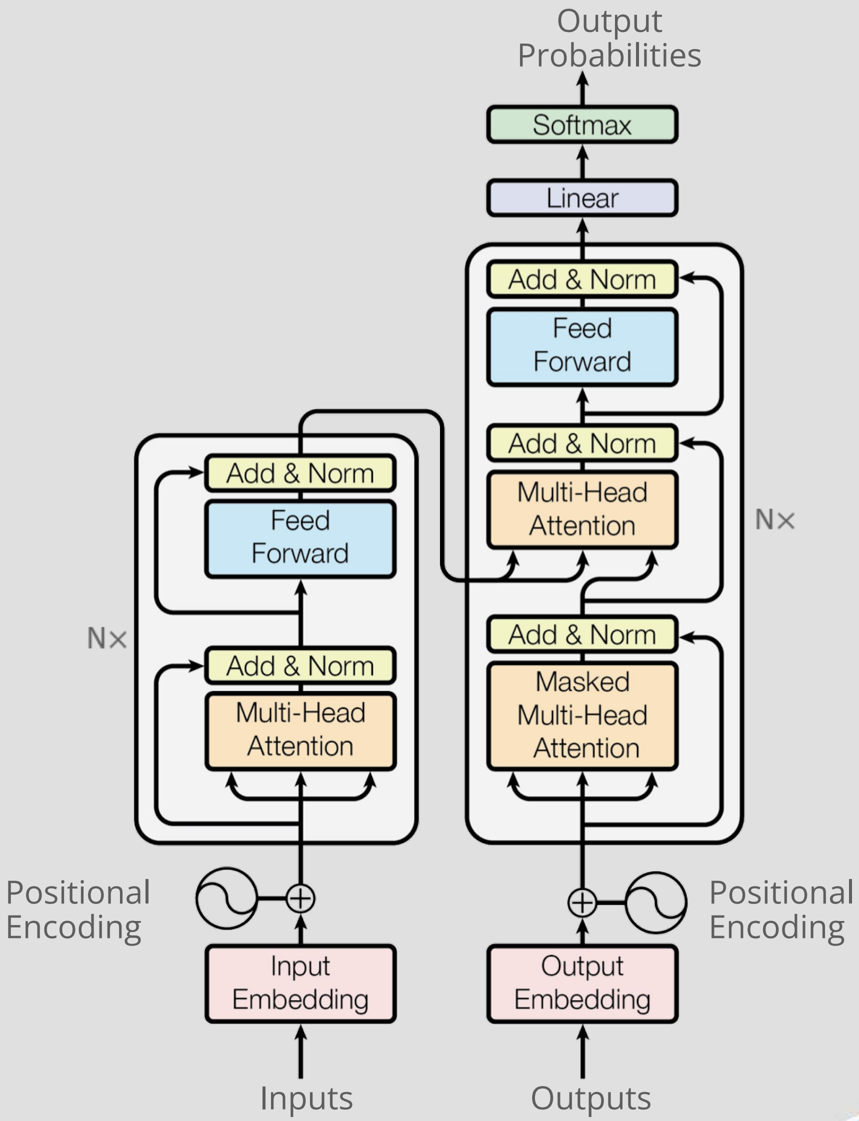
Transformer



- *Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In Advances in neural information processing systems, pp. 5998-6008. 2017.*

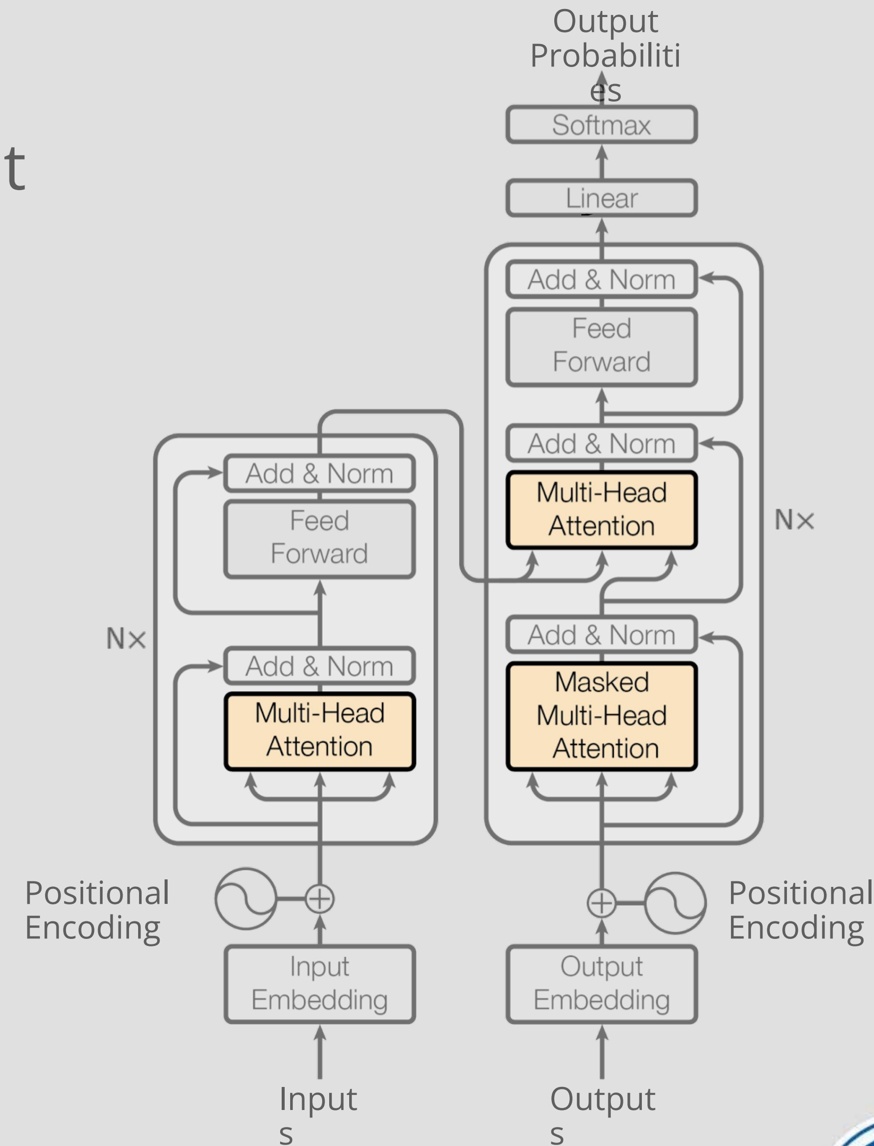
Transformer

- Encoder-decoder without RNN



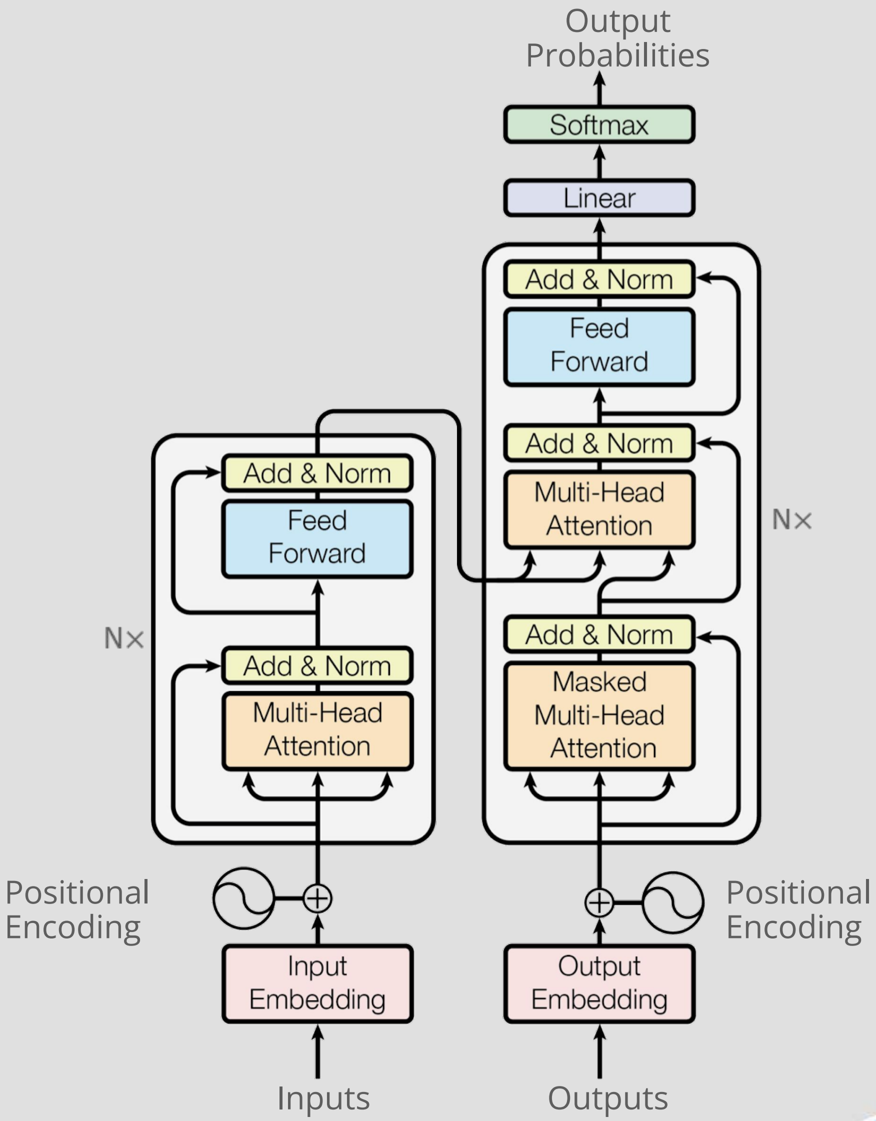
Transformer

- Encoder-decoder without RNN
- New layer: multi-head self-attention
- Outstanding results in both learning speed and translation quality



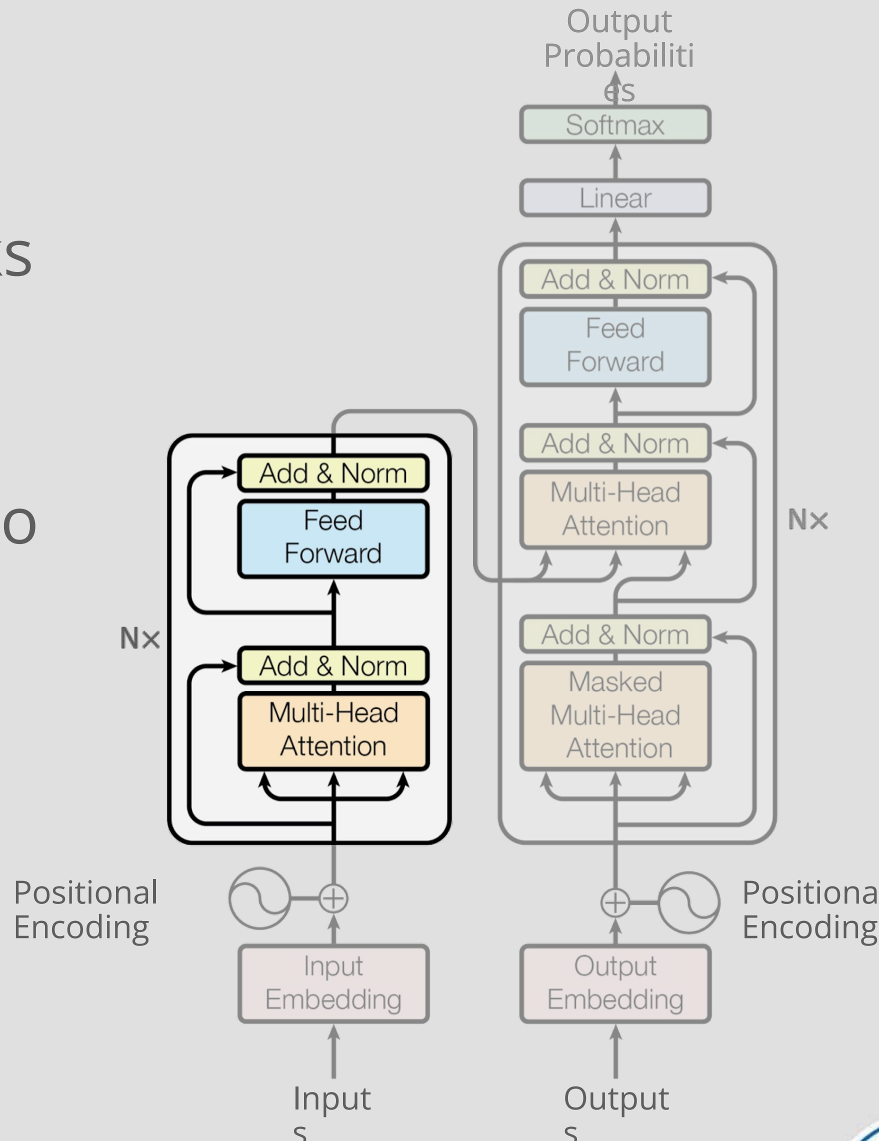
Transformer

- Both encoder and decoder consist of 6 identical blocks
- The blocks are organised sequentially
- Each block has its own weights



Encoder

- First block input - word embeddings, other blocks take previous block's output as input
- Each block consists of two parts: self-attention and feed forward layer



Encoder

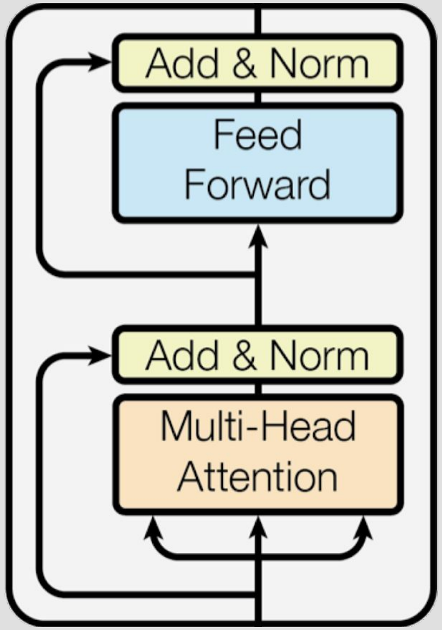
Self-attention: each input embedding is transformed 3 times:

query: $q_i = W^Q x_i$

key: $k_i = W^K x_i$

value: $v_i = W^V x_i$

Matrices W^Q, W^K, W^V are trainable.



Encoder

- Self-attention in matrix form:

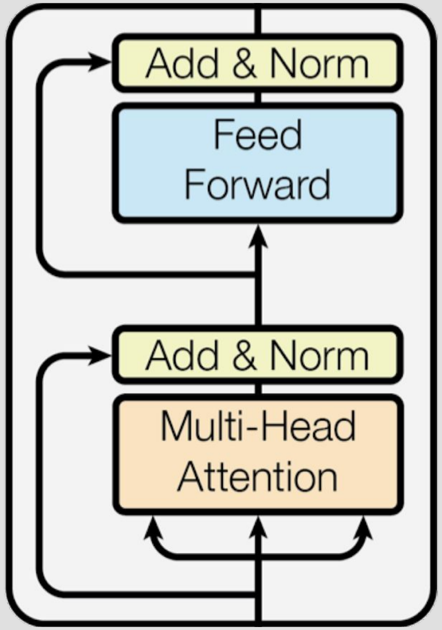
$$Z = softmax\frac{(QK^{\top})}{\sqrt{d_k}}V$$

- Several self-attention heads:

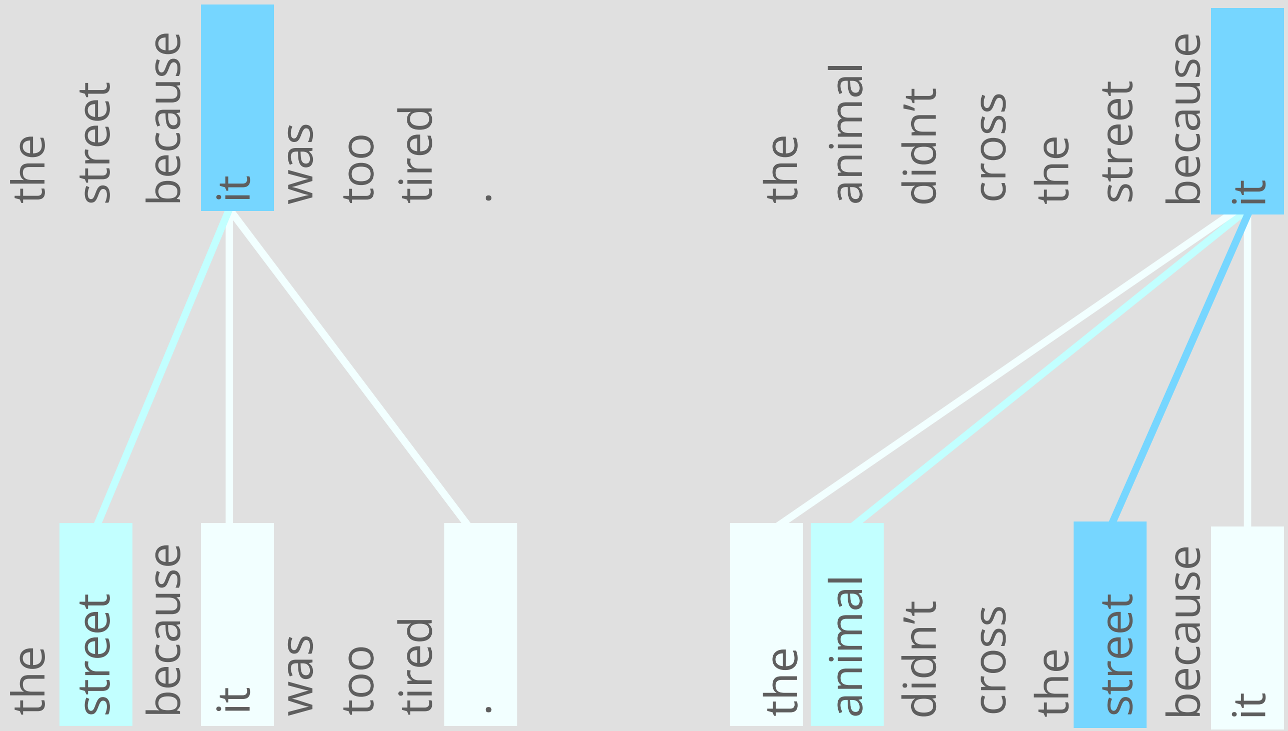
$$Z_1, \ldots, Z_8$$

- Concatenate self-attention embeddings:

$$Z = W^{O}[Z_1, \ldots, Z_8]$$



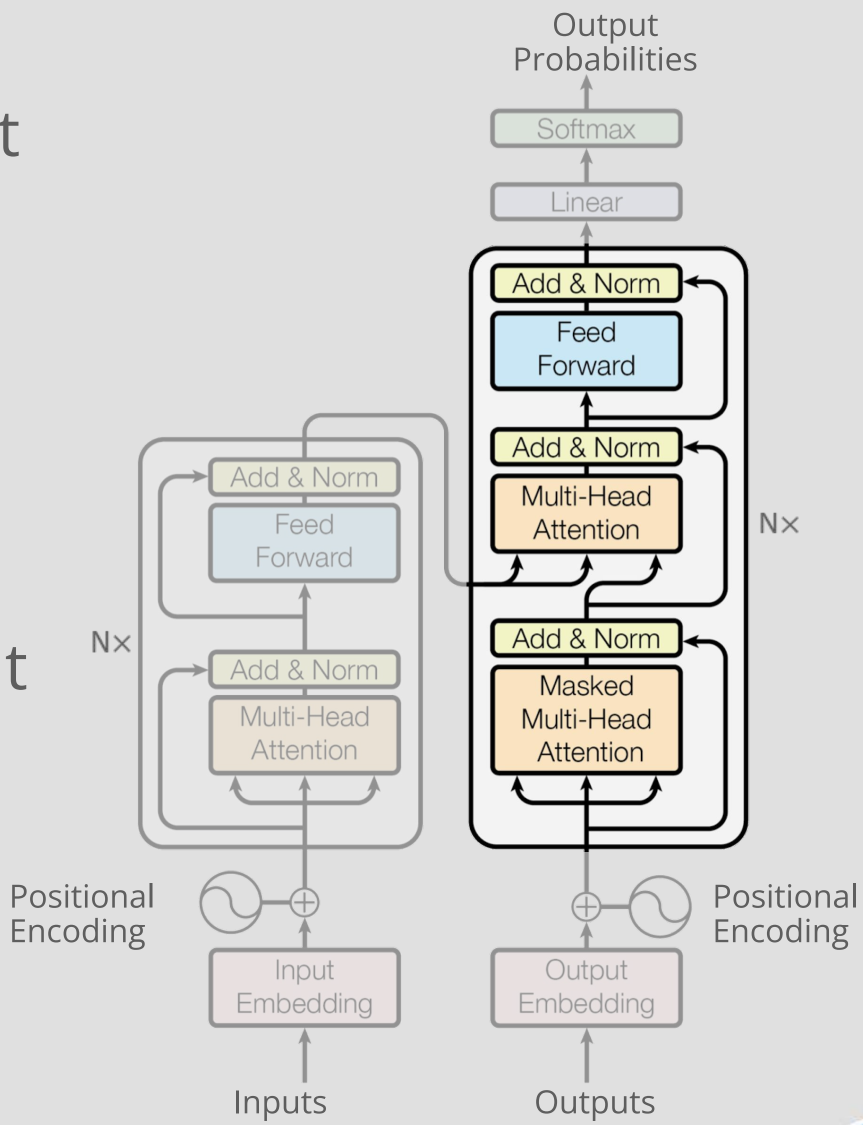
Attention weights



ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Decoder

- Encoder transforms input words into matrices: $Q^{enc}, K^{enc}, V^{enc}$
- Encoder passes matrices K^{enc}, V^{enc} into the decoder
- Decoder transforms input words into matrix Q^{dec}

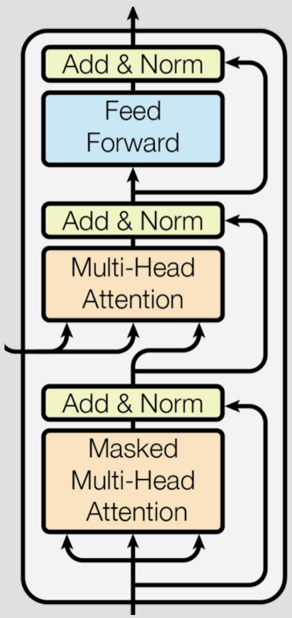


Decoder

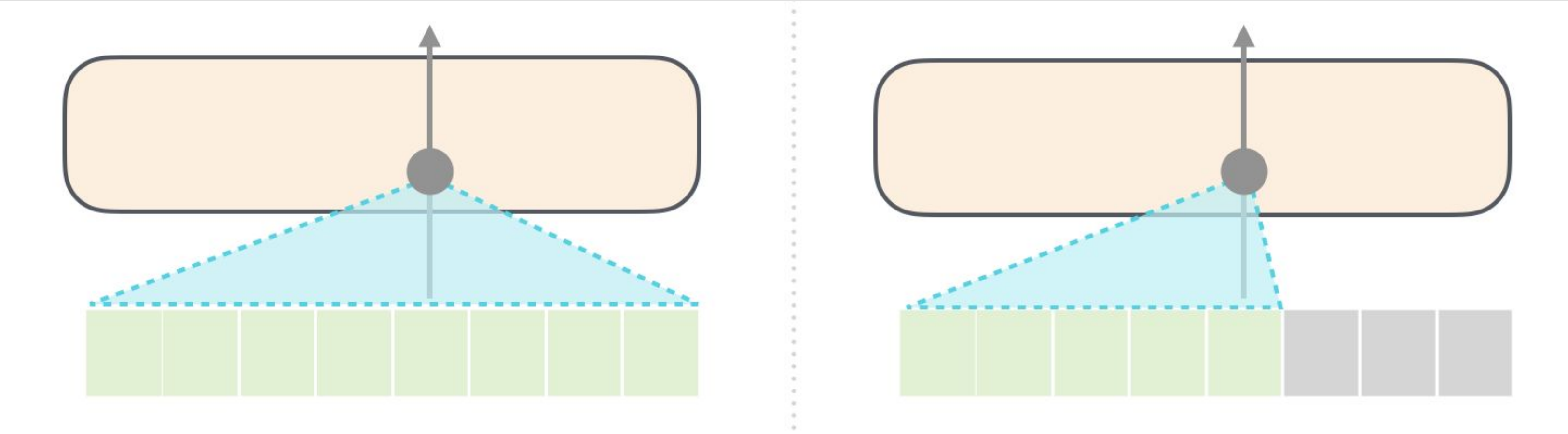
- Decoder translates word by word

- Masked attention mechanism prevents looking at the following words (by setting their weights to $-\infty$

- Masked attention is applied only during training process



Attention mechanism in Transformer



Attention mechanism in encoder

Attention mechanism in decoder

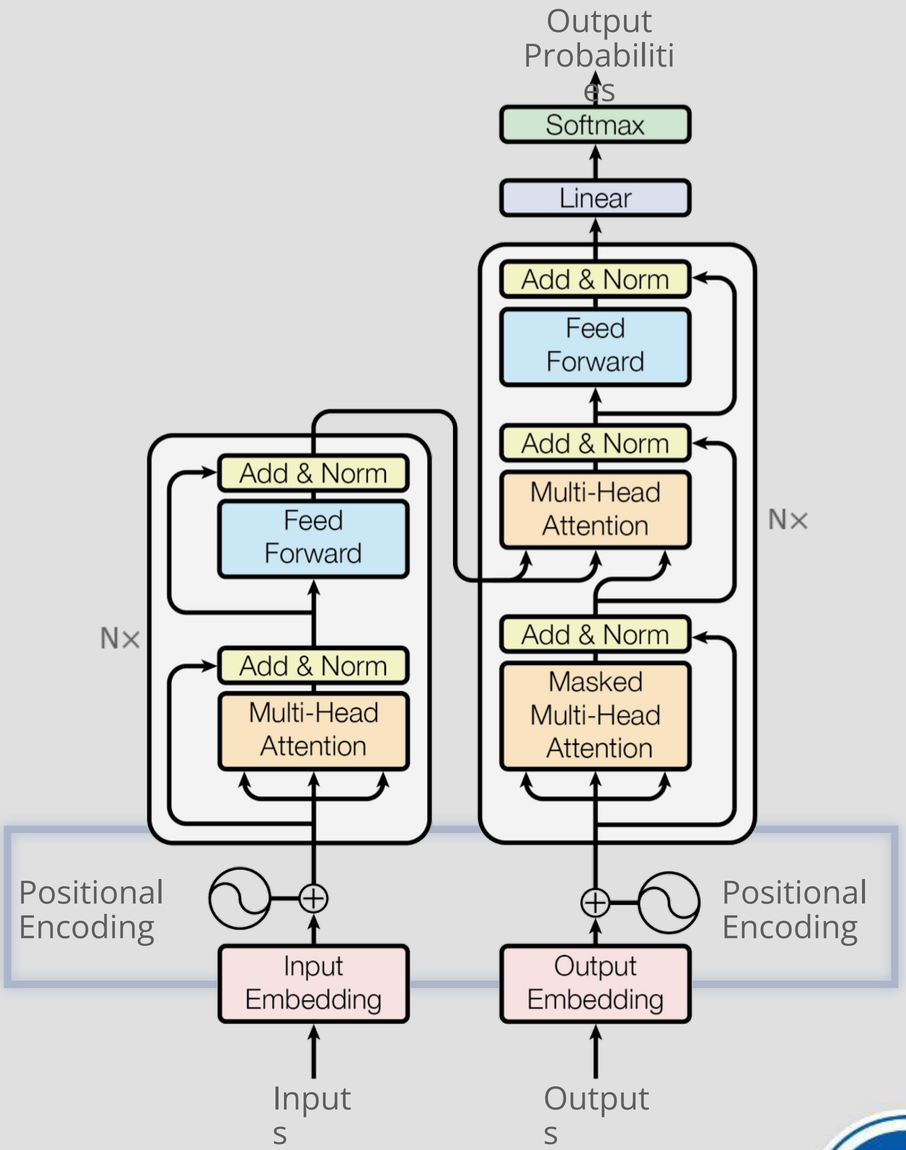
Transformer

- Word order is taken into account using positional encoding:

$$x_i = x_i + pe_i$$

- It encapsulates information about words mutual distance in the sentence

- Trainable positional encoding is trained with the model

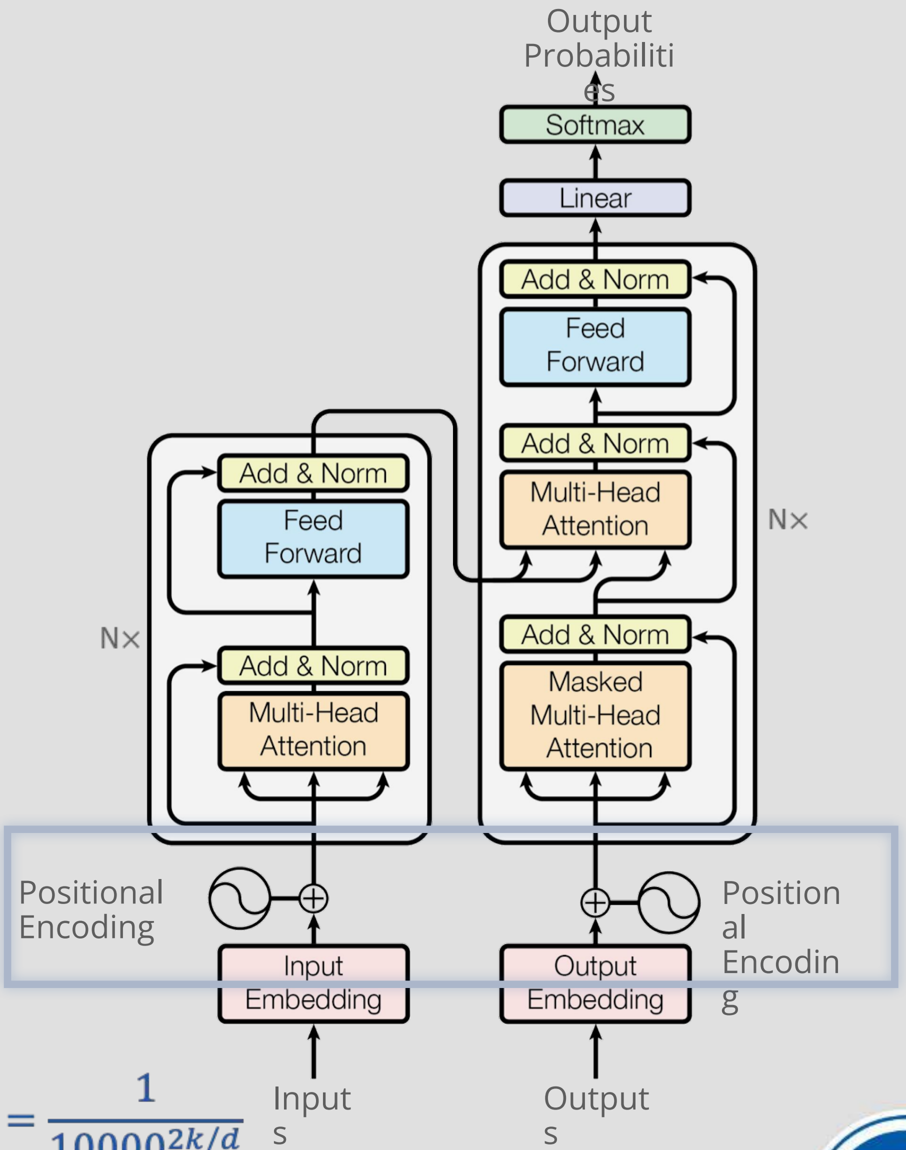


Transformer

- Word order is taken into account using positional encoding:

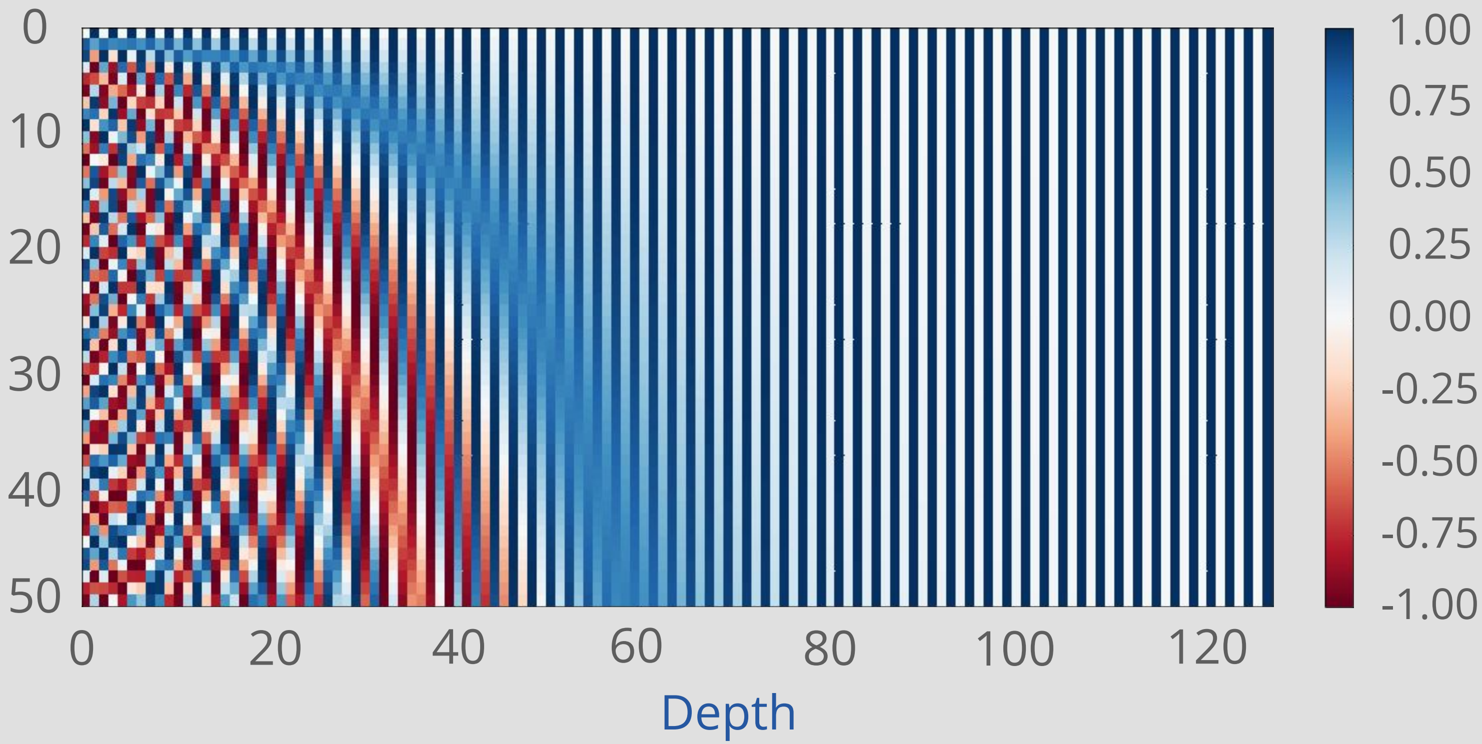
$$x_i = x_i + pe_i$$

- Trigonometric positional encoding:



- $$pe_t^{(i)} = \begin{cases} \sin(\omega_k, t), \text{если } i = 2k \\ \cos(\omega_k, t), \text{если } i = 2k+1 \end{cases}, \text{где } \omega_k = \frac{1}{10000^{2k/d}}$$

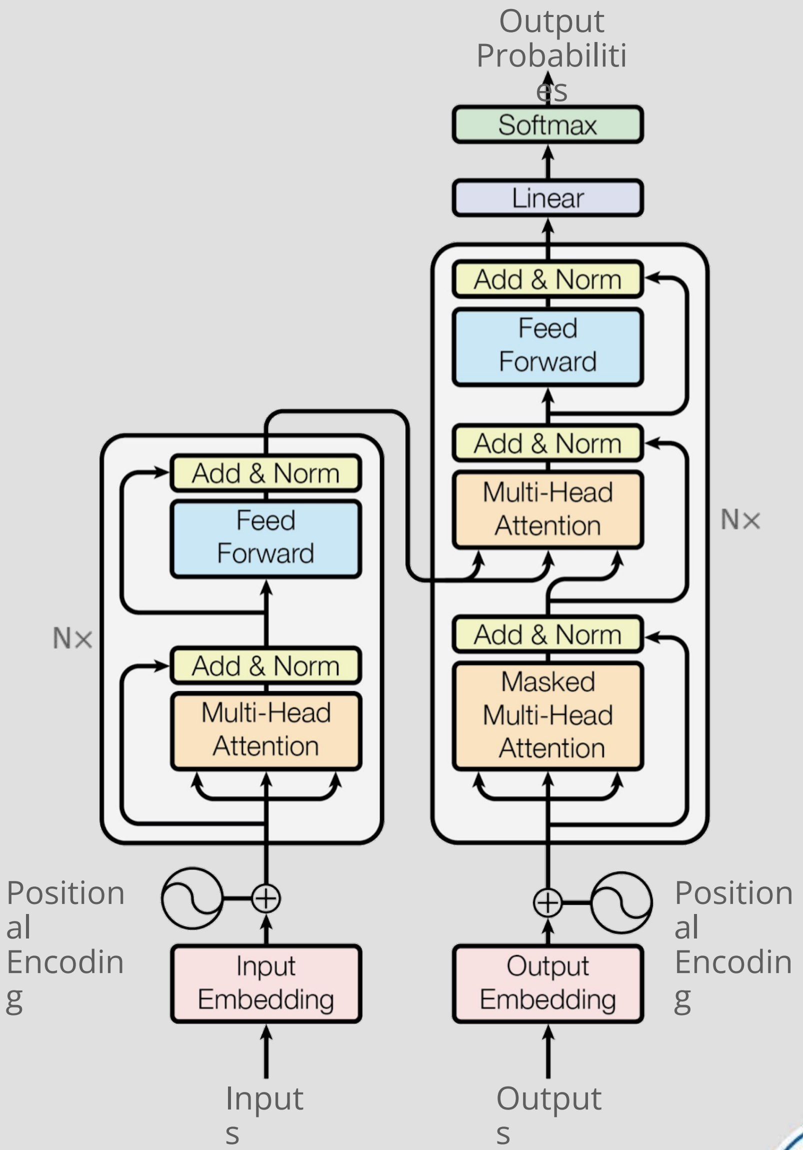
Trigonometric positional embeddings



- *https://kazemnejad.com/blog/transformer_architecture_positional_encoding/*

Transformer

- Dominant architecture since mid-2018 in most automated word processing tasks
- Surpasses previous architectures in speed and quality



Byte pair encoding

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	
3 000	
5 000	
10 000	
25 000	
50 000	
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	
5 000	
10 000	
25 000	
50 000	
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	
10 000	
25 000	
50 000	
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	_г , рё , зо , б , ла , жен , ству , ющий ,
10 000	
25 000	
50 000	
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	_г , рё , зо , б , ла , жен , ству , ющий ,
10 000	_г , рё , зоб , ла , жен , ству , ющий ,
25 000	
50 000	
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	_г , рё , зо , б , ла , жен , ству , ющий ,
10 000	_г , рё , зоб , ла , жен , ству , ющий ,
25 000	_г , рё , зоб , ла , жен , ствующий ,
50 000	
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	_г , рё , зо , б , ла , жен , ству , ющий ,
10 000	_г , рё , зоб , ла , жен , ству , ющий ,
25 000	_г , рё , зоб , ла , жен , ствующий ,
50 000	_г , рё , зоб , ла , жен , ствующий ,
100 000	
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	_г , рё , зо , б , ла , жен , ству , ющий ,
10 000	_г , рё , зоб , ла , жен , ству , ющий ,
25 000	_г , рё , зоб , ла , жен , ствующий ,
50 000	_г , рё , зоб , ла , жен , ствующий ,
100 000	_грё , зоб , ла , жен , ствующий ,
200 000	

Byte pair encoding (BPE)

- Combining frequent character pairs
- Perform a fixed number of merge operations

Iteration	BPE
0	_г р ё з о б л а ж е н с т в у ю щ и й
1 000	_г , р , ё , зо , б , ла , же , н , ству , ющи , й ,
3 000	_г , рё , зо , б , ла , жен , ству , ющий ,
5 000	_г , рё , зо , б , ла , жен , ству , ющий ,
10 000	_г , рё , зоб , ла , жен , ству , ющий ,
25 000	_г , рё , зоб , ла , жен , ствующий ,
50 000	_г , рё , зоб , ла , жен , ствующий ,
100 000	_грё , зоб , ла , жен , ствующий ,
200 000	_грё , зобла , жен , ствующий

Representation by symbols, subwords and words

Symbols

Representation by symbols, subwords and words

Symbols

в	е	с	е	л	ы	й
3	6	19	6	13	29	11
в	е	с	е	л	е	е
3	6	19	6	13	6	6
б	о	д	р	ы	й	
2	16	5	18	29	11	
б	о	д	р	е	е	
2	16	5	18	6	6	

Representation by symbols, subwords and words

Symbols

dictionary size

в	е	с	е	л	ы	й							
3	6	19	6	13	29	11							
в	е	с	е	л	е	е							
3	6	19	6	13	6	6							
б	о	д	р	ы	й								
2	16	5	18	29	11								
б	о	д	р	е	е								
2	16	5	18	6	6								

Representation by symbols, subwords and words

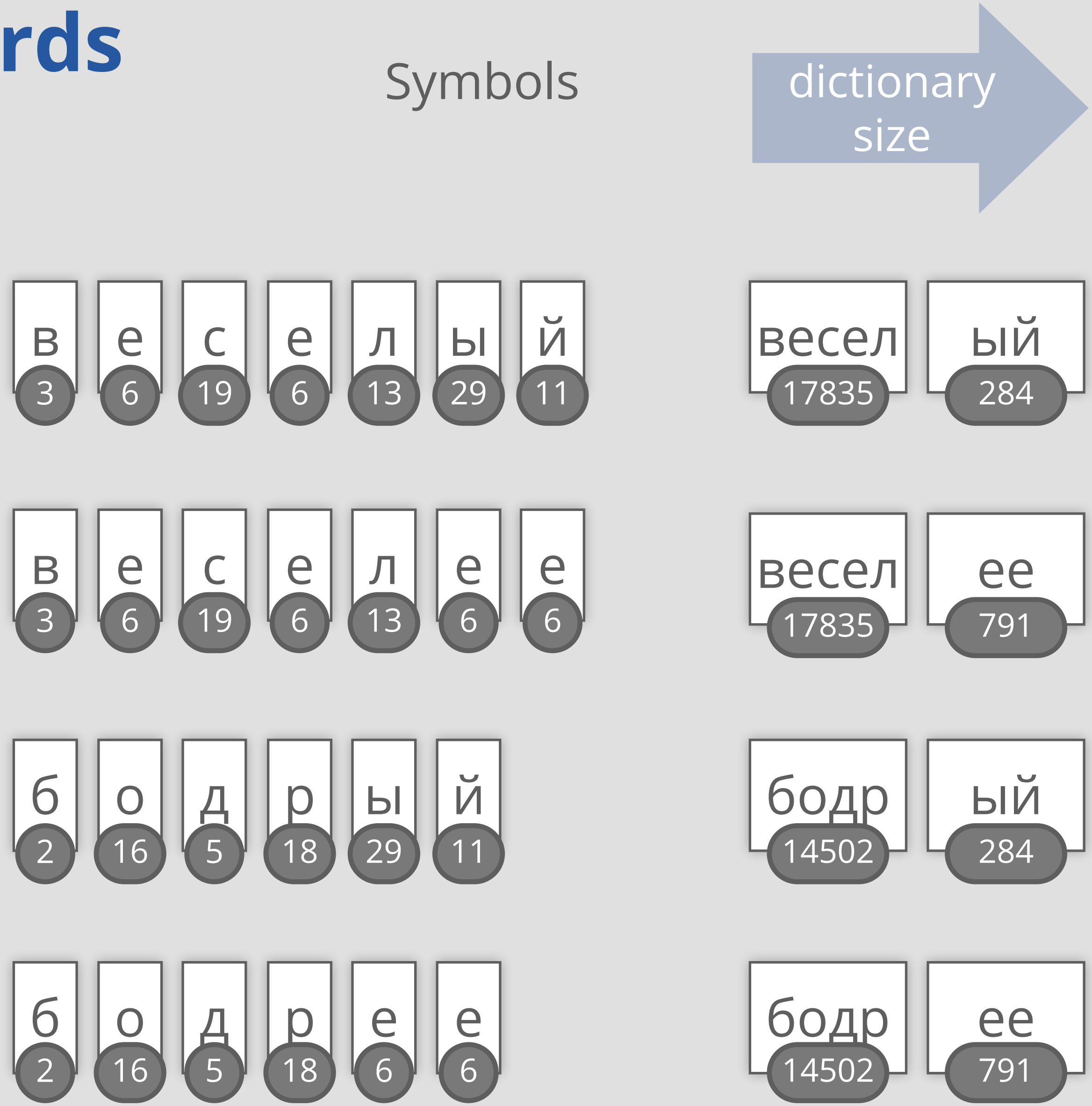
Symbols

dictionary size

в	е	с	е	л	ы	й
3	6	19	6	13	29	11
в	е	с	е	л	е	е
3	6	19	6	13	6	6
б	о	д	р	ы	й	
2	16	5	18	29	11	
б	о	д	р	е	е	
2	16	5	18	6	6	

весел	ый
17835	284
весел	ее
17835	791
бодр	ый
14502	284
бодр	ее
14502	791

Representation by symbols, subwords and words



Representation by symbols, subwords and words

Symbols	dictionary size →	Words
в 3, е 6, с 19, е 6, л 13, ы 29, й 11	весел 17835, ый 284	веселый 23941
в 3, е 6, с 19, е 6, л 13, е 6, е 6	весел 17835, ее 791	веселее 13008
б 2, о 16, д 5, р 18, ы 29, й 11	бодр 14502, ый 284	бодрый 1472
б 2, о 16, д 5, р 18, е 6, е 6	бодр 14502, ее 791	бодрее 18772

Symbols, subwords or words?

Translation by symbols

+ better transliteration
+ better morphology understanding
- requires difficult computations

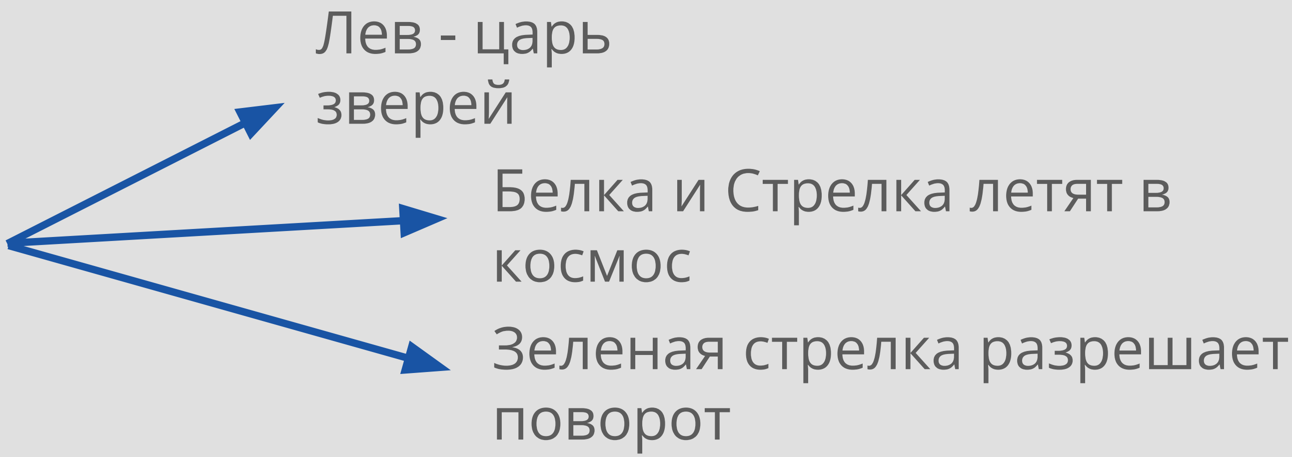
Translation by subwords

+ grammatically correct
+ supports richer dictionary
+ understand syntax better
+ performs better

Sentence embeddings

Sentence embeddings

- Small vector (~ 300 components)
- The dimension of the vector does not depend on the number of words in the sentence
- Each sentence corresponds to one vector
- The closer the sentences are in meaning, the closer the sentence vectors



Averaging word embeddings

averaging

$$\begin{cases} [w_{11}, w_{12}, \ldots, w_{1n}], \\ [w_{21}, w_{22}, \ldots, w_{2n}], \\ [w_{31}, w_{32}, \ldots, w_{3n}], \end{cases}$$

$\overrightarrow{\text{лев}}$

$\overrightarrow{\text{царь}}$

$\overrightarrow{\text{зверей}}$

$$\underbrace{\left[\frac{w_{11} + w_{21} + w_{21}}{3}, \frac{w_{12} + w_{22} + w_{32}}{3}, \ldots, \frac{w_{1n} + w_{2n} + w_{3n}}{3}\right]}_{\text{sentence embedding}}$$

$\overrightarrow{\text{лев царь зверей}}$

Averaging word embeddings

Advantages:

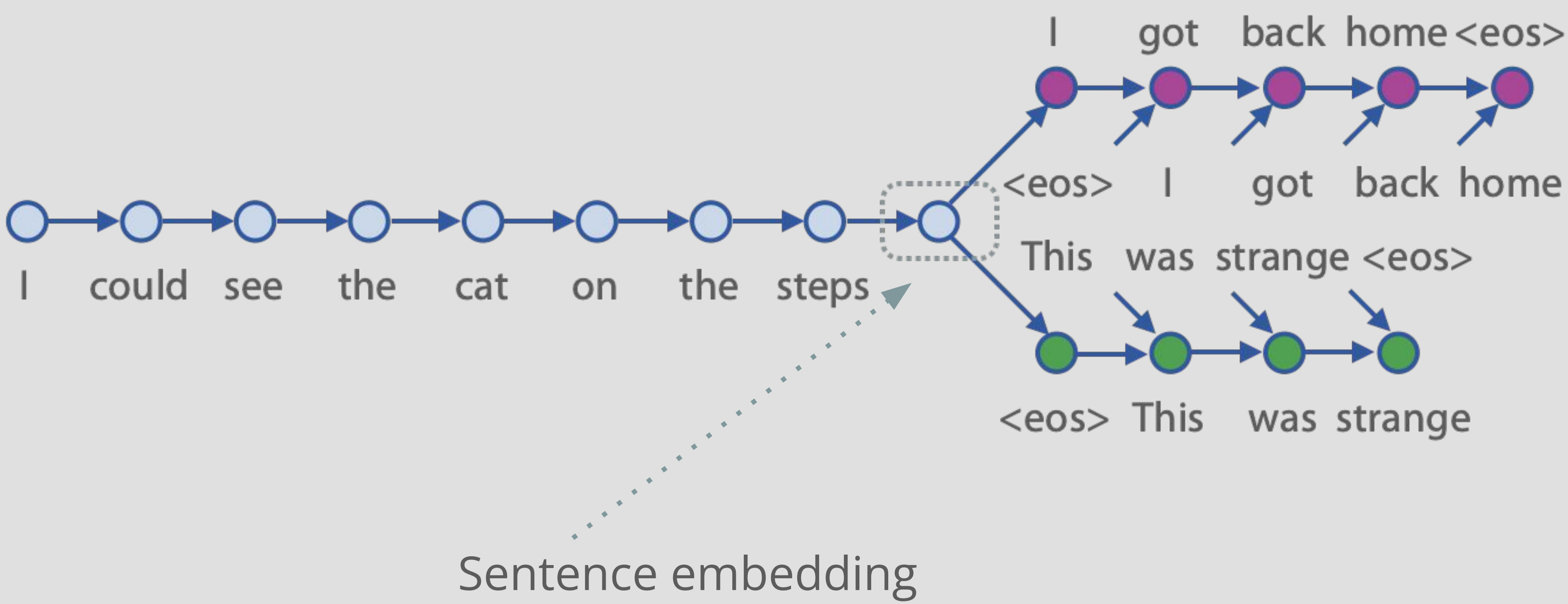
- quick
- pre-trained word embeddings can be used

Problems:

- words’ impact is equal (can be solved by adding tf-idf weights)
- all word embedding problems remain

Skip-thought model

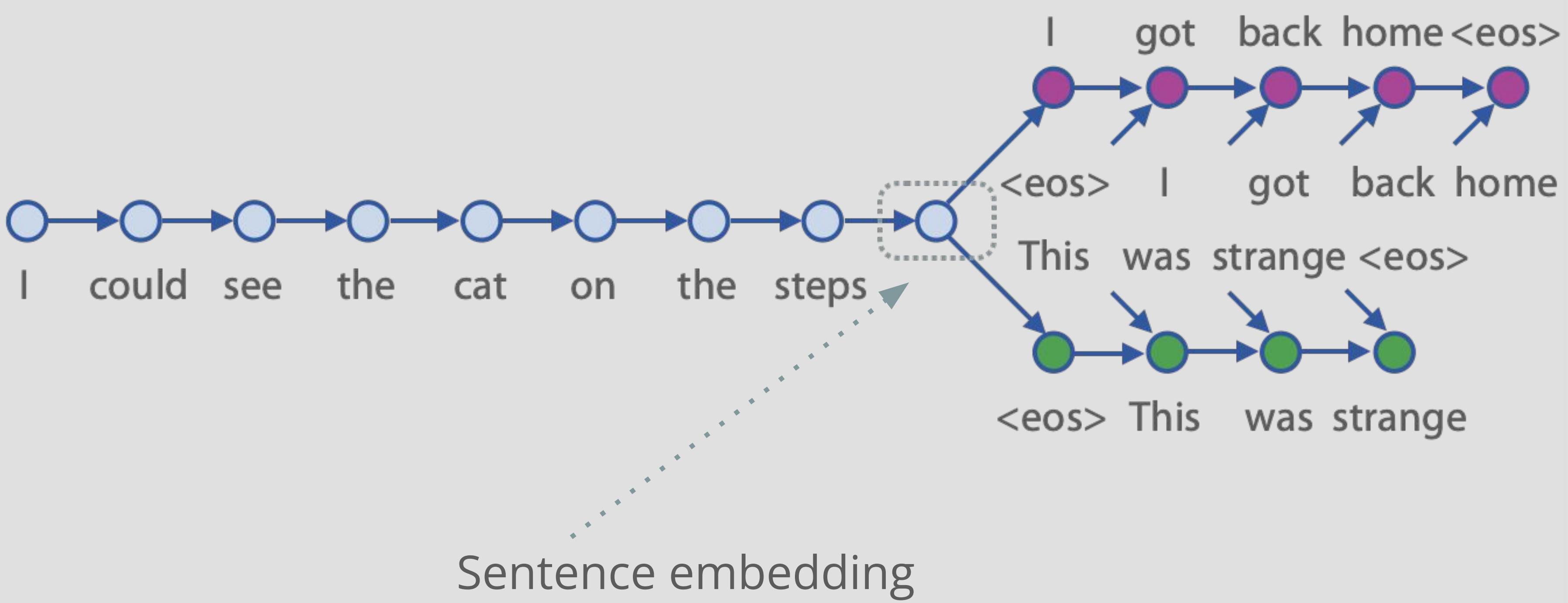
- model is based on encoder-decoder architecture



- *Ryan Kiros, Yukun Zhu, Russ R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, Sanja Fidler. "Skip-thought vectors." NeurIPS, 2015.*

Skip-thought model

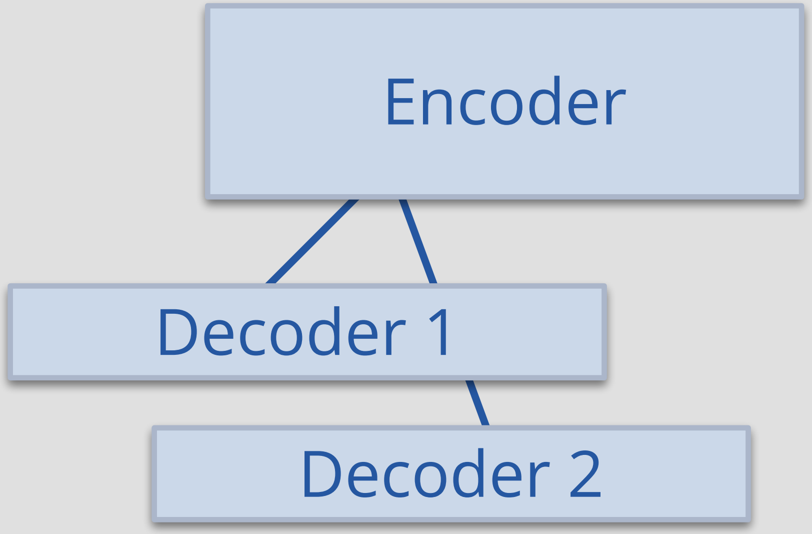
- model is based on encoder-decoder architecture
- predict previous and next sentence based on the current



- *Ryan Kiros, Yukun Zhu, Russ R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, Sanja Fidler. "Skip-thought vectors." NeurIPS, 2015.*

Universal sentence encoder

- model is based on encoder-decoder architecture
- encoder consists of Transformer blocks
- Several decoders for:
- Semi-supervised training: predict the next sentence
- Supervised training: natural language inference, NLI

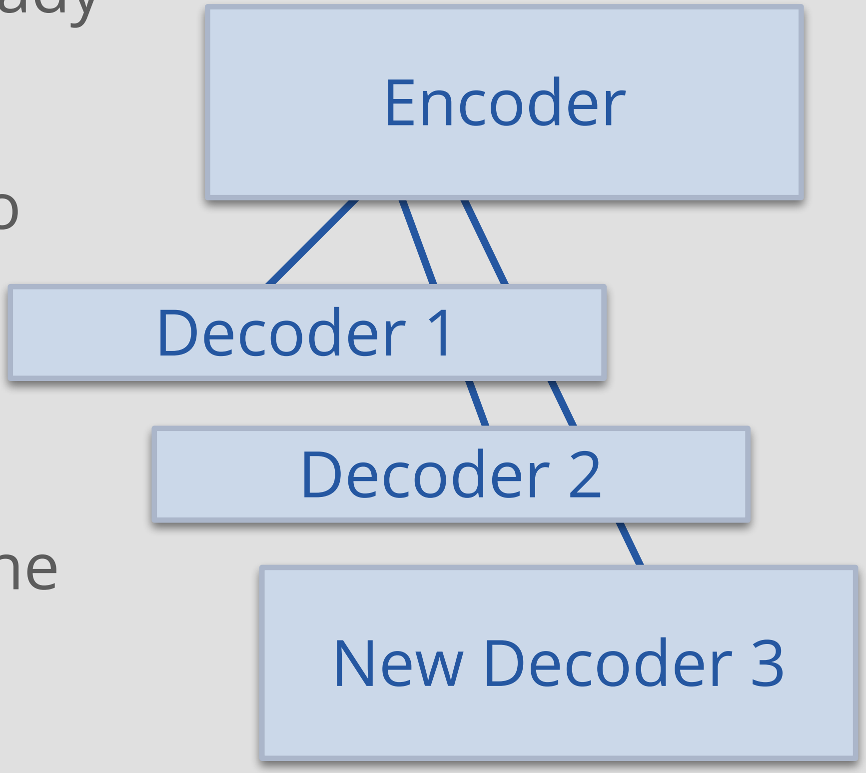


- *Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant . "Universal sentence encoder for English". EMNLP, 2018*

Universal sentence encoder

Transfer learning:

- Suppose the encoder is already trained
- Let's add a new decoder 3 to determine the sentiment of the sentence
- Decoders 1 and 2 played a supporting role in training the encoder

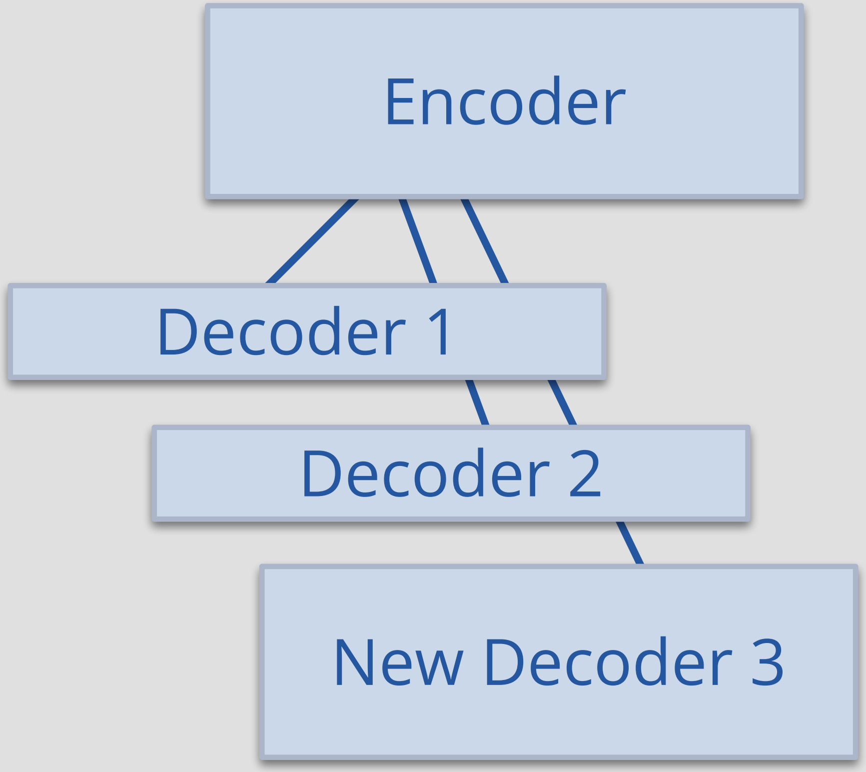


- *Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant . "Universal sentence encoder for English". EMNLP, 2018*

Universal sentence encoder

Transfer learning

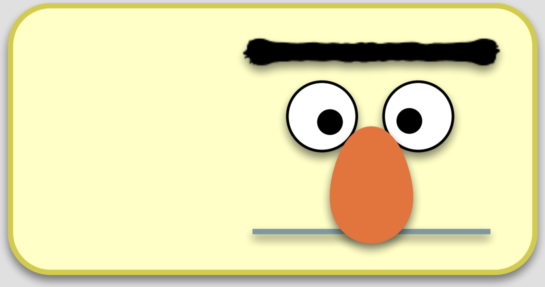
- Decoder 3 is used to solve a practical problem
- As a rule, it gives an increase in quality compared to training from scratch.



- *Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant . "Universal sentence encoder for English". EMNLP, 2018*

BERT model

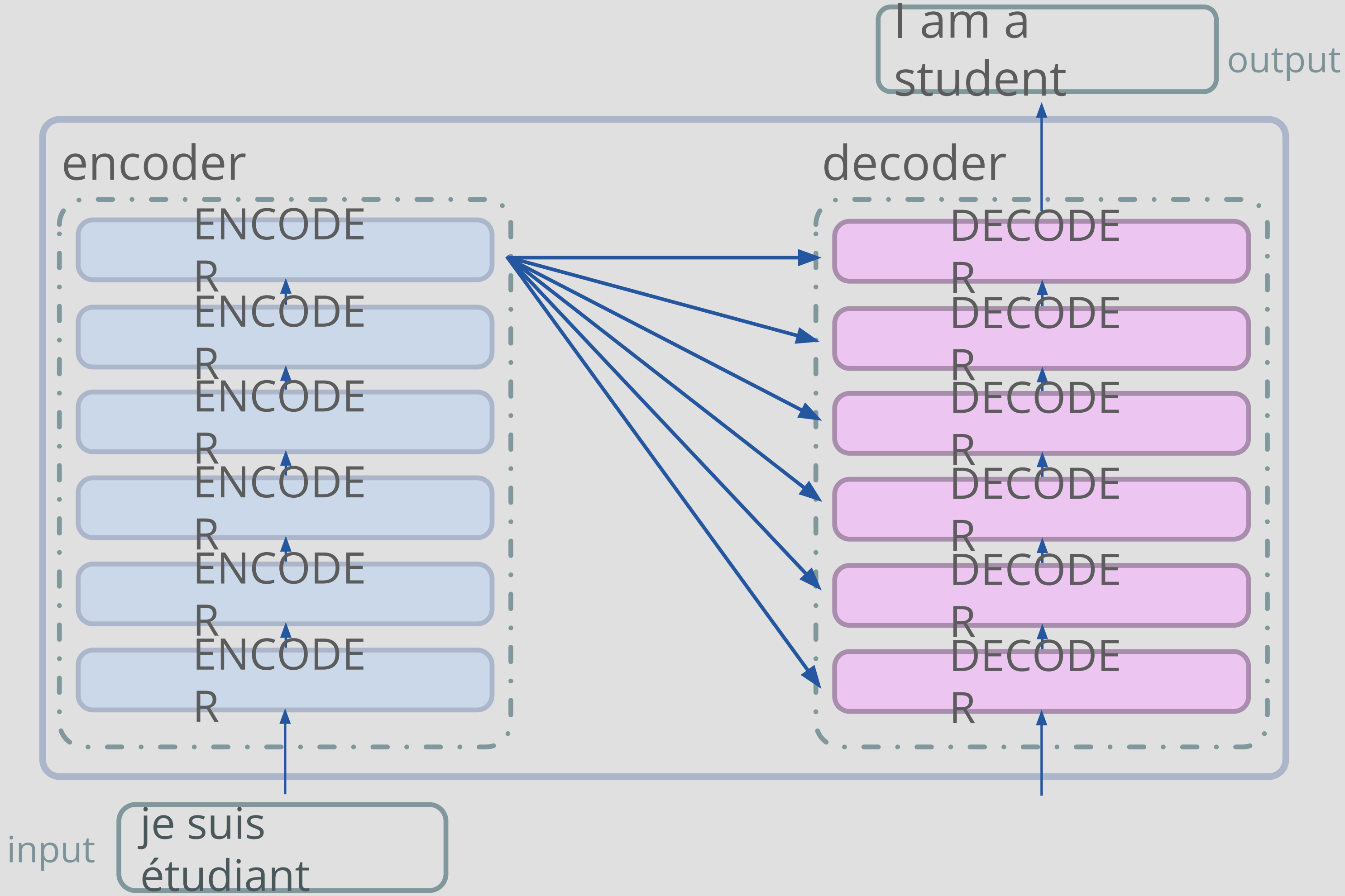
BERT model



- Appeared at the end of 2018 and made a real breakthrough, setting new records in most of the NLP tasks
- New paradigm in word processing: tuning a large pre-trained model for particular problems
- **The pre-trained model already knows a lot about the language**
- The use of pre-trained models opens up new perspectives: less data, higher quality indicators, less tuning time
- However, pre-training can take several weeks

- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

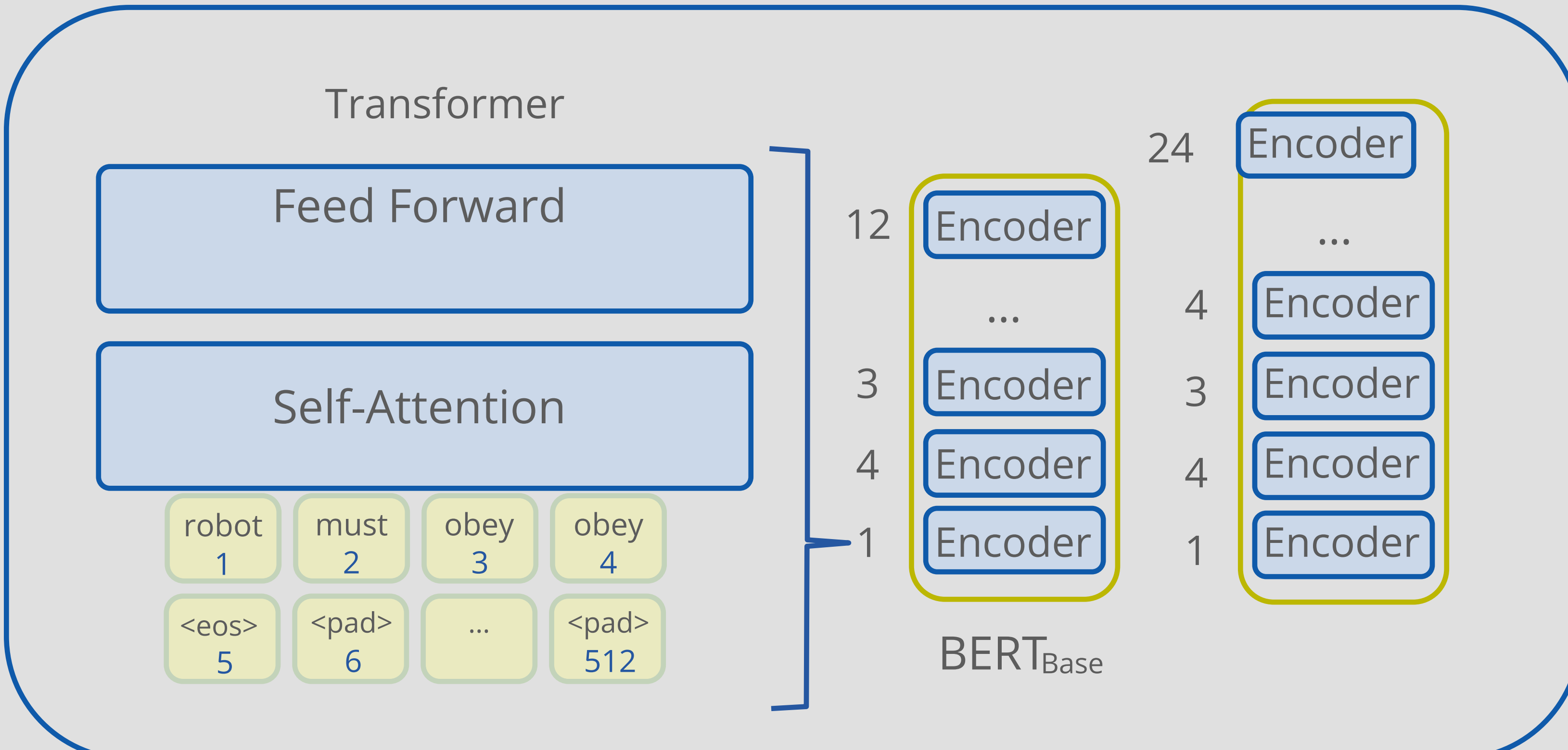
Transformer architecture



- *Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin. "Attention is all you need." NeurIPS, 2017.*

BERT

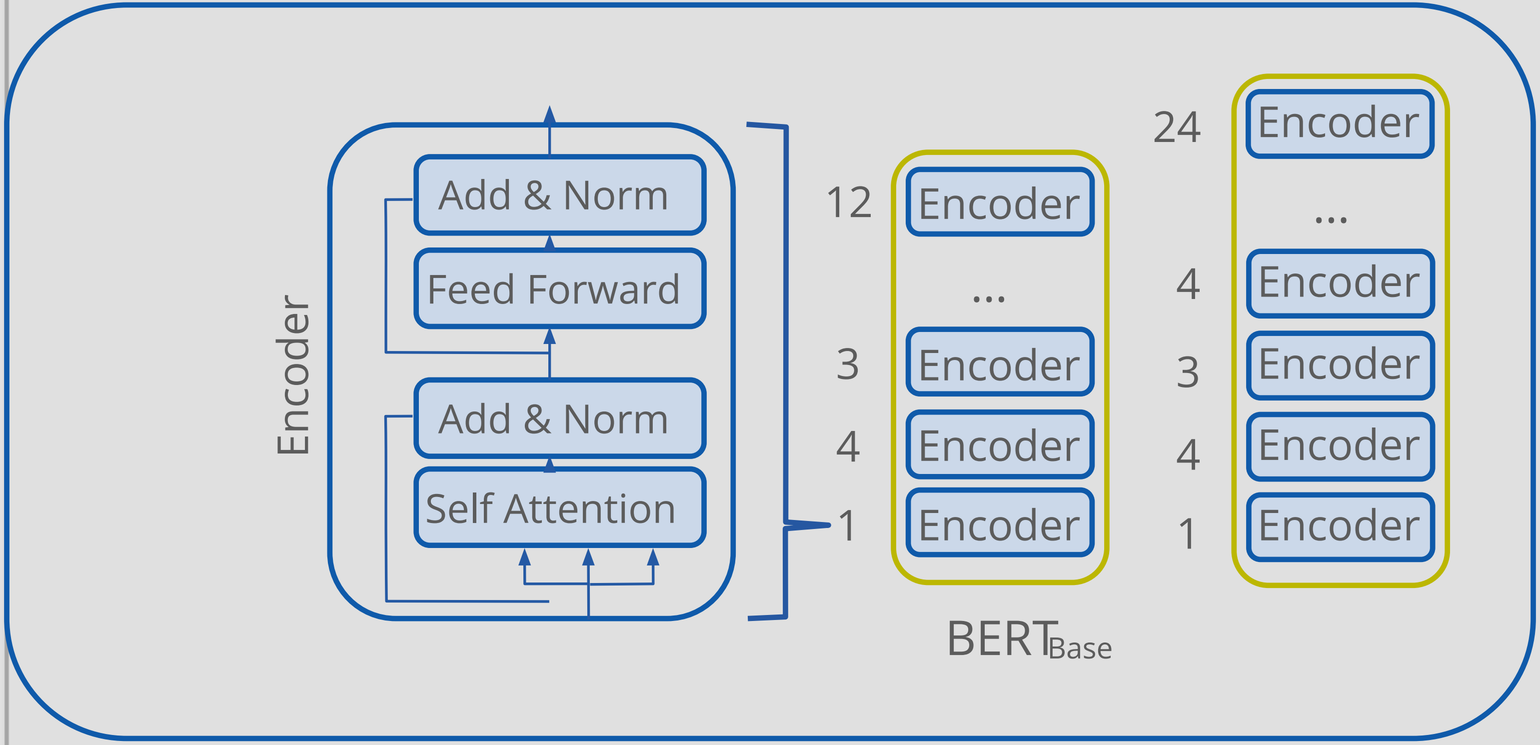
- model is based on encoder
- two configurations: base and large



- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT

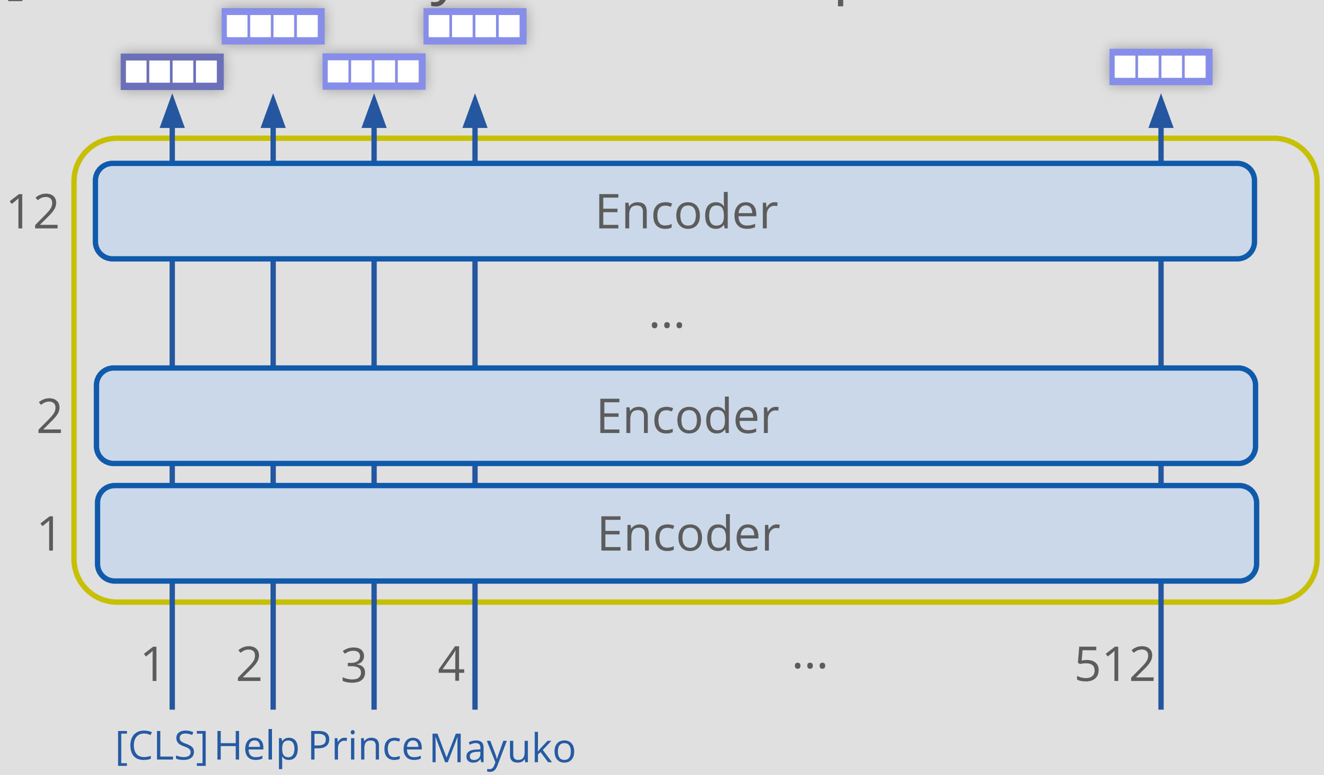
- model is based on encoder
- two configurations: base and large



- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT

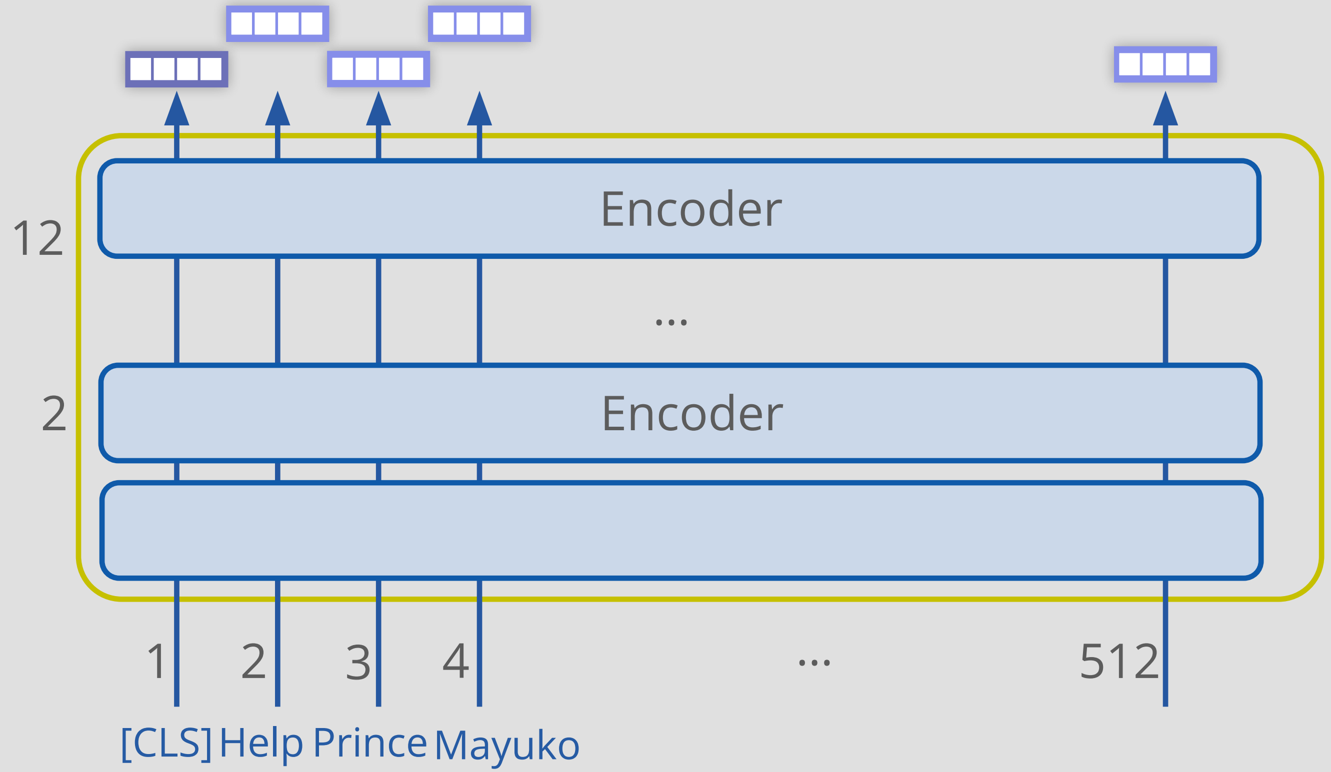
- Two types of tokens: subwords and special tokens
- [CLS] token is always at the first position



- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT

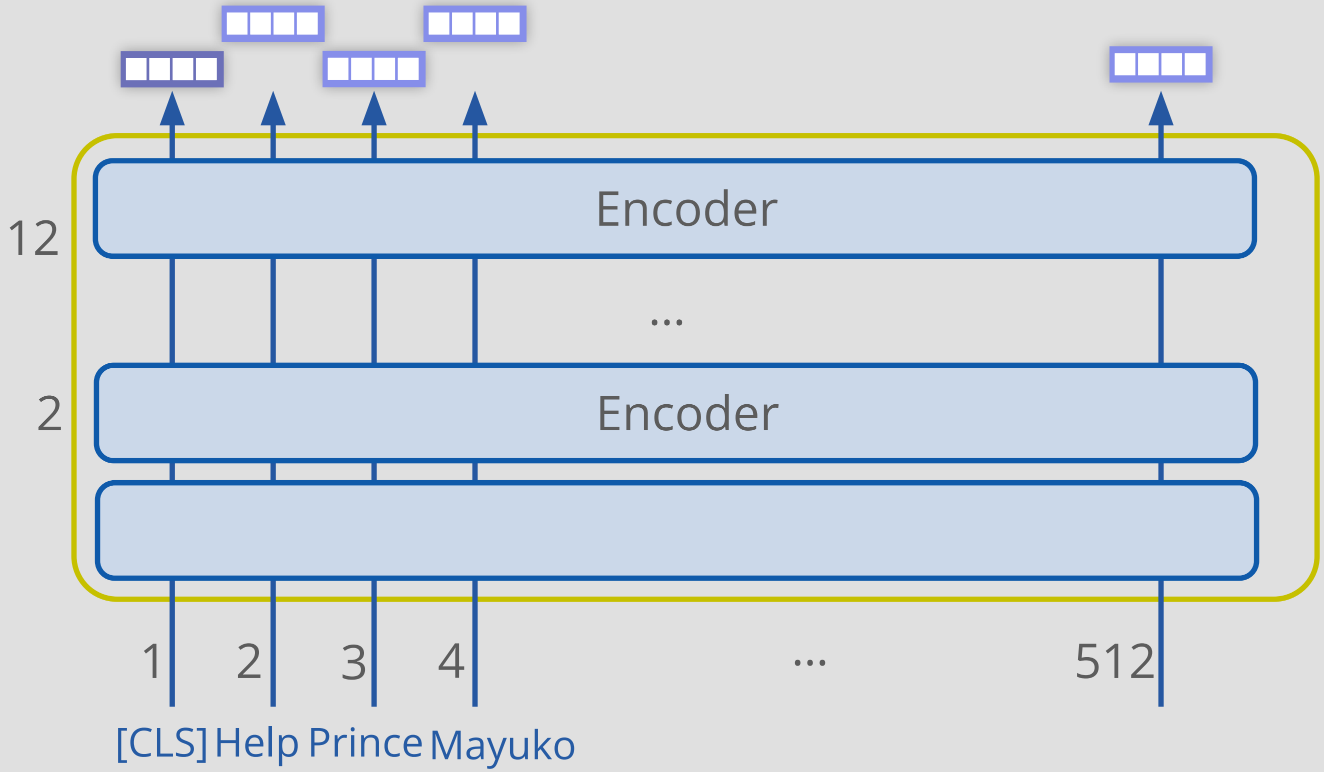
- Вектор токена [CLS] - векторное представление предложения



- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT

- Остальные токены - векторные представления подслов

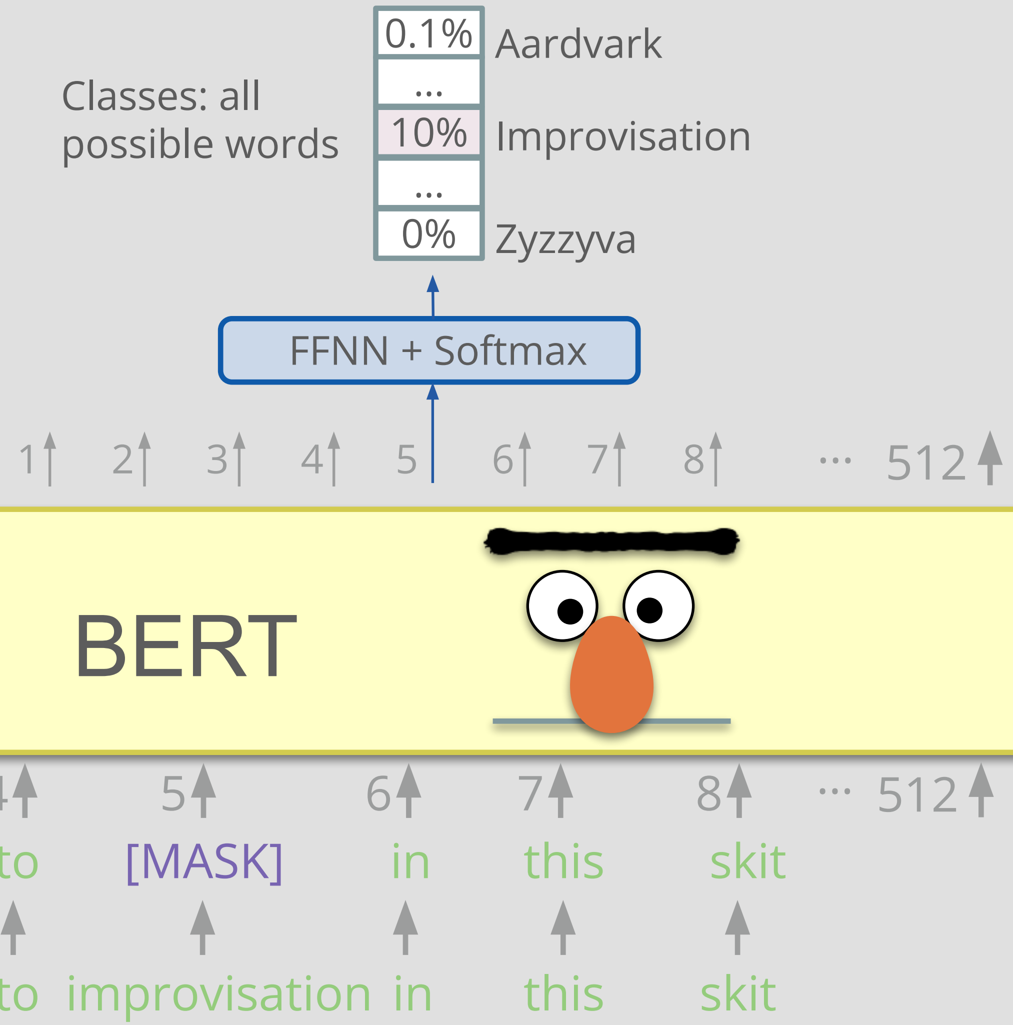


- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT training

1. Masked language model

Predict a token hidden by [MASK]



- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT training

1. Masked language model

15%	80%	my dog is [MASK]
	10%	my dog is apple
	10%	my dog is hairy

- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT training

2. Next sentence prediction

Predict if one sentence follows another, the sentences are connected by the [sep] token

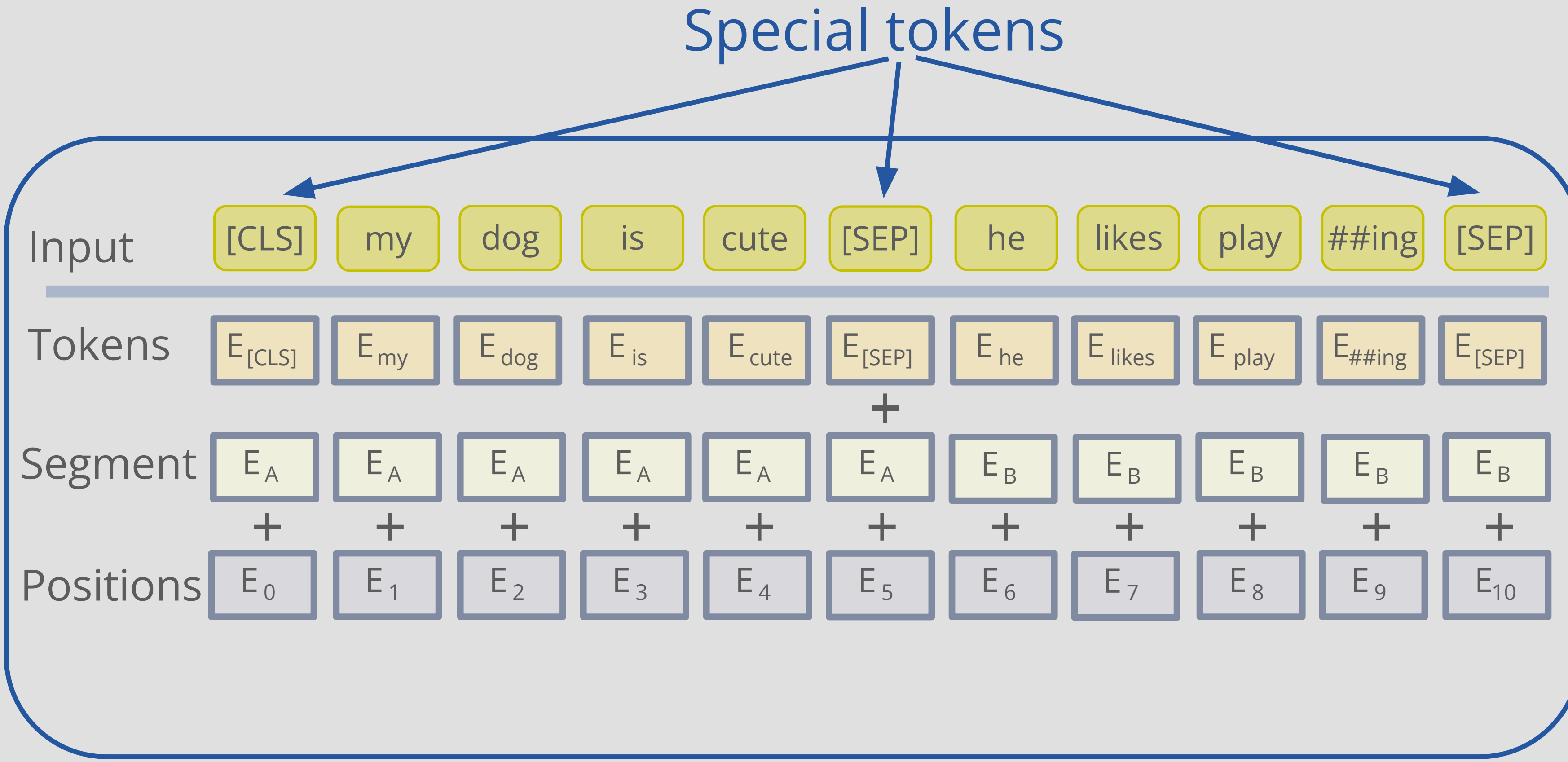
Input	[CLS] the man went to the [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]
Tag	IsNext
Input	[CLS] the man went to the [MASK] store [SEP] penguin [MASK] are flight ##less birds [SEP]
Tag	NotNext

- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT training

Input

Input size: 512

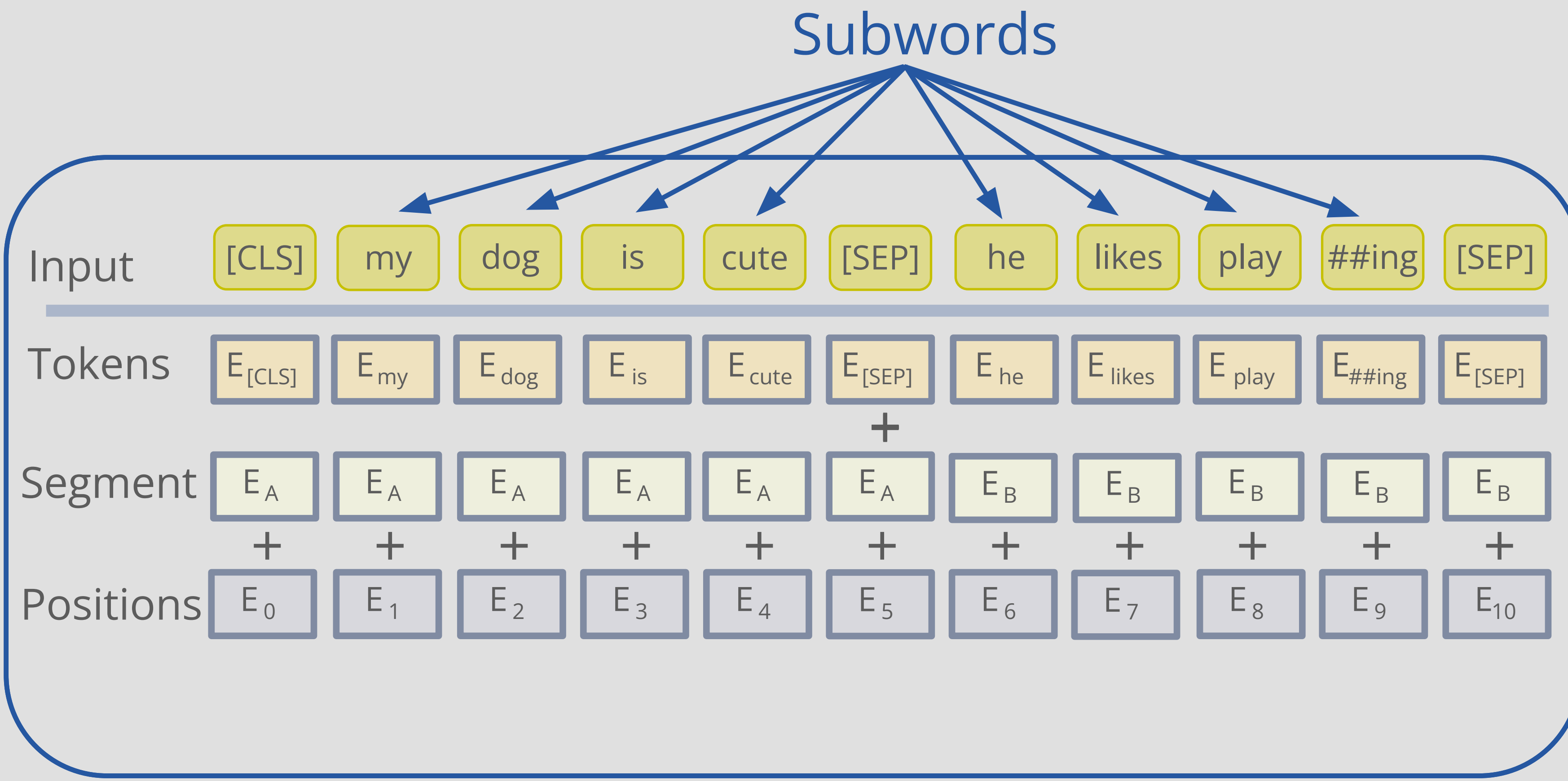


- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT training

Input

Input size: 512

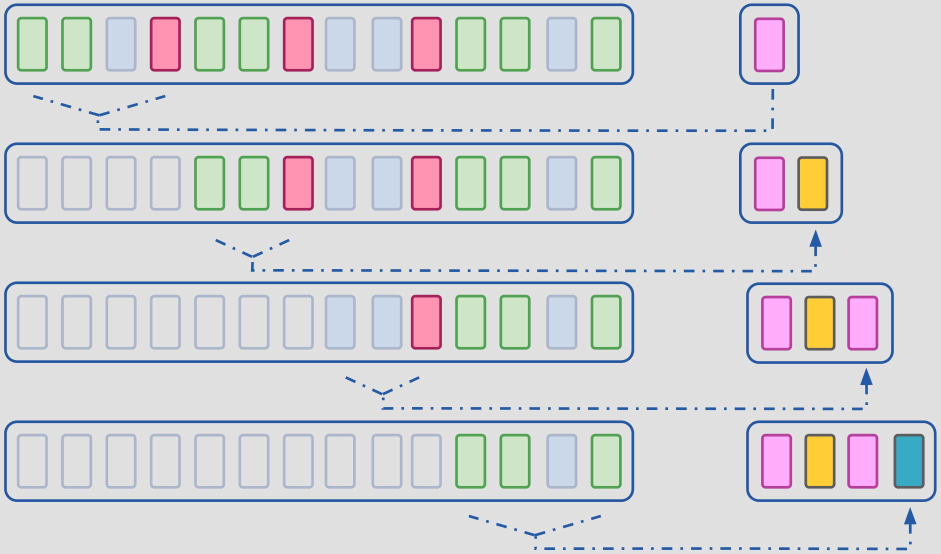


- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

Subwords

Byte pair encoding, BPE

- We count the frequencies of the pairs of symbols
- We glue the most frequent pair of symbols and turn it into a new symbol
- We continue to repeat the operation a fixed number of times



- *Benjamin Heinzerling, Michael Strube. "BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages." LREC, 2018*

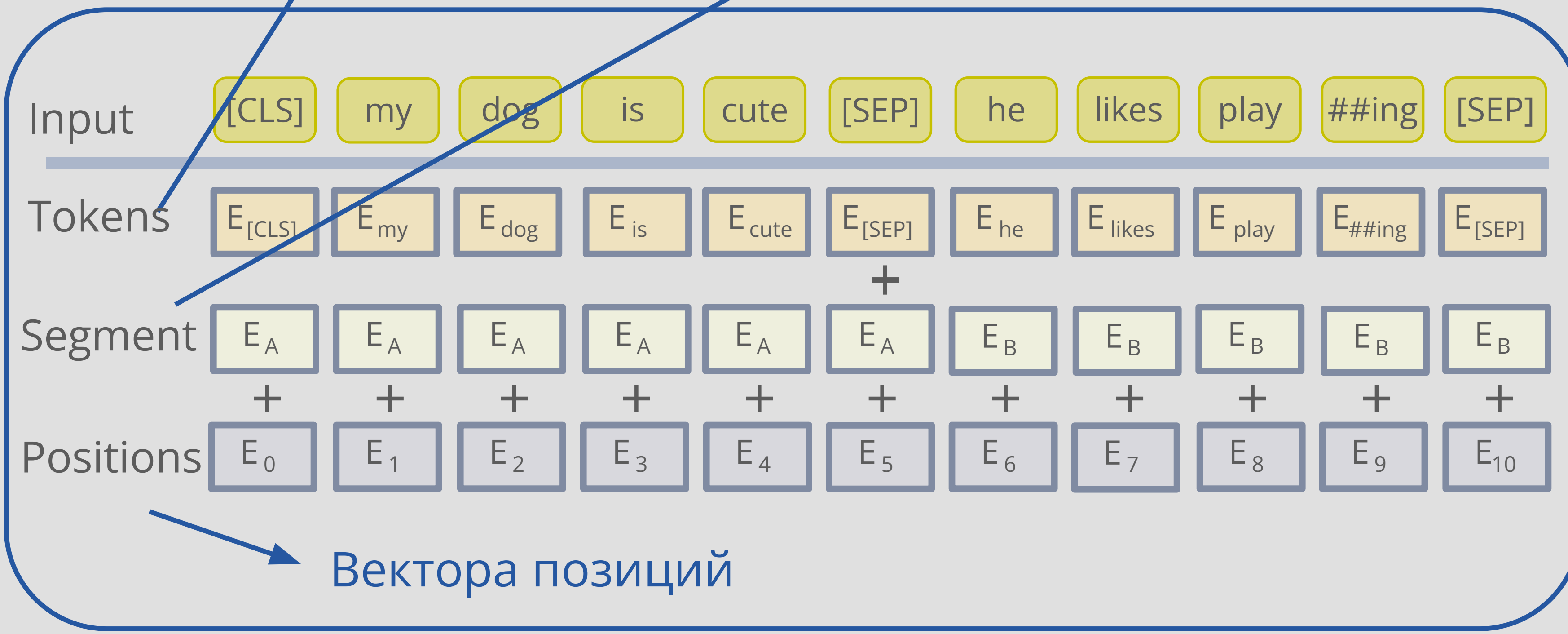
BERT training

Input

Input size: 512

Subword embeddings

Segment embeddings



- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT training

Input

Input size: 512

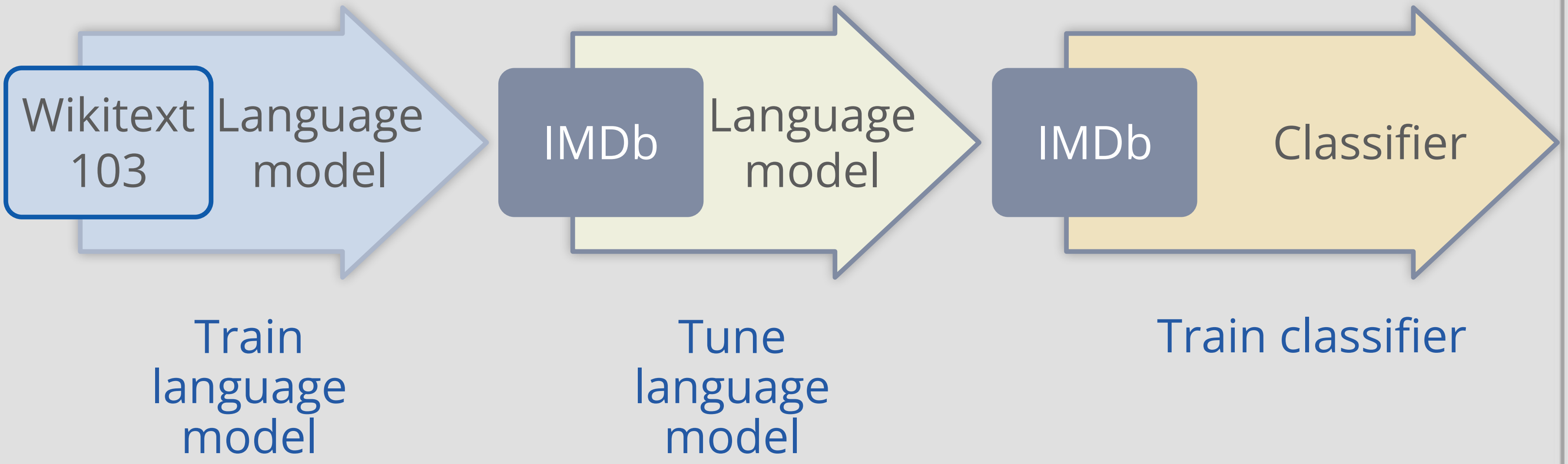
Sentence A | Sentence B

Input	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	##ing	[SEP]
Tokens	$E_{[CLS]}$	E_{my}	E_{dog}	E_{is}	E_{cute}	$E_{[SEP]}$	E_{he}	E_{likes}	E_{play}	$E_{\#\#ing}$	$E_{[SEP]}$
						+					
Segment	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
	+	+	+	+	+	+	+	+	+	+	+
Positions	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

- *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT, 2019*

BERT applications

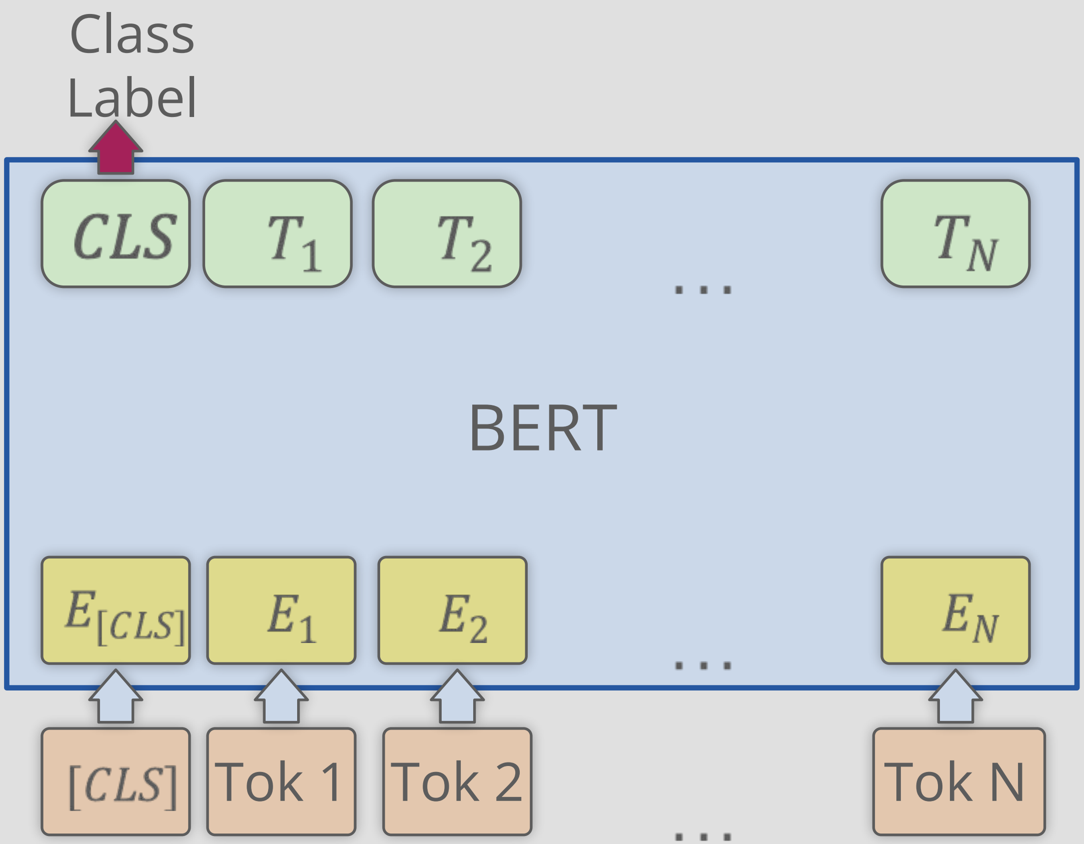
General scheme



BERT applications

Text classification

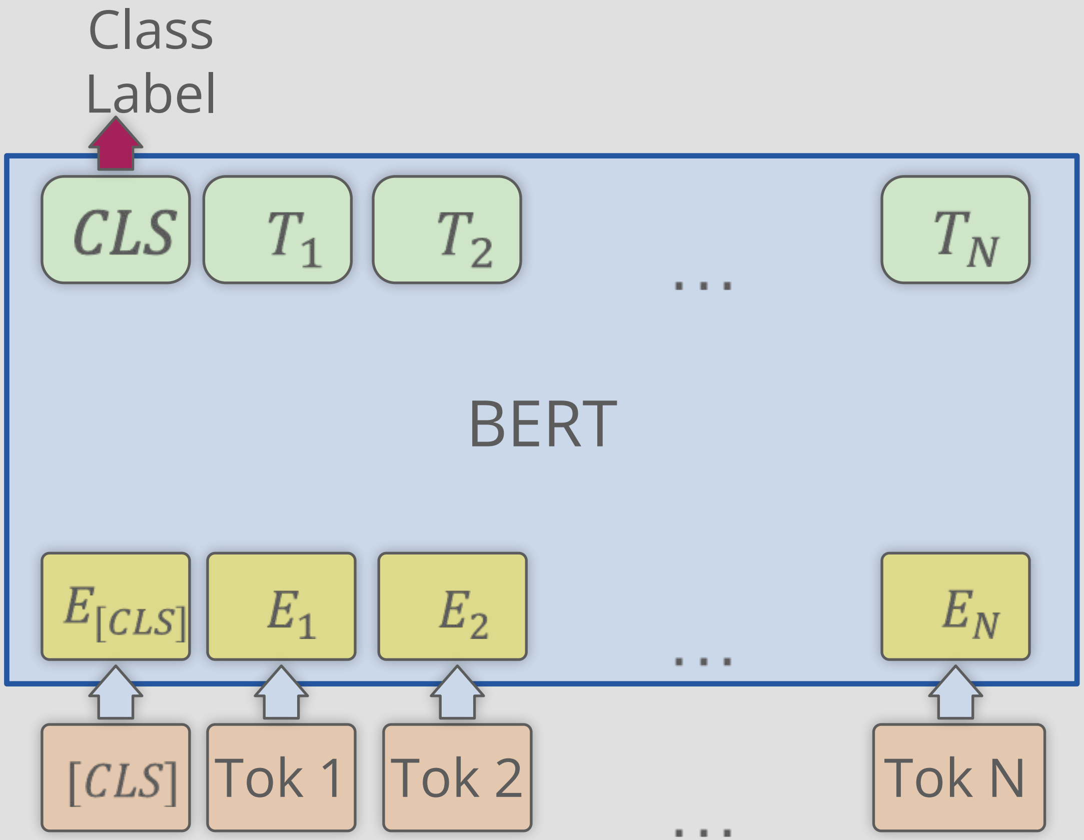
- (Optional) Tune language model



BERT applications

Text classification

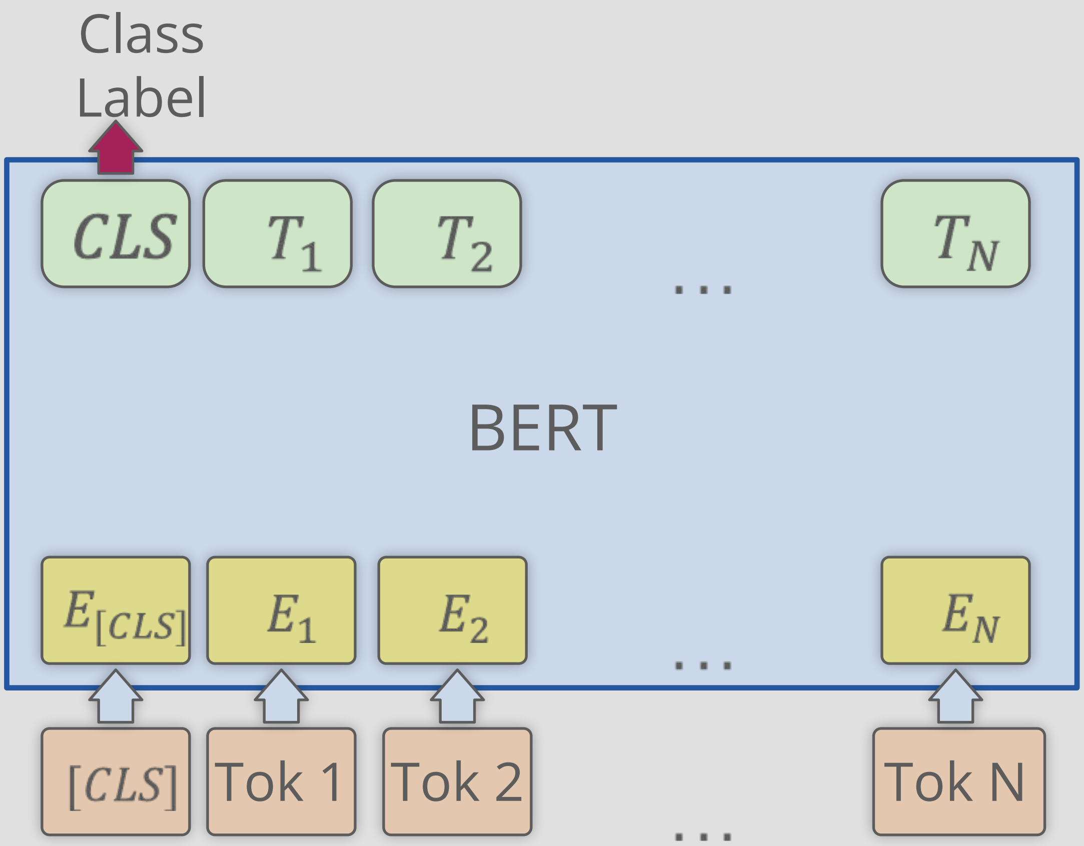
- Add new fully connected layer, layer input - sentence embedding ([CLS] token embedding)



BERT applications

Text classification

- Tune the model or several last layers



Sentence embeddings from BERT model

- [CLS] embedding at the last layer
- [MEAN] word embeddings at the last layer
- [MAX] component-wise maximum of word embeddings at the last layer

Transformer models evaluation

General Natural Language Understanding, GLUE

10 tasks that require a deep understanding of the text

- Sentence classification
  - CoLA: evaluate grammatical correctness
  - SST-2: sentiment prediction
- Classification of a pair of sentences
  - MRPC, QQP, STS-B: find a paraphrase
  - MNLI, QNLI, RTE, WNLI: determine the logical connection between the premise and the conclusion
- Diagnostic test: a wide range of linguistic phenomena
- Comparison with the human evaluation

- *Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. "Glue: A multi-task benchmark and analysis platform for natural language understanding." arXiv preprint arXiv:1804.07461 (2018).*

SuperGLUE

8 tasks that require an even deeper understanding of the text

- Classification of a pair of sentences
  - BoolQ: determine the answer to a binary question by the relevant paragraph
  - CommitmentBank: determine if there is a logical connection between the user's text and the question

- *Wang, Alex, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. "Superglue: A stickier benchmark for general-purpose language understanding systems." In Advances in Neural Information Processing Systems, pp. 3261-3275. 2019.*

SuperGLUE

8 tasks that require an even deeper understanding of the text

- Classification of a pair of sentences
  - WiC: determine if a polysemantic word is used in the same sense in given contexts
  - COPA: choose a conclusion from two options for a given hypothesis

- *Wang, Alex, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. "Superglue: A stickier benchmark for general-purpose language understanding systems." In Advances in Neural Information Processing Systems, pp. 3261-3275. 2019.*

SuperGLUE

8 tasks that require an even deeper understanding of the text

- Reading comprehension
  - MultiRC, ReCoRD: paragraph and multiple choice question
- Intersection with GLUE: WNLI, RTE

- *Wang, Alex, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. "Superglue: A stickier benchmark for general-purpose language understanding systems." In Advances in Neural Information Processing Systems, pp. 3261-3275. 2019.*

Leaderbords

- Leaderboards GLUE, SuperGlue, DecaNLP allow to compare language models
- More parameters and data - the language model is higher in the leaderboard
- Costs for speed, memory and resources are not taken into account
- A high rating of the language model does not mean good results in practice

Evolution of language models