SkillFactory

Transformer BERT

Sidorov Nikita

MLE (NLP) Sber

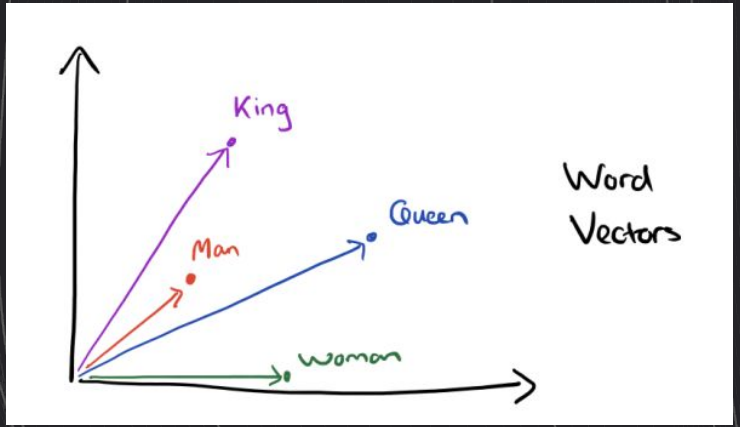
What we will learn today

- word embeddings;
- contextualized word embeddings;
- multi-head attention
- positional encoding
- Transformer model
- BERT overview
- GPT overview
- transfer learning task

Word representation

Previously we have one representation for each word

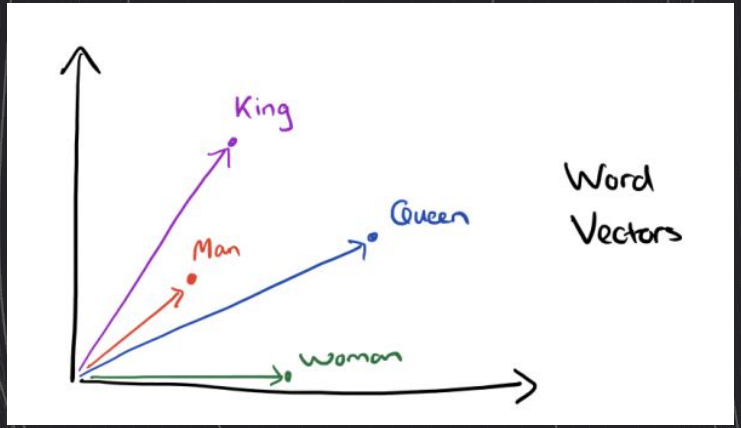
What the problem with it?



Word representation

Previously we have one representation for each word

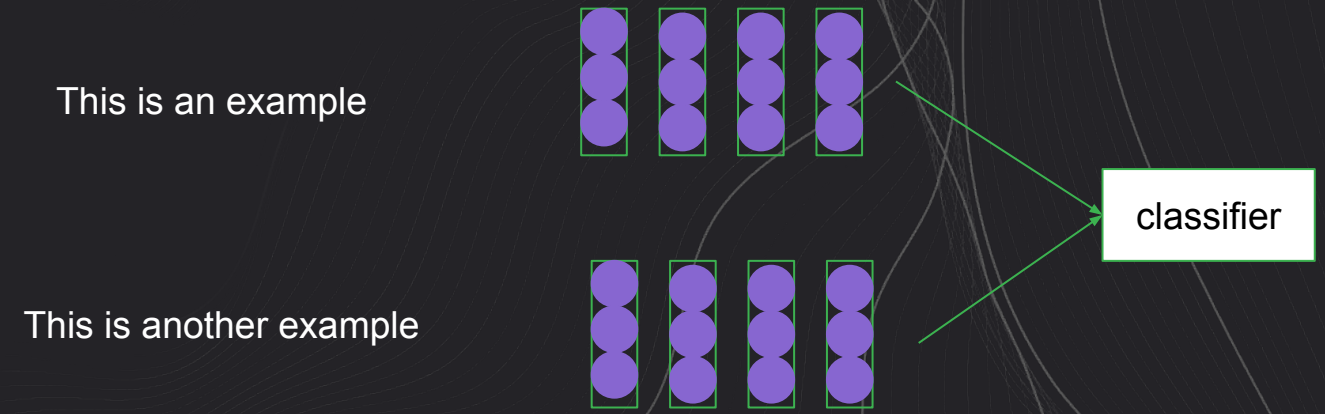
What the problem with it?



- Always the same representation for a word type regardless of the context in which a word token occurs.
- we just have one representation for a word, but word have different aspects, including semantics behavior, and register/connotations.

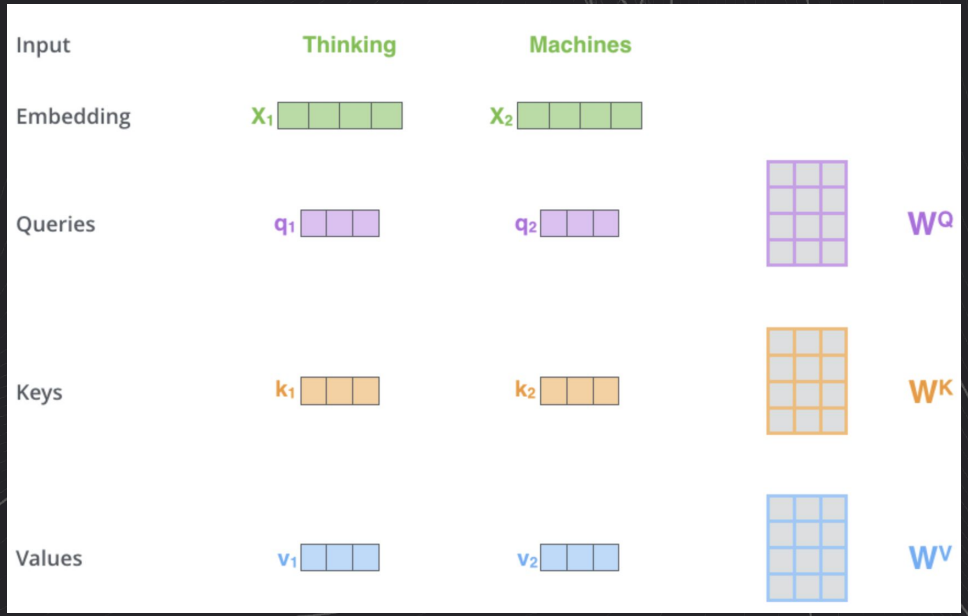
Contextual word representation

Instead of one vector per sentence we can get one vector per word.



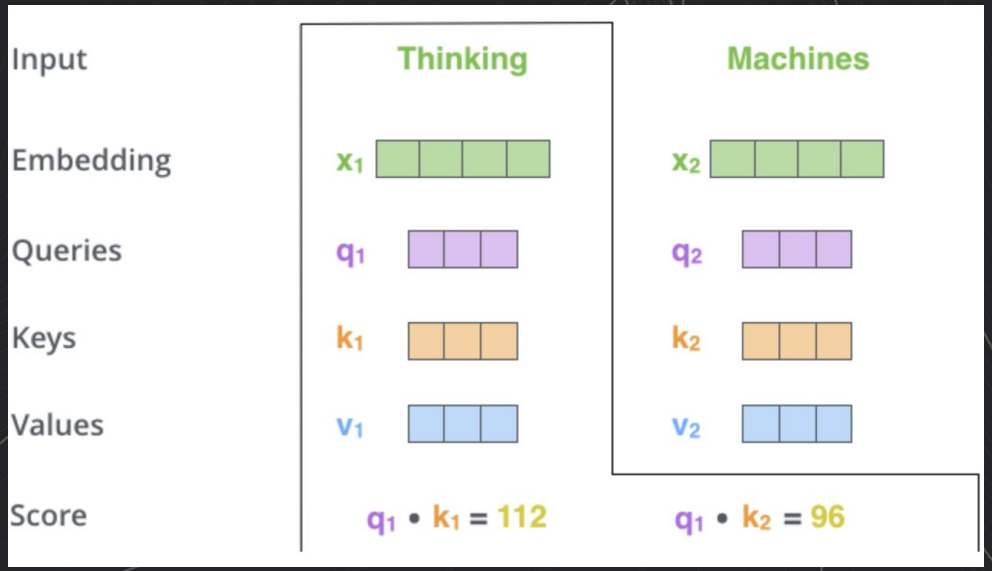
Self-Attention in detail

Step 1. Get three representations of each word vector (Query, Key, Value) by multiplying on corresponding matrix. Dimension of each vector will be 64.



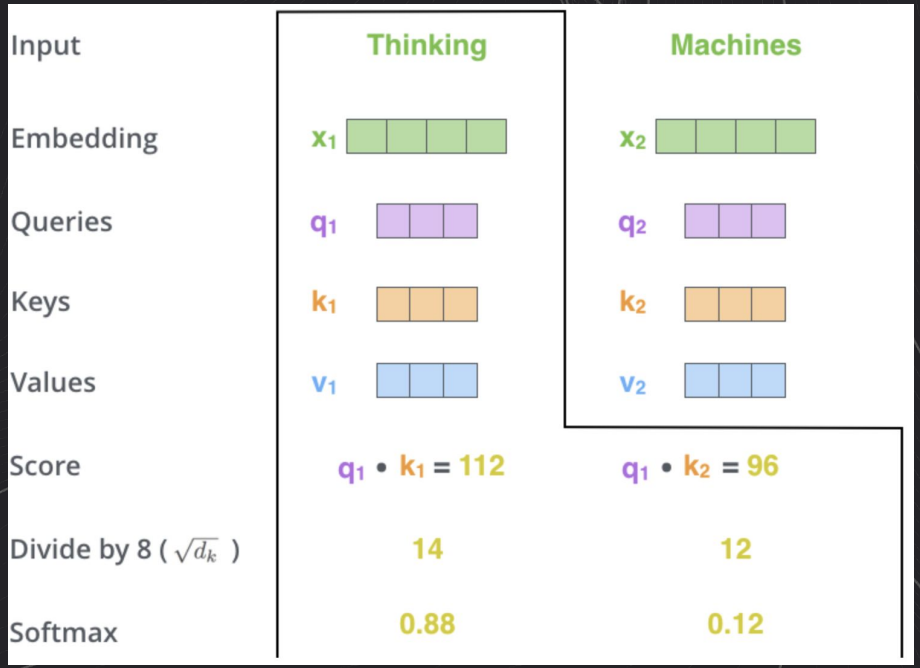
Self-Attention in detail

Step 2. Now let's calculate a score. Say we're calculating the self-attention for the first word in this example, "Thinking". We need to score each word of the input sentence against this word. The score is calculated by taking the dot product of the query vector with the key vector of the respective word we're scoring.



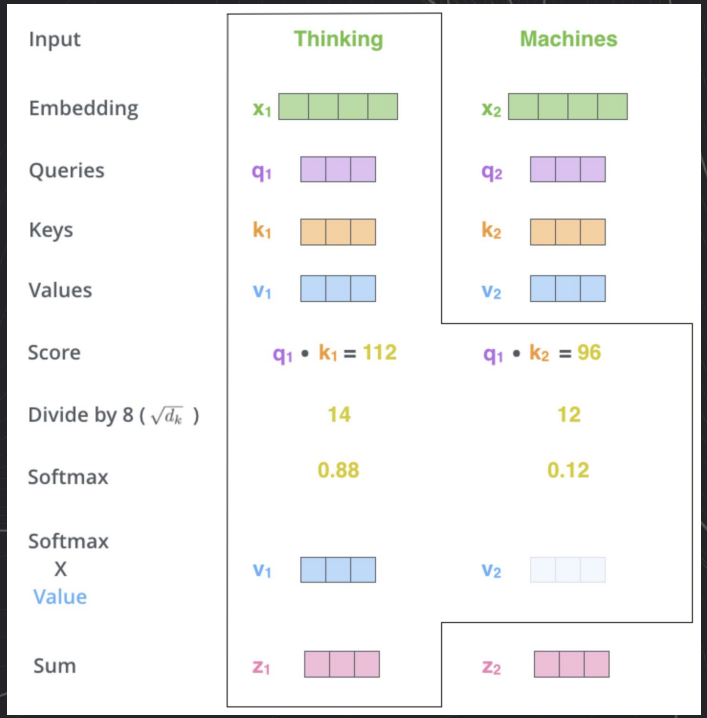
Self-Attention in detail

Step 3 and 4. Divide the scores by the square root of the dimension of the key vectors. Then pass the result through a softmax operation. Softmax normalizes the scores so they're all positive and add up to 1.



Self-Attention in detail

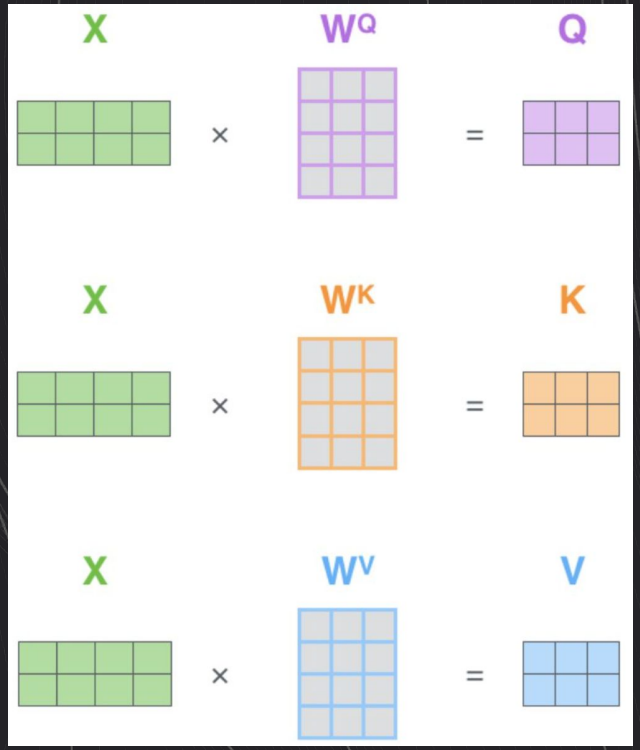
Step 5 and 6. Multiply each value vector by the softmax score. Then sum up the weighted value vectors.



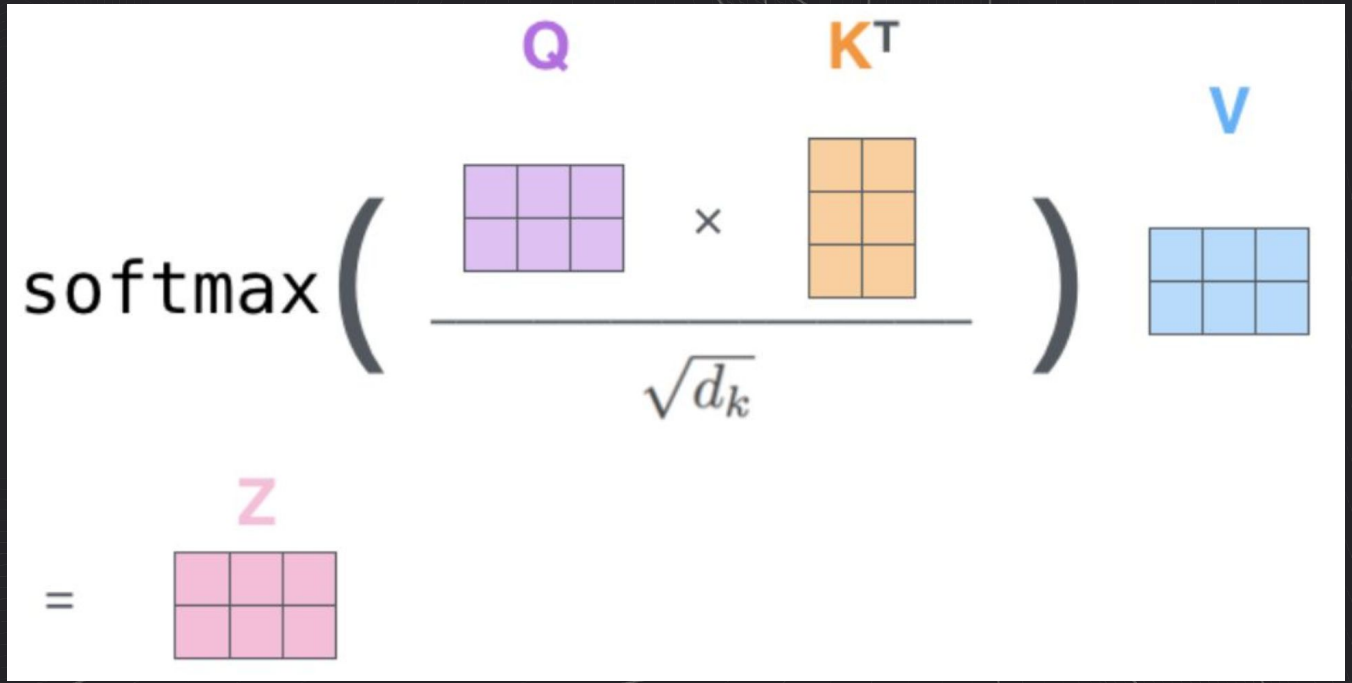
Self-Attention matrix multiplication

Pick embeddings into matrix X

Multiply X by weight matrices we've trained (W^q, W^k, W^v)



Self-Attention matrix multiplication



Multi-head Attention

Problem with simple self-attention:

- one way for words to interact with each other.

Multi-head Attention

Problem with simple self-attention:

- one way for words to interact with each other.

Solution is Multi-head attention.

Multi-head Attention

Problem with simple self-attention:

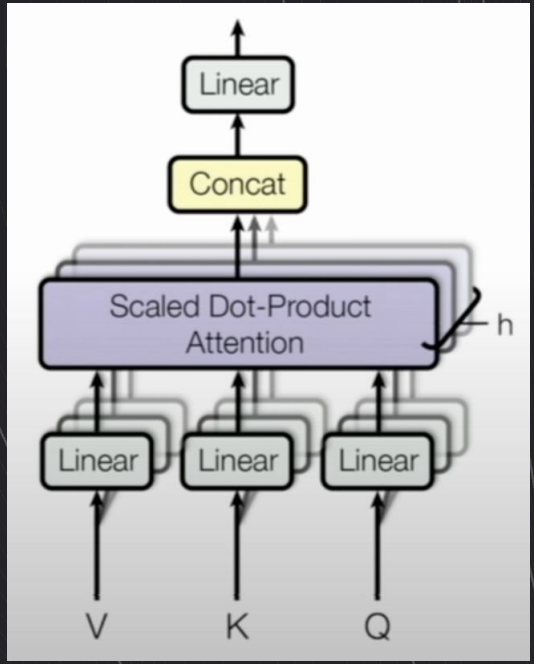
- one way for words to interact with each other.

Solution is Multi-head attention.

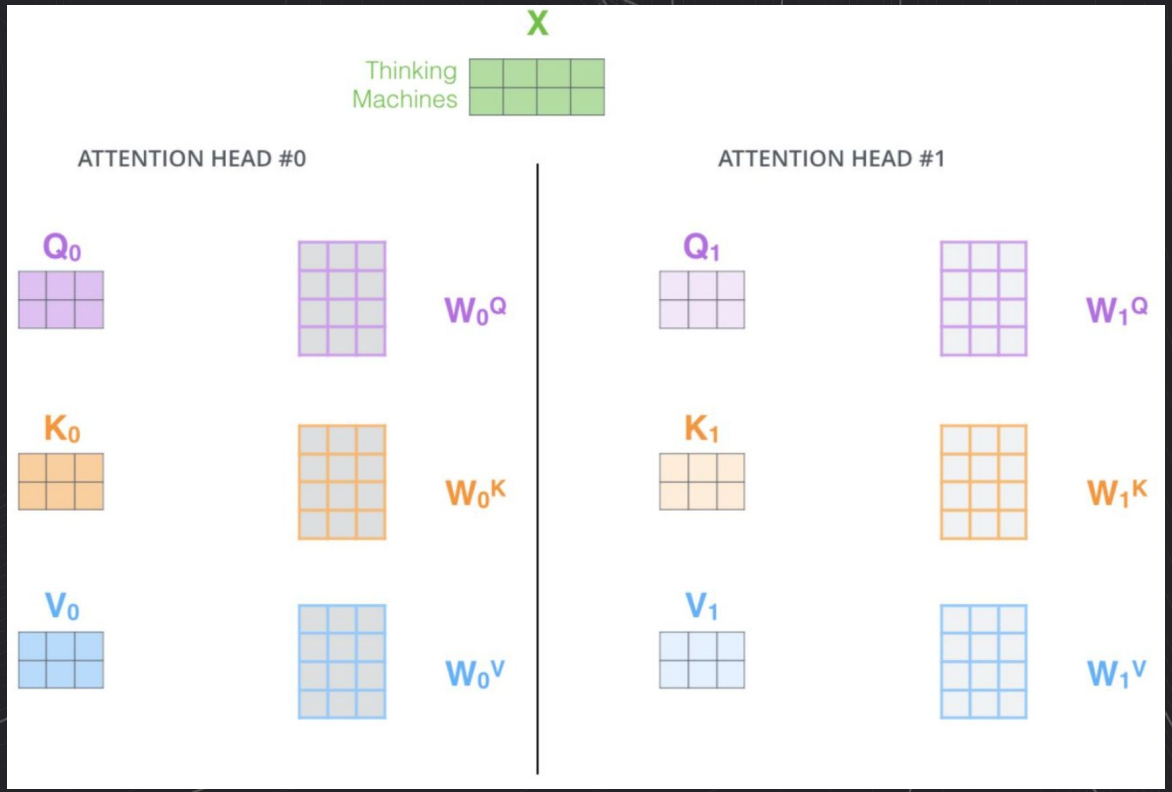
First map Q, K, V into h=8 many lower dimensional spaces via W matrices.

Then apply attention, and after that concatenate each representation and pipe through linear layer.

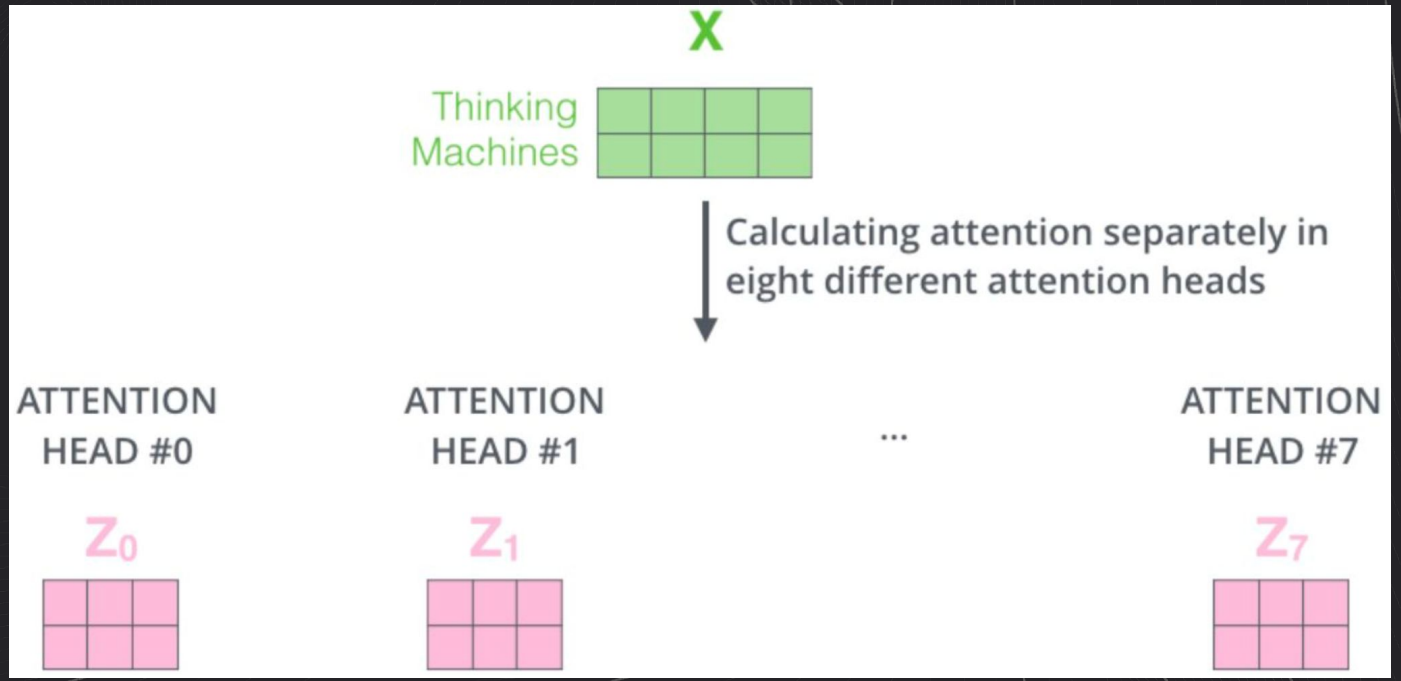
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



Multi-head Attention



Multi-head Attention

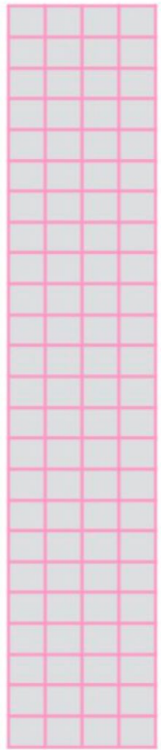


Multi-head Attention

1) Concatenate all the attention heads

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

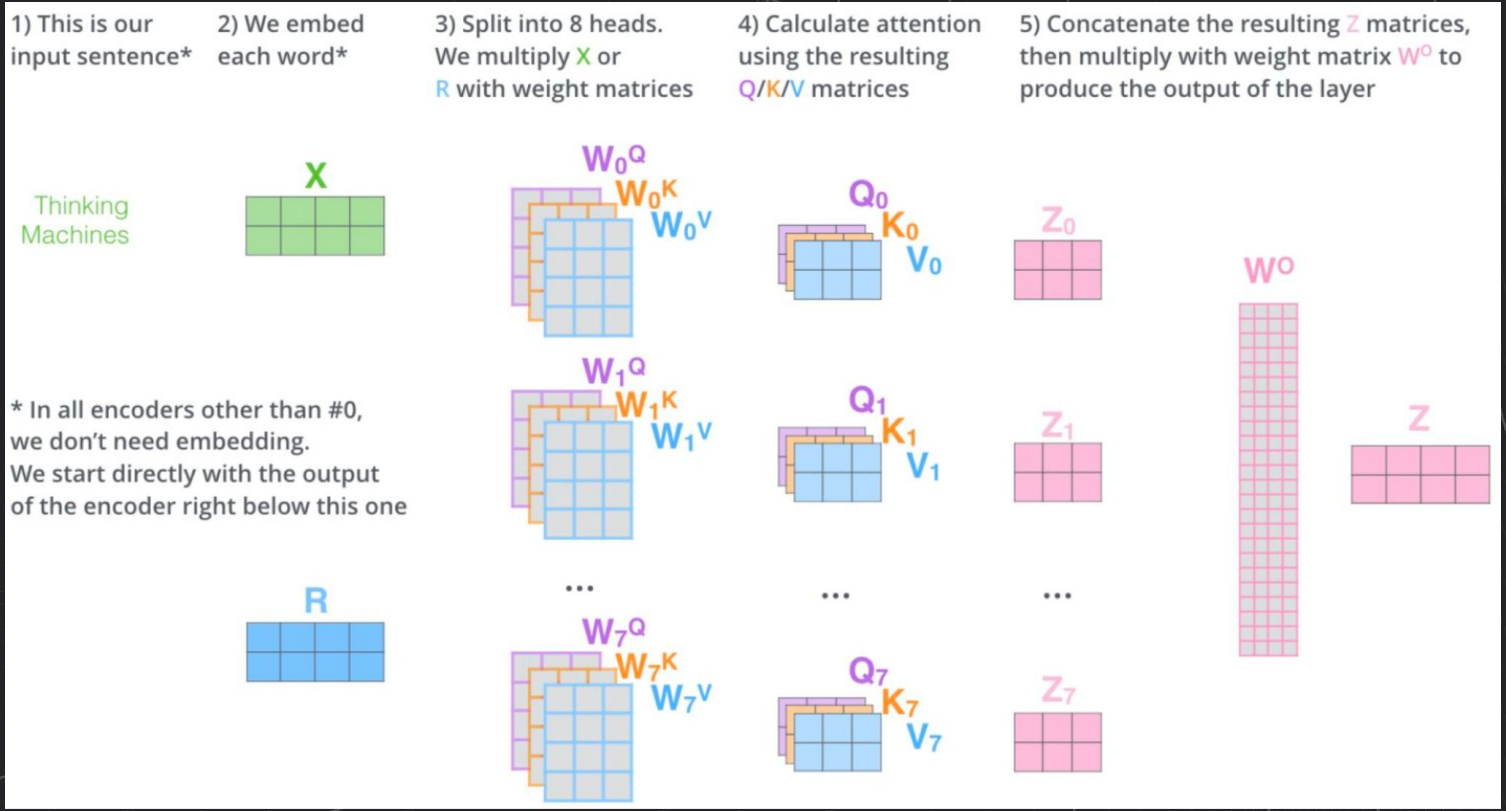
X



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-head Attention



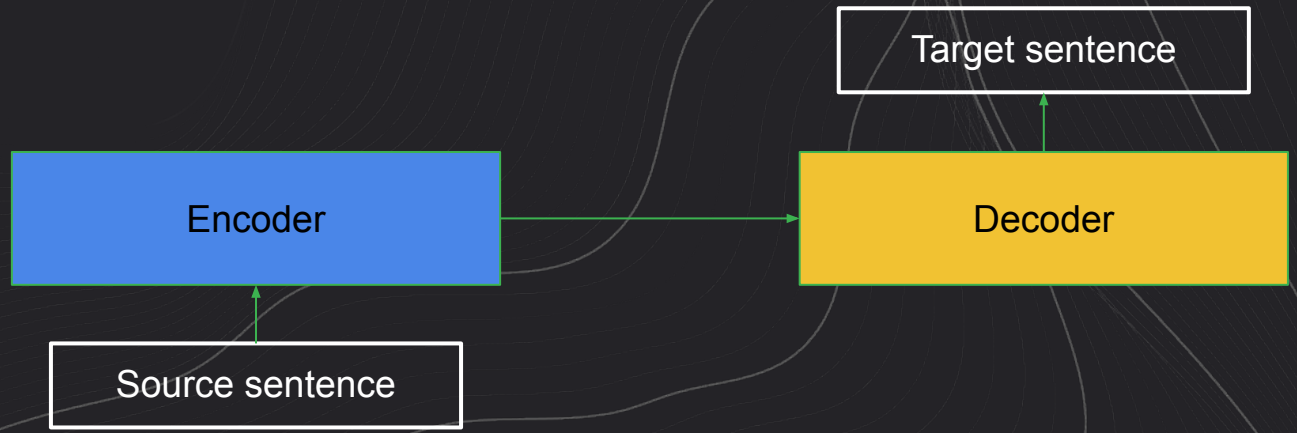
Encoder-Decoder Framework

	Seq2Seq without attention	Seq2Seq with attention
processing within encoder	RNN/CNN	RNN/CNN
processing within decoder	RNN/CNN	RNN/CNN
encoder-decoder interaction	static fixed-sized vector	attention

Encoder-Decoder Framework

The standard modeling paradigm:

- Encoder - reads the source sentence and builds its representation
- Decoder - uses source representation from the encoder to generate the target sequence



Attention is all you need

	Seq2Seq without attention	Seq2Seq with attention	Transformer
processing within encoder	RNN/CNN	RNN/CNN	attention
processing within decoder	RNN/CNN	RNN/CNN	attention
encoder-decoder interaction	static fixed-sized vector	attention	attention

Why it is better then RNNs?

I arrived at the bank after crossing the …

What does **bank** means in this sentence?

Why it is better then RNNs?

I arrived at the bank after crossing the …

What does **bank** means in this sentence?

RNN need to read until the end of the sentence. O(N) steps to process sentence.

Why it is better then RNNs?

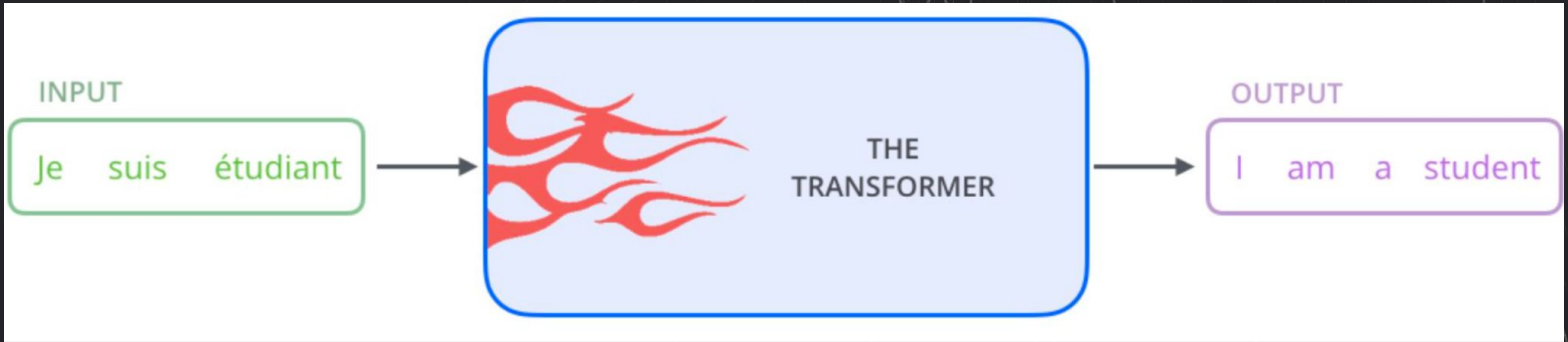
I arrived at the bank after crossing the …

What does **bank** means in this sentence?

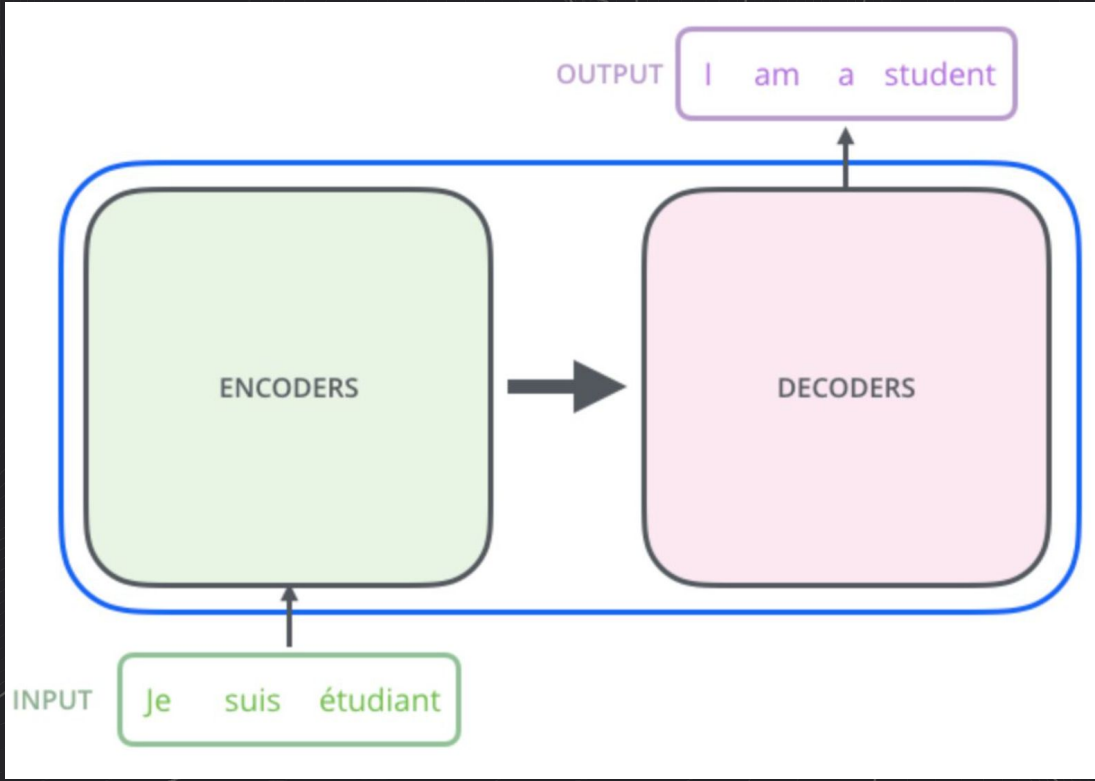
RNN need to read until the end of the sentence. O(N) steps to process sentence.

Self-attention see all words at once. It takes constant number of steps to process any sentence.

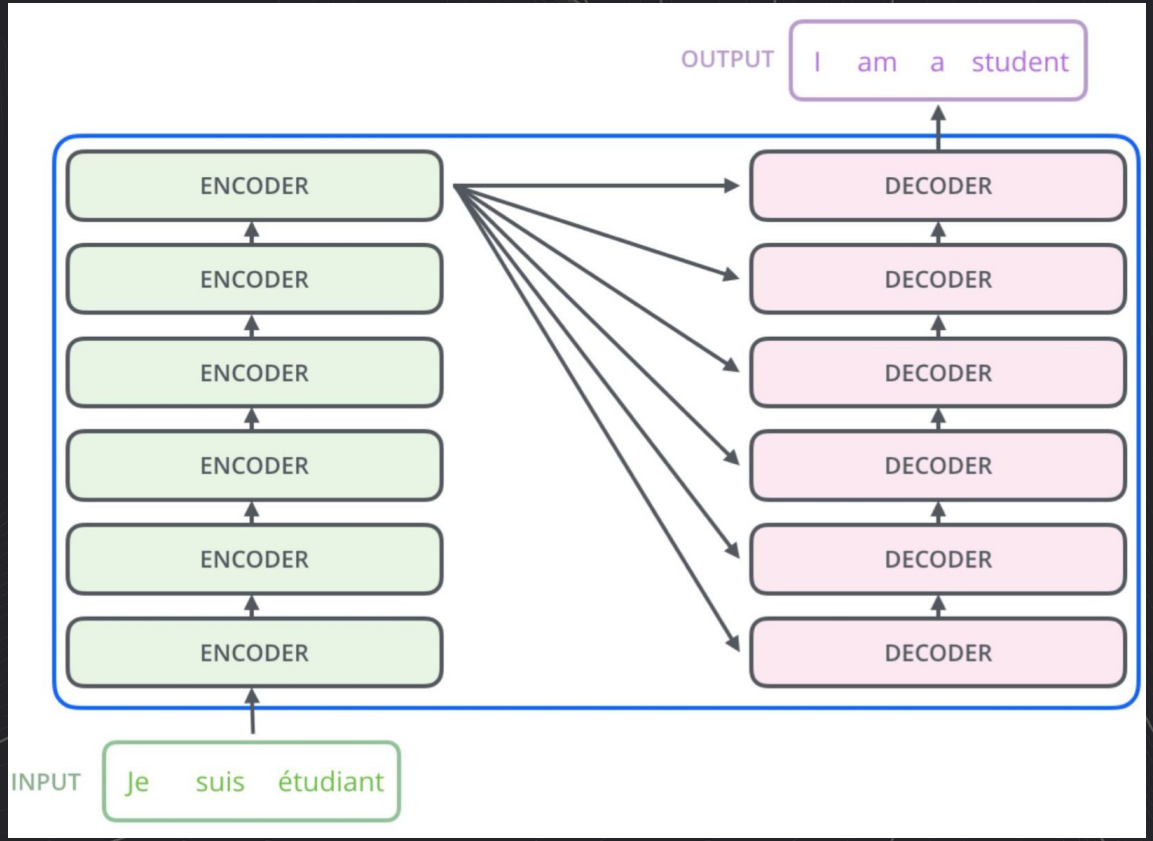
Transformer



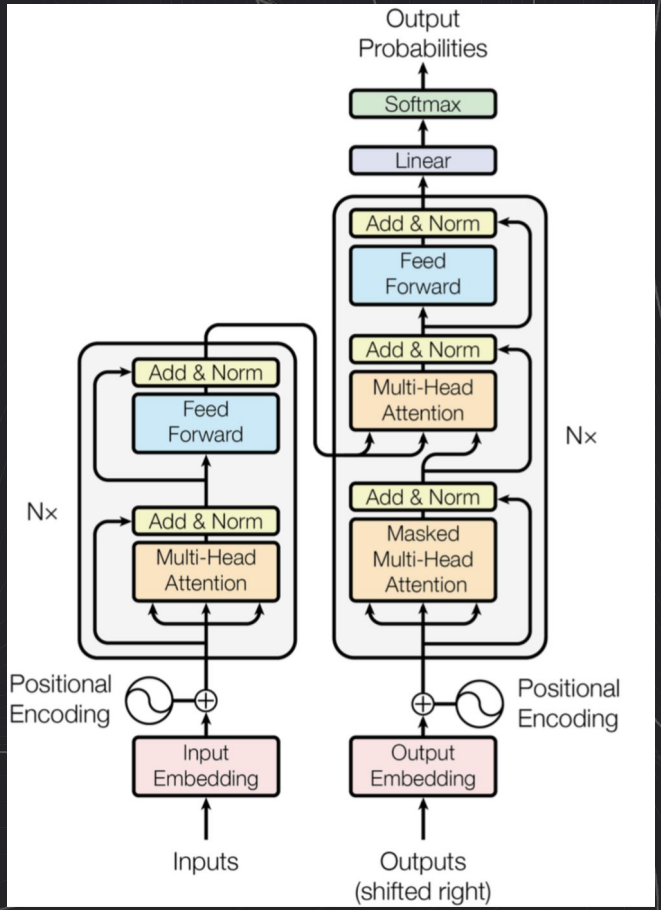
Transformer



Transformer



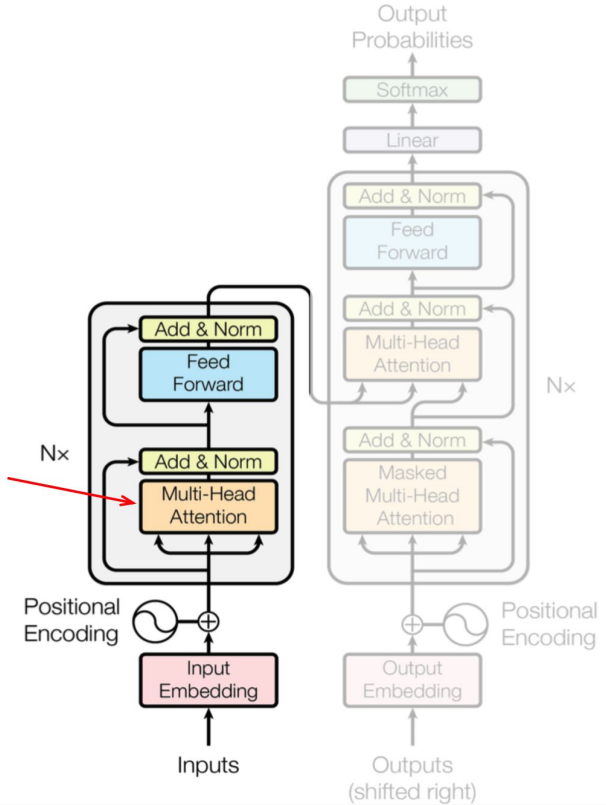
Transformer



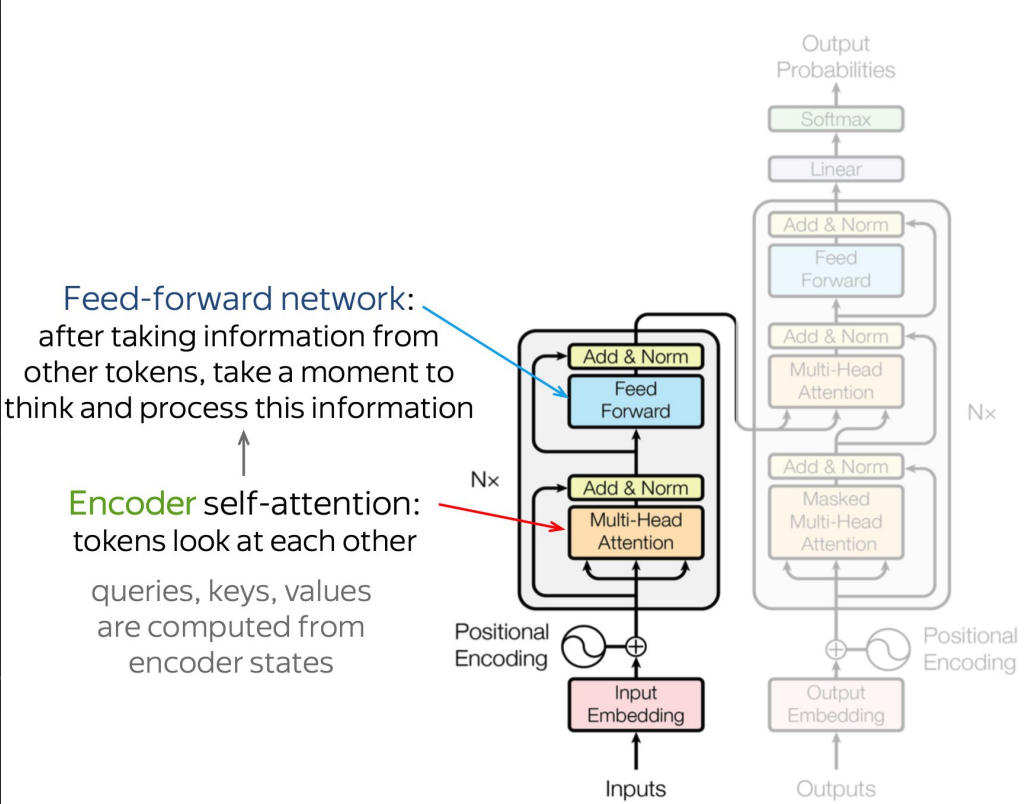
Transformer

Encoder self-attention:
tokens look at each other

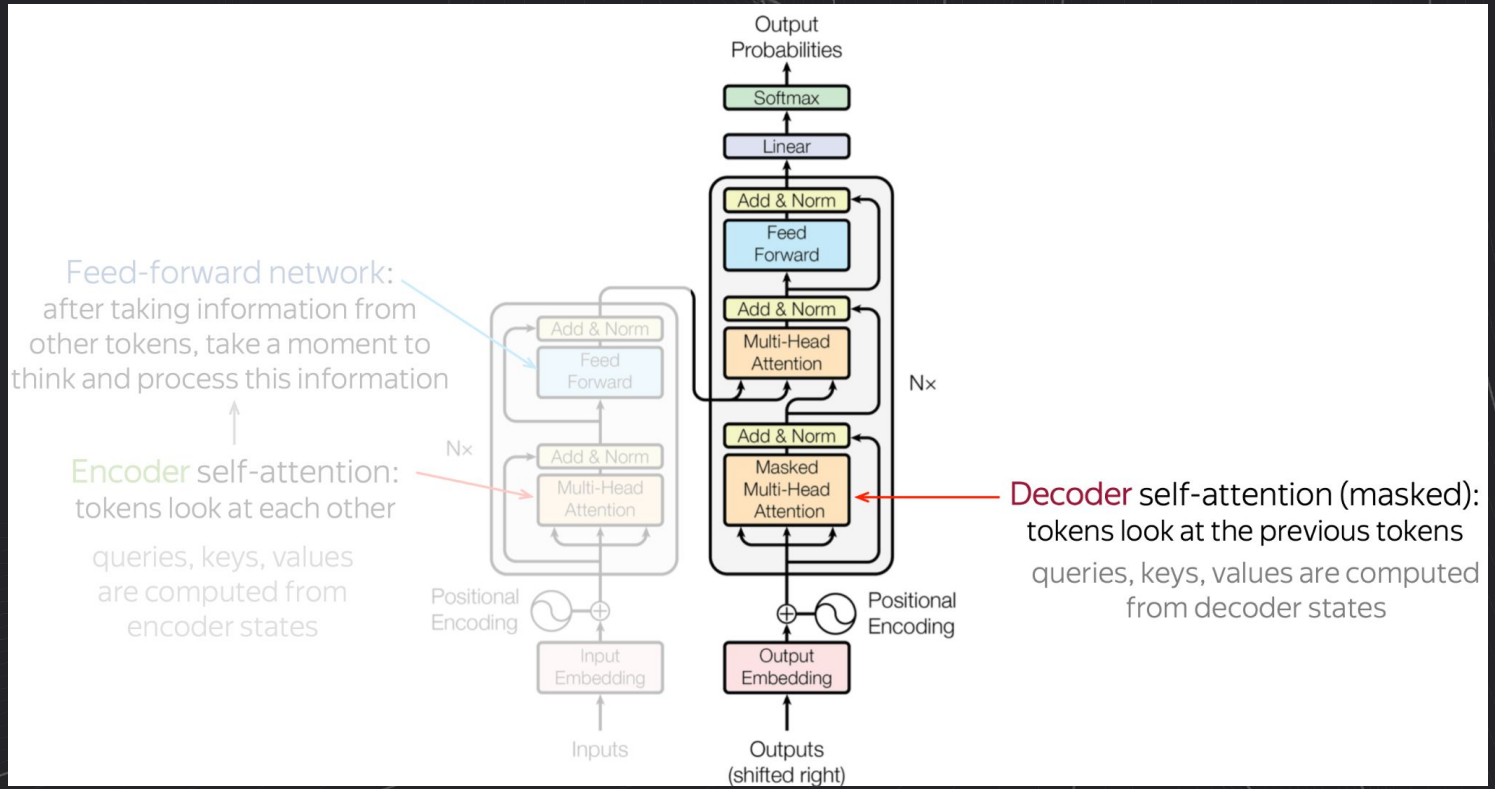
queries, keys, values are computed from encoder states



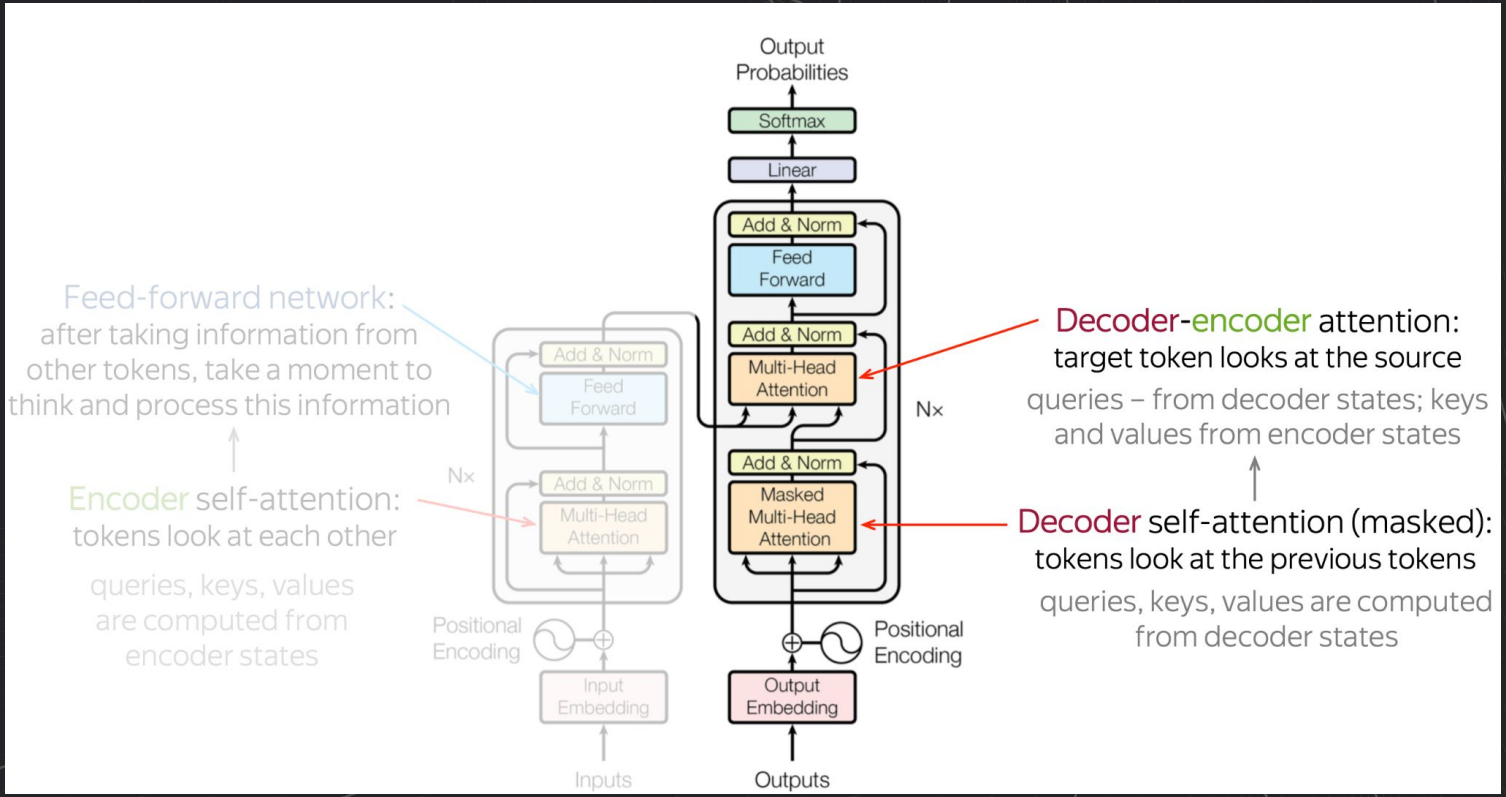
Transformer



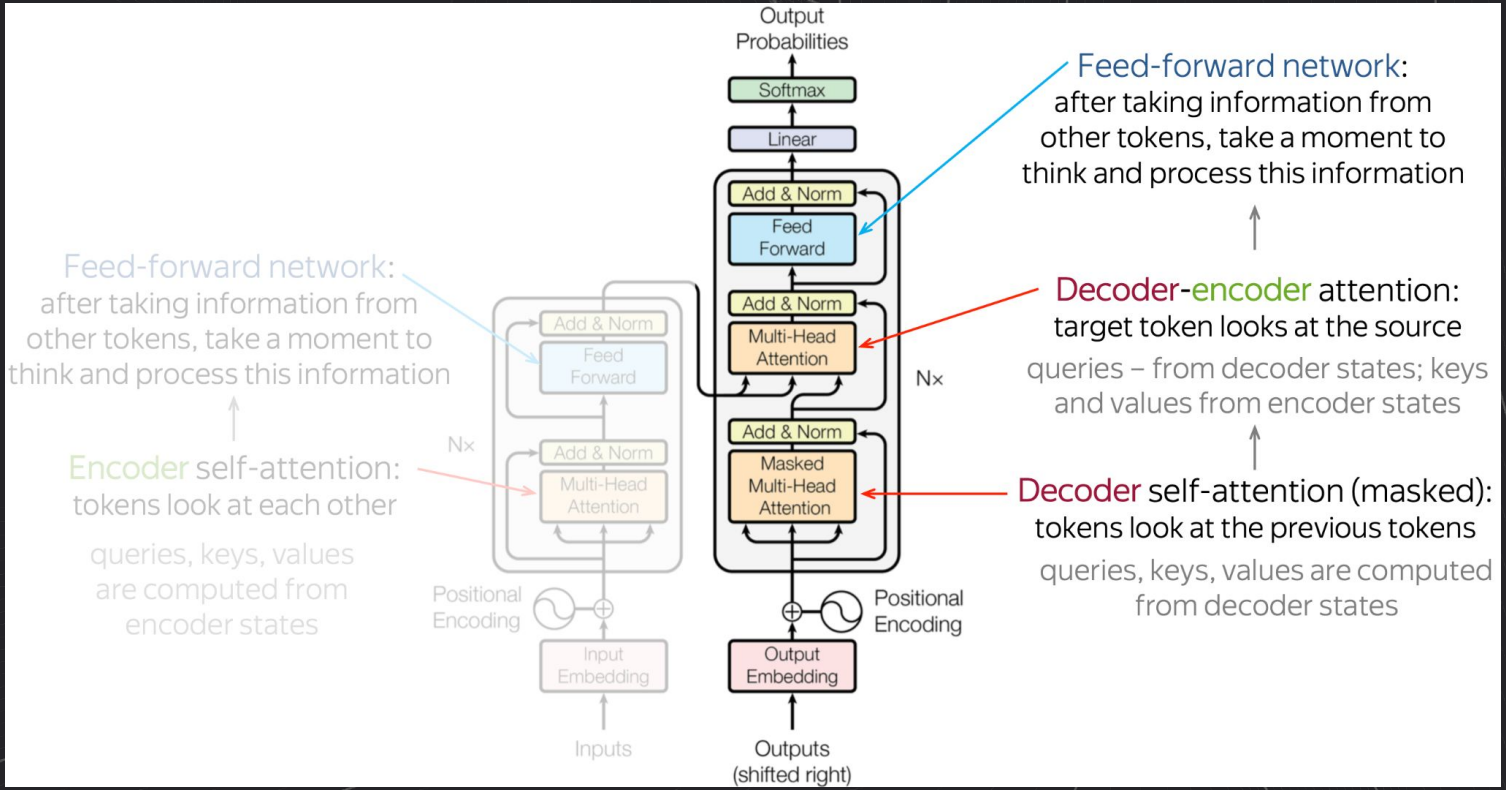
Transformer



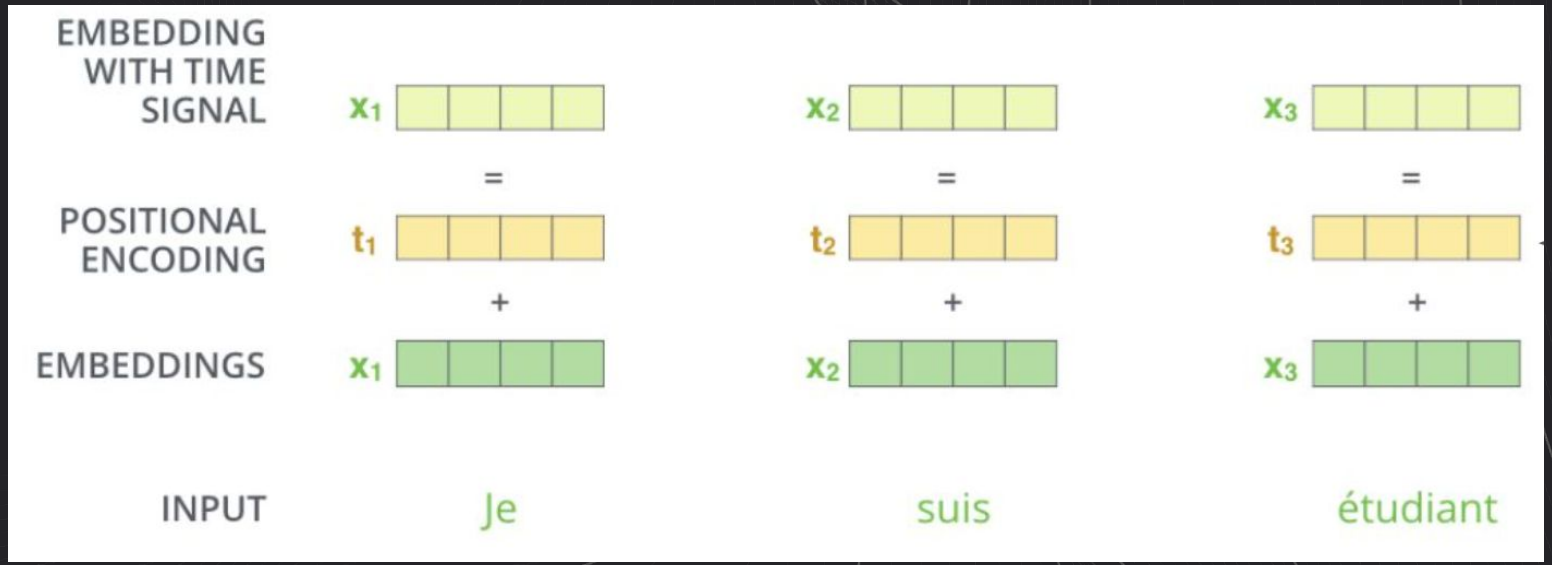
Transformer



Transformer

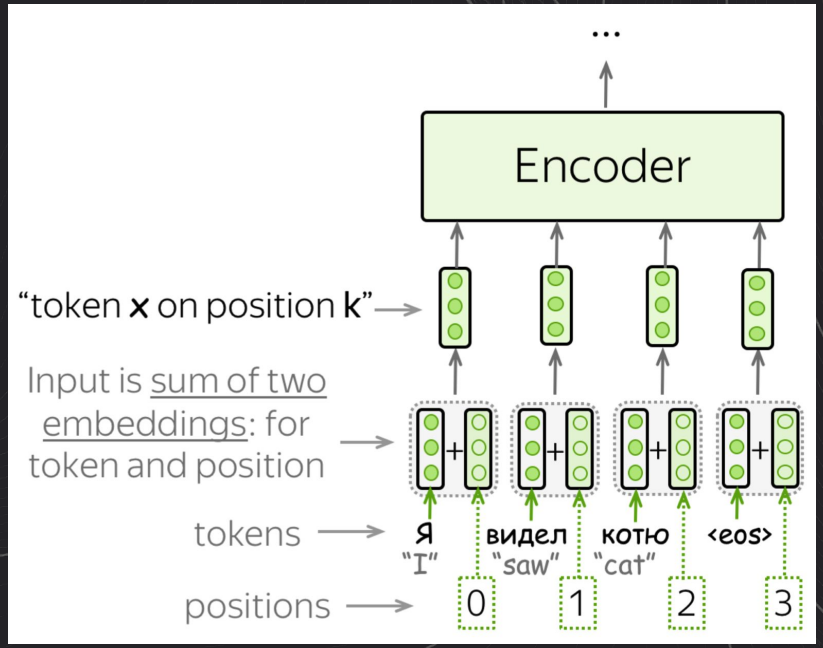


Positional Encoding

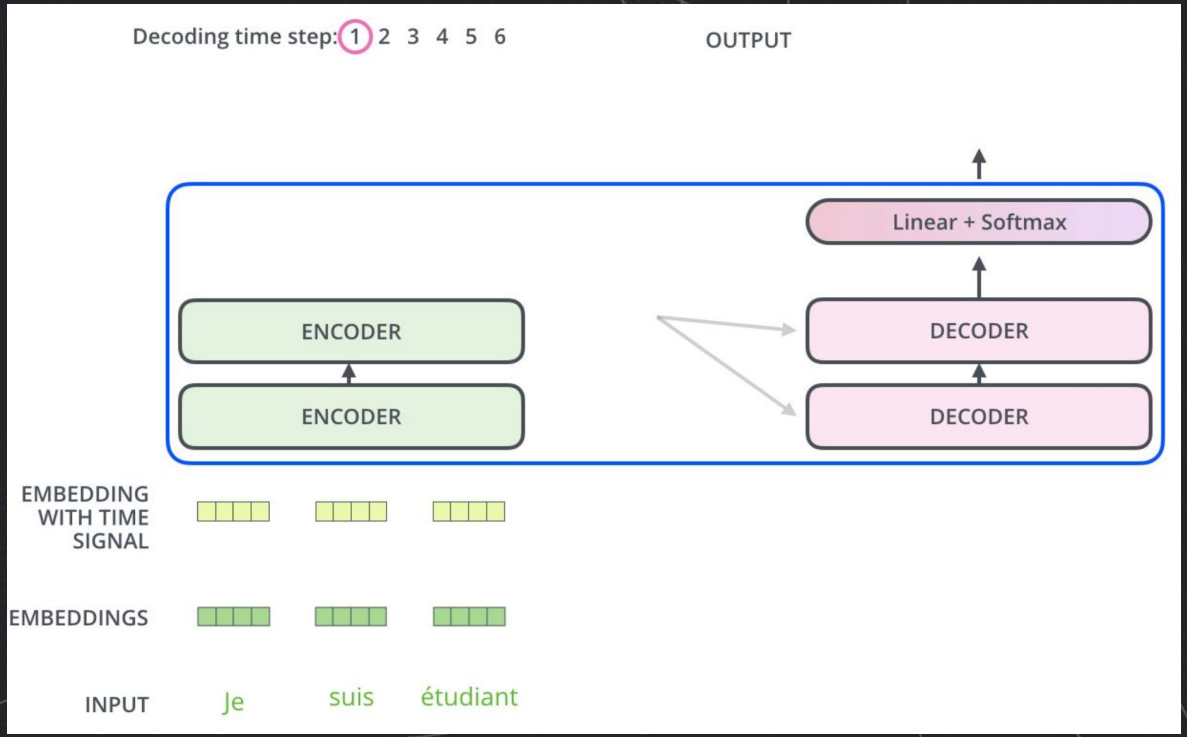


It provides meaningful distances between the embedding vectors once they're projected into Q/K/V vectors and during dot-product attention

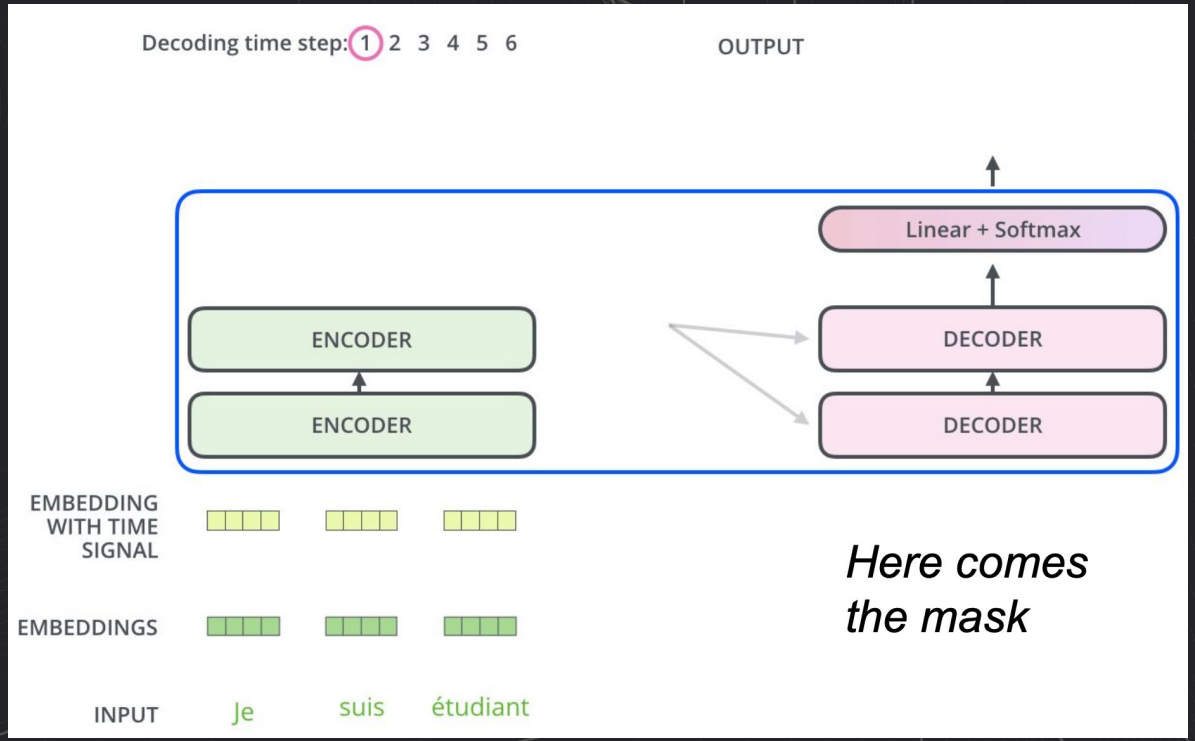
Positional Encoding



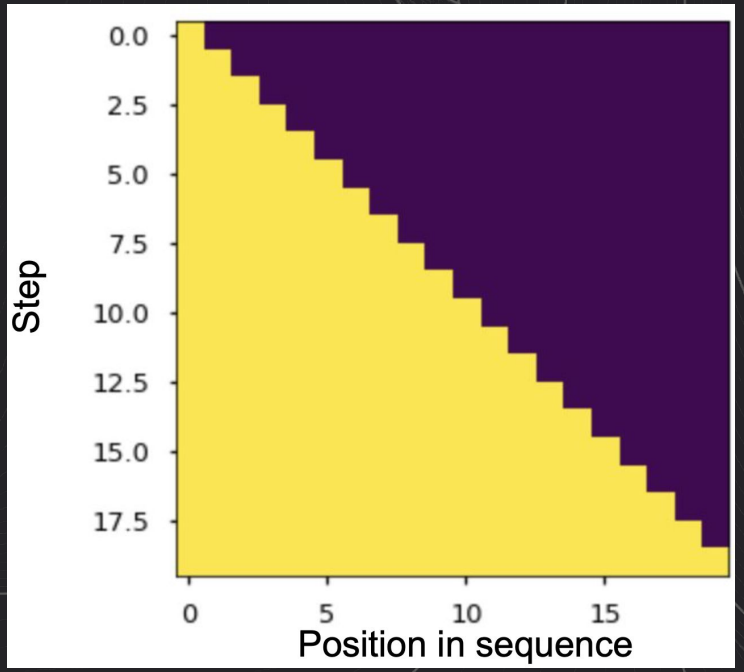
Decoder side



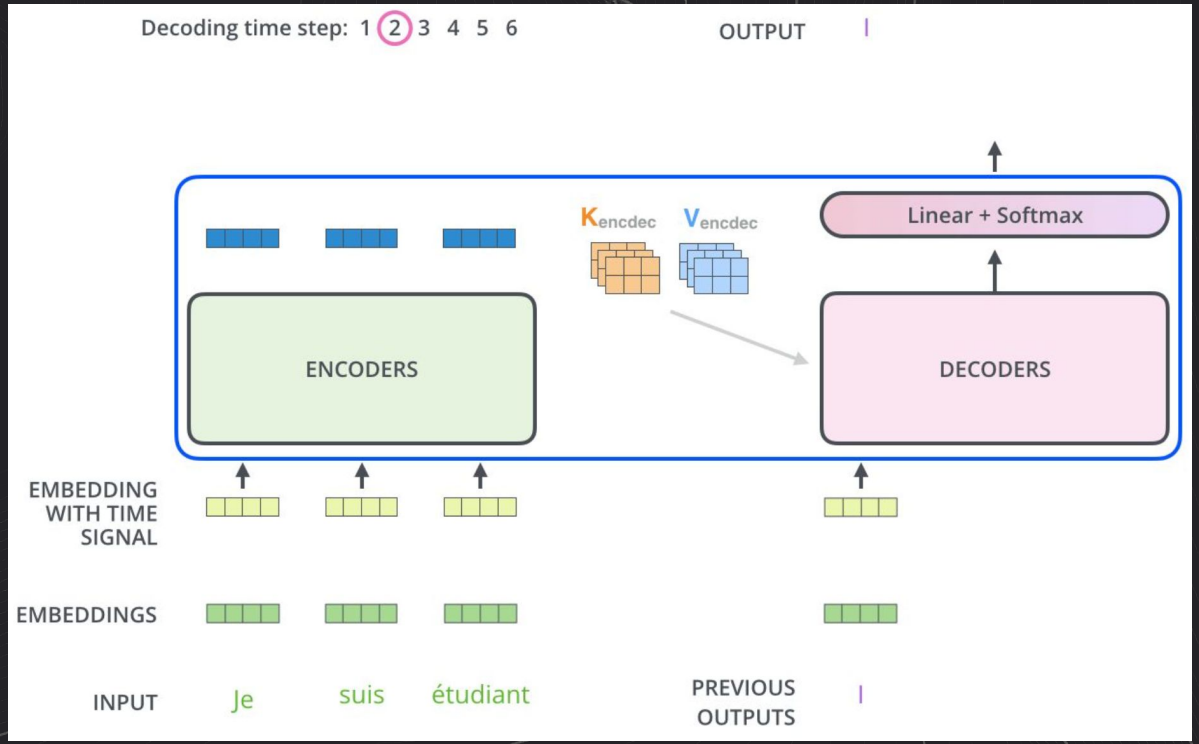
Decoder side



Masked decoder input



Decoder side



Open AI Transformer

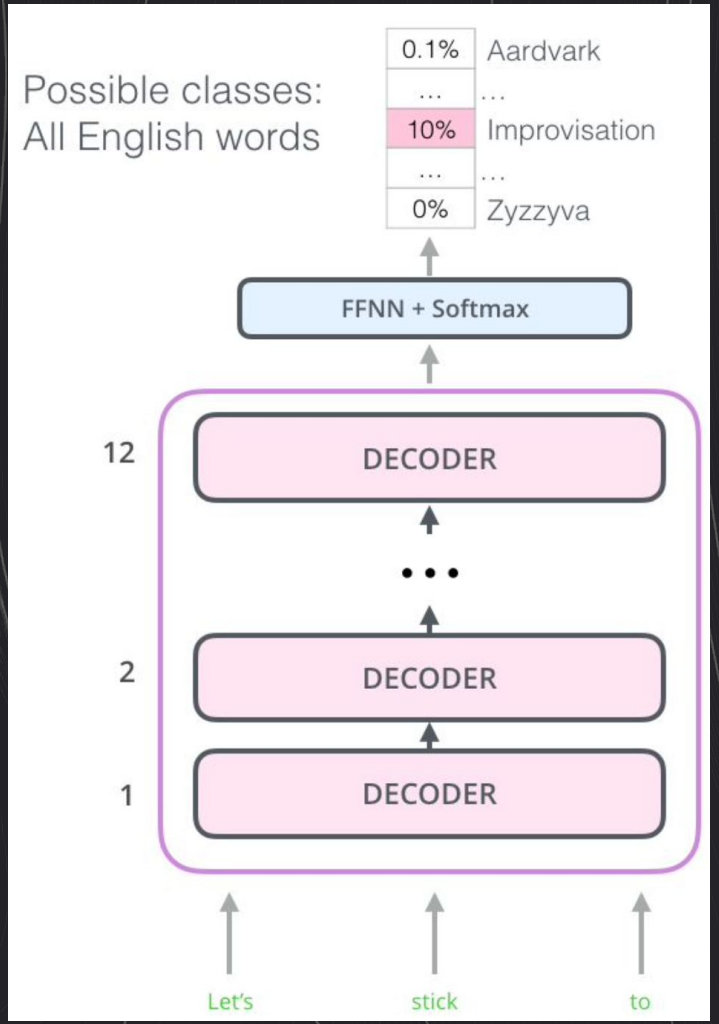
The Encoder-Decoder architecture of the transformer made it perfect for machine translation.

Main goal is to train a language model that can be fine-tuned for other tasks.

Open AI Transformer

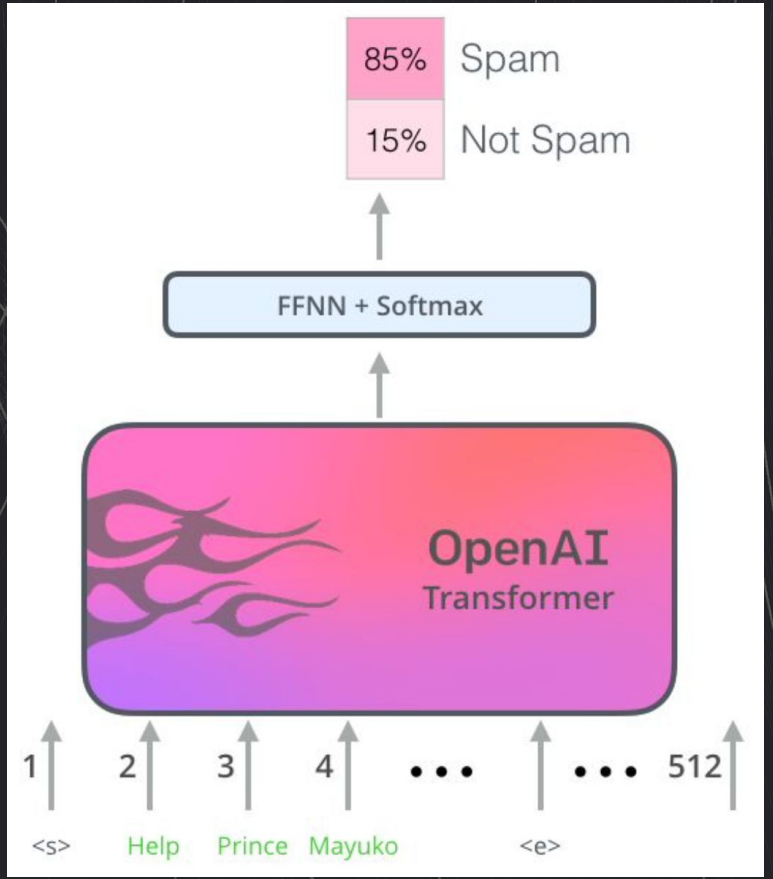
Differences from Vanilla Transformer:

- no encoder;
- decoder layers would not have the encoder-decoder attention sublayer;
- pre-train the model on predicting the next word using massive (unlabele) datasets.



Open AI Transformer

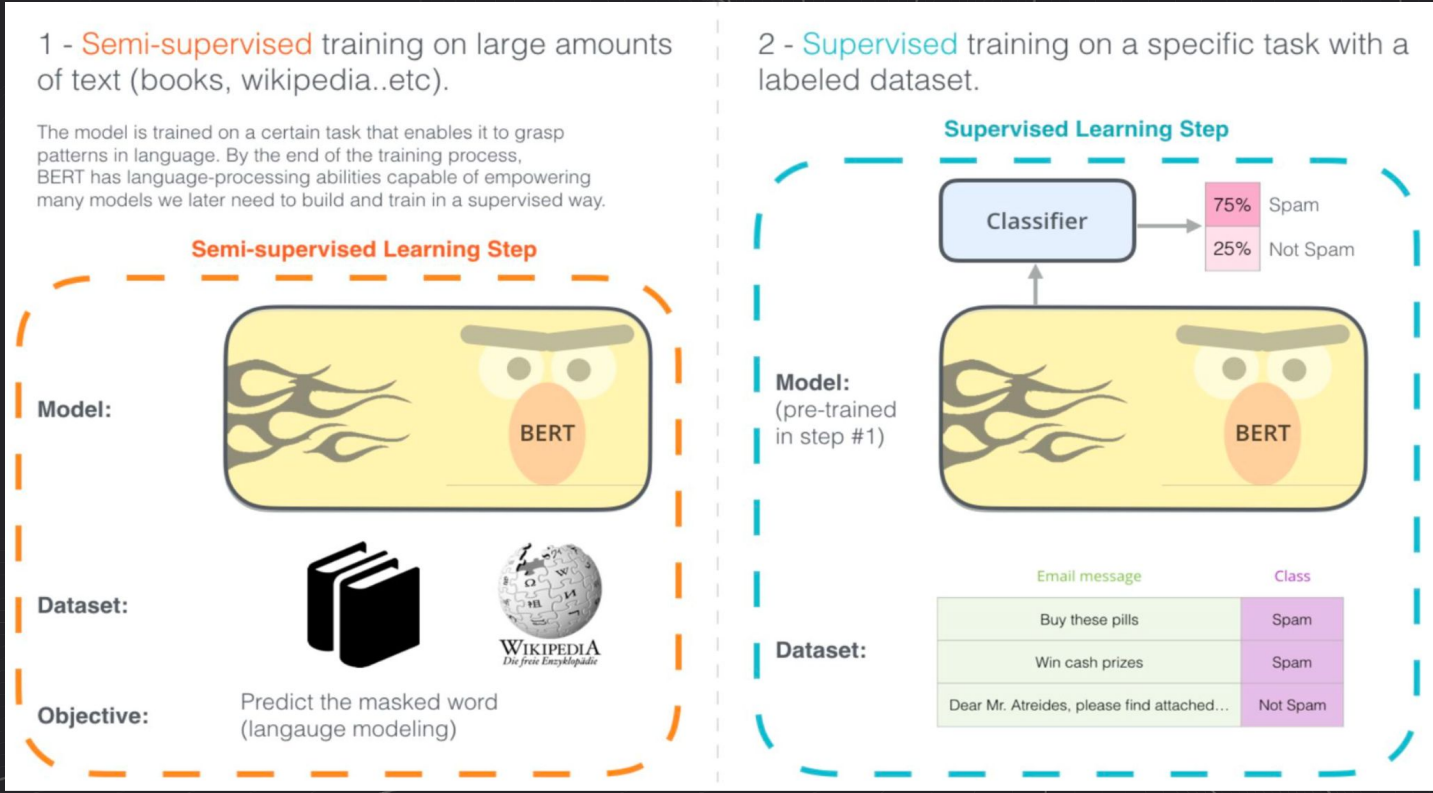
Now we can use this model for downstream tasks, such as sentence classification and so on.



BERT

BERT - Bidirectional Encoder Representation from Transformers.

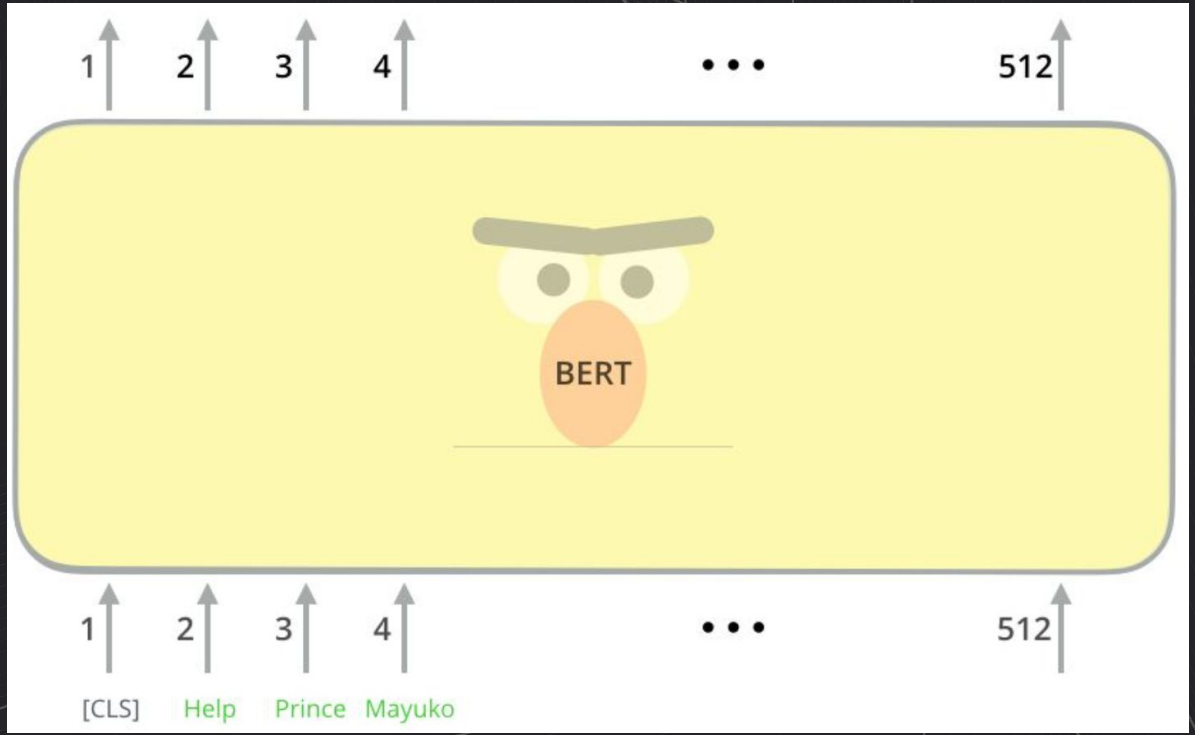
BERT


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.
Semi-supervised Learning Step
Model:
BERT
Dataset:
WIKIPEDIA
Die freie Enzyklopädie
Objective:
Predict the masked word (langauge modeling)
2 - Supervised training on a specific task with a labeled dataset.
Supervised Learning Step
Classifier
75% Spam
25% Not Spam
Model: (pre-trained in step #1)
BERT
Dataset:
Email message
Class
Buy these pills
Spam
Win cash prizes
Spam
Dear Mr. Atreides, please find attached…
Not Spam

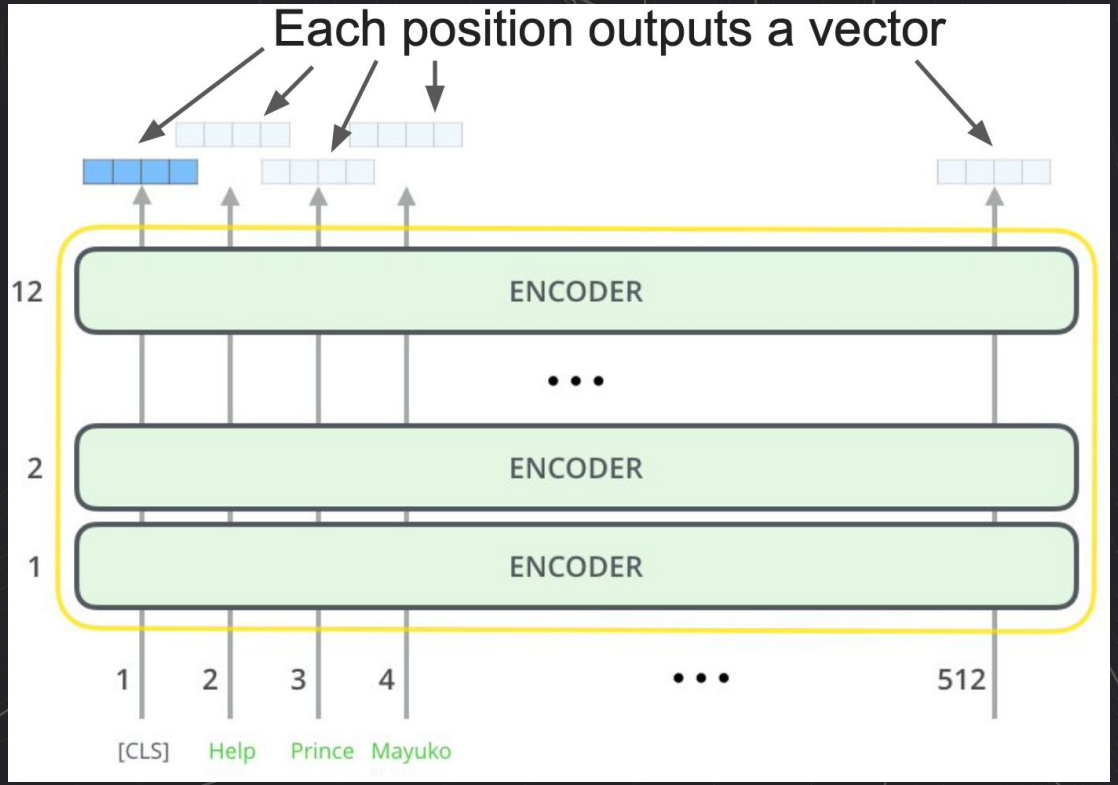
BERT vs Transformer

			
		Base BERT	**Large BERT**
Encoders	6	12	24
Units in FFN	512	768	1024
Attention Heads	8	12	16

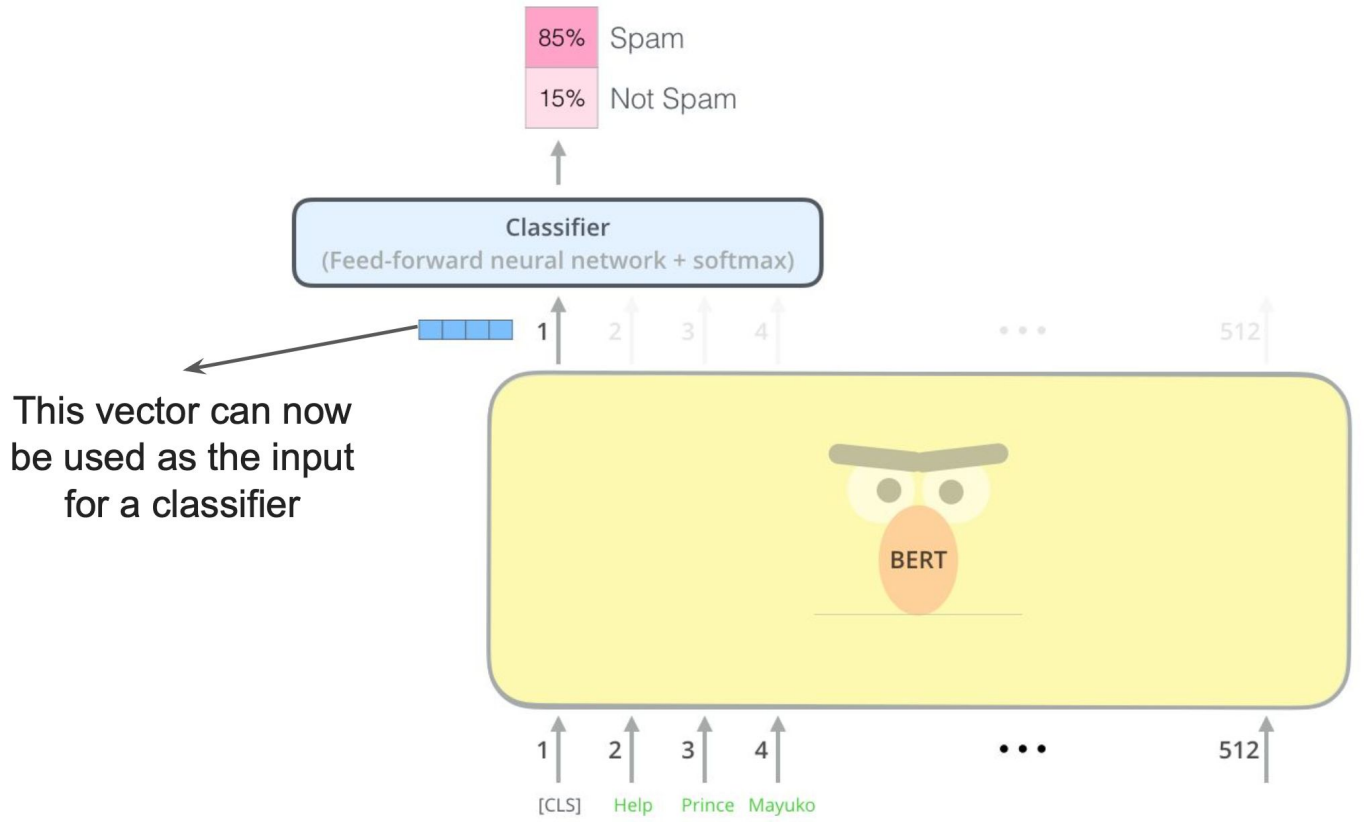
BERT



BERT


Each position outputs a vector
12
ENCODER
2
ENCODER
1
ENCODER
1
2
3
4
512
[CLS]
Help
Prince
Mayuko

BERT


85% Spam
15% Not Spam
Classifier
(Feed-forward neural network + softmax)
1
2
3
4
512
This vector can now be used as the input for a classifier
BERT
1
2
3
4
512
[CLS]
Help
Prince
Mayuko

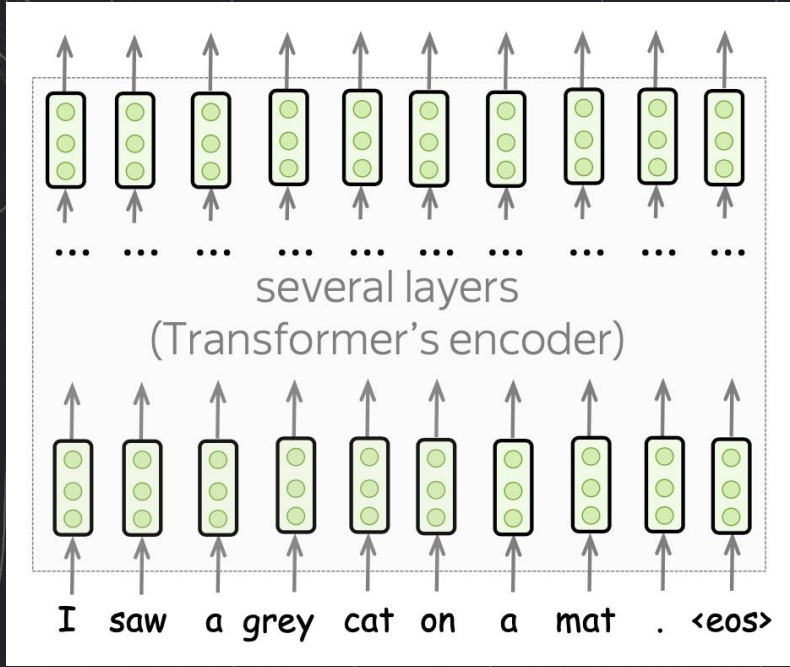
BERT

Model architecture:

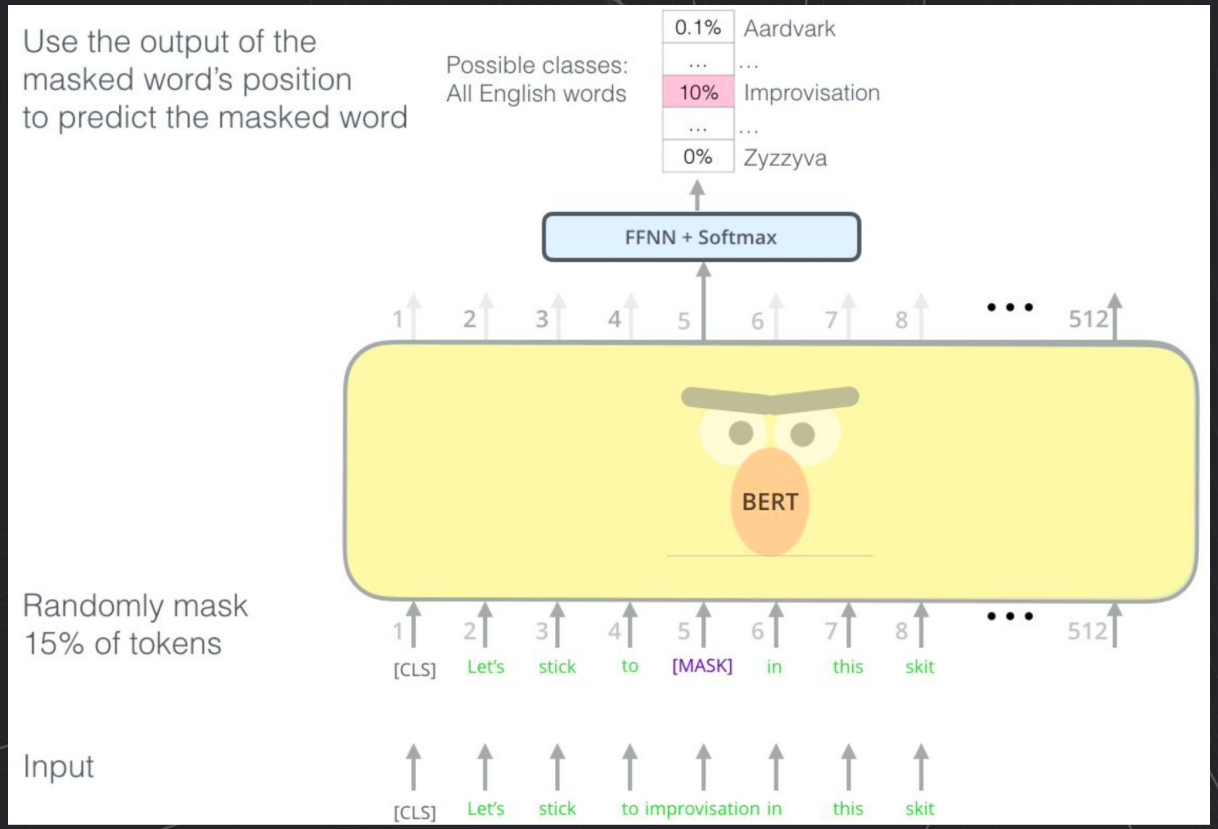
- Transformer Encoder

What is special about it:

- Training objectives:
  - MLM: Masked Language Modeling
  - NSP: Next Sentence Prediction
- Lots of data



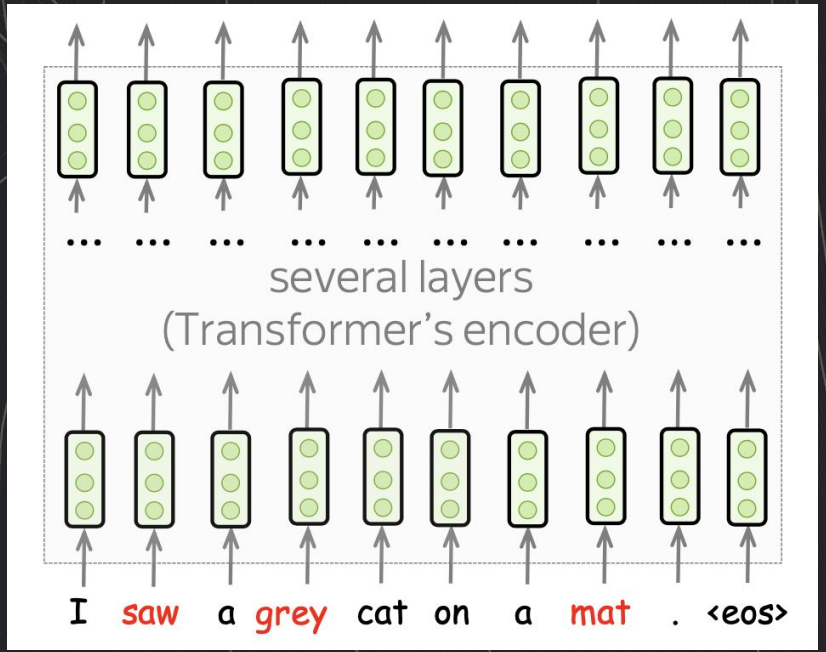
BERT training (MLM)



BERT training (MLM)

At each training step:

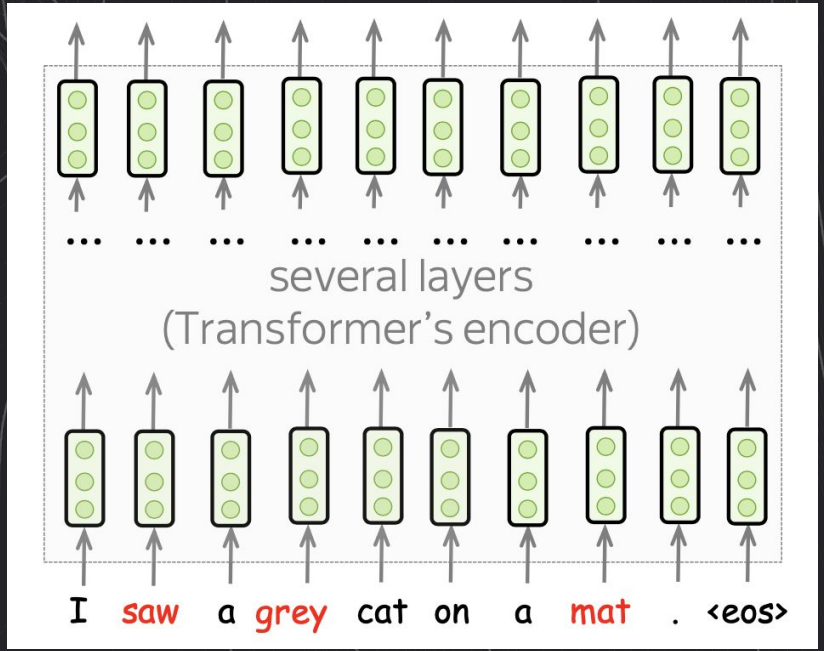
- pick randomly 15% of tokens



BERT training (MLM)

At each training step:

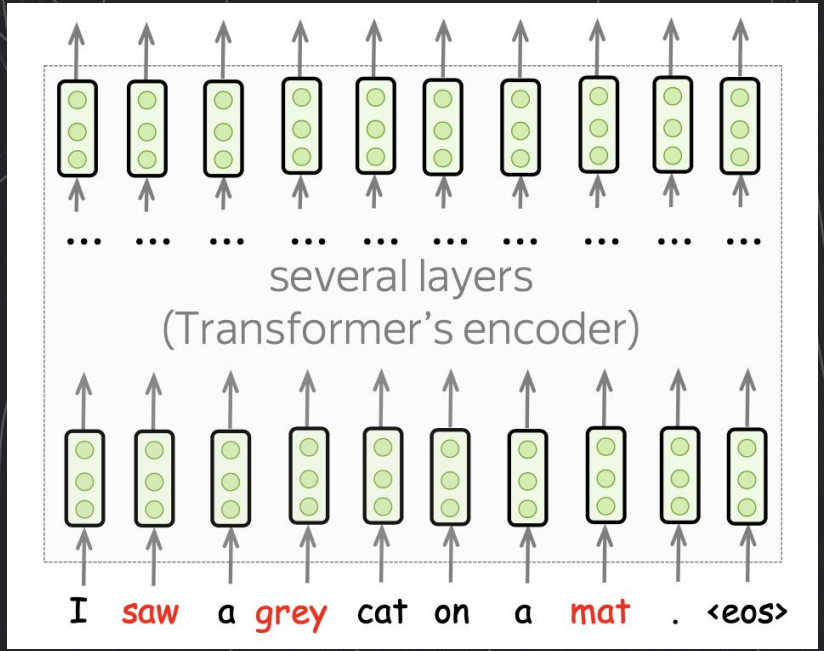
- pick randomly 15% of tokens
- replace each of chose tokens with:
  - [MASK] with 80% prob



BERT training (MLM)

At each training step:

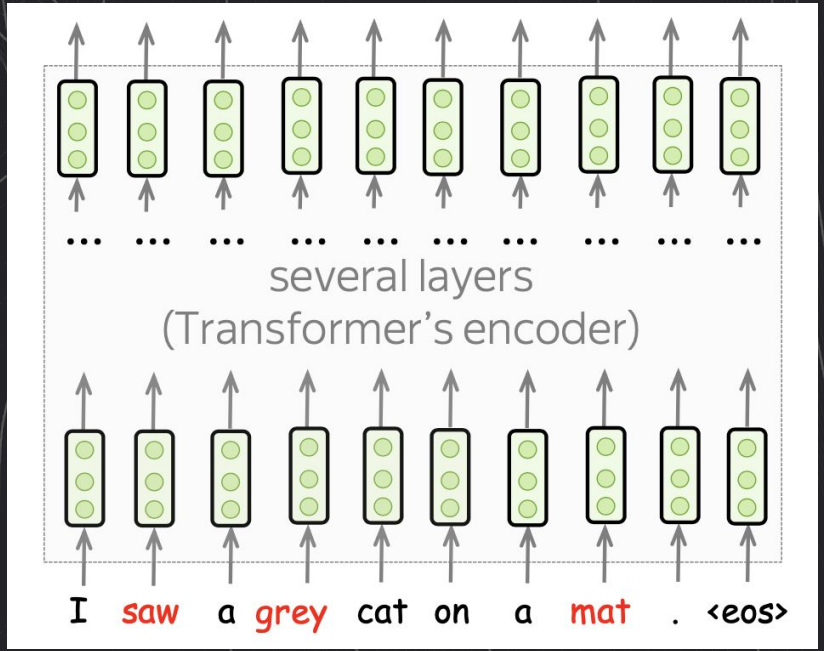
- pick randomly 15% of tokens
- replace each of chose tokens with:
  - [MASK] with 80% prob
  - random token with 10% prob



BERT training (MLM)

At each training step:

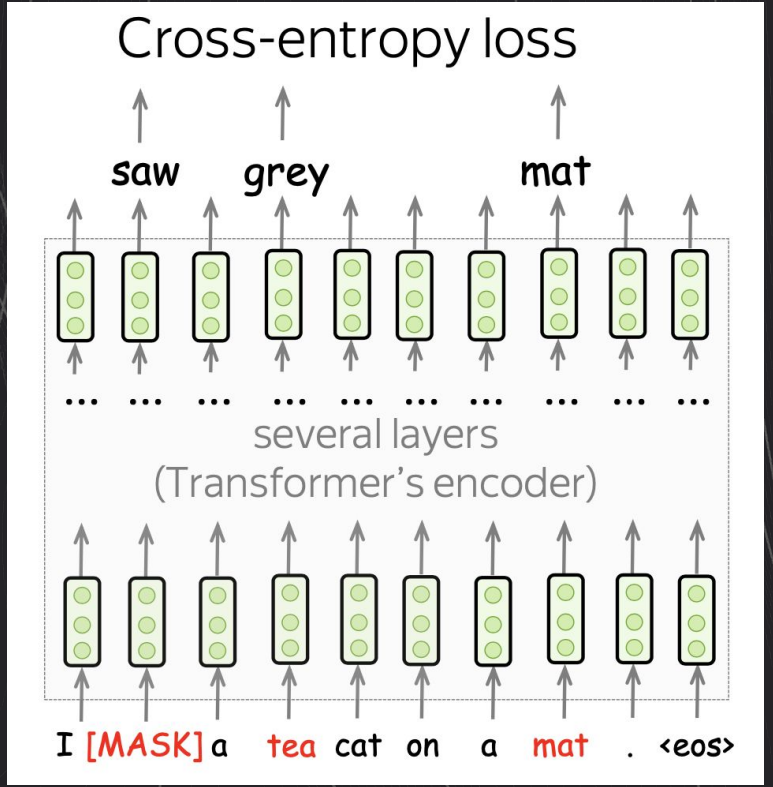
- pick randomly 15% of tokens
- replace each of chose tokens with:
  - [MASK] with 80% prob
  - random token with 10% prob
  - self with 10% prob



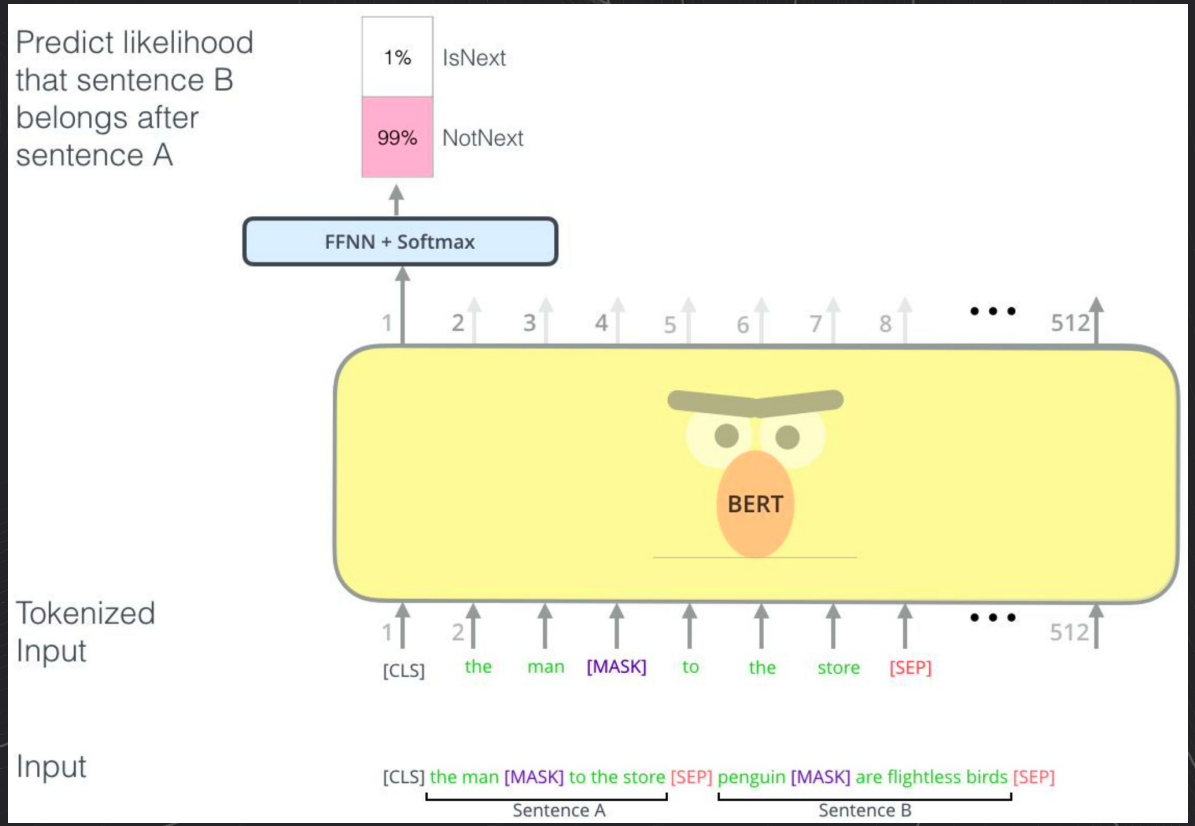
BERT training (MLM)

At each training step:

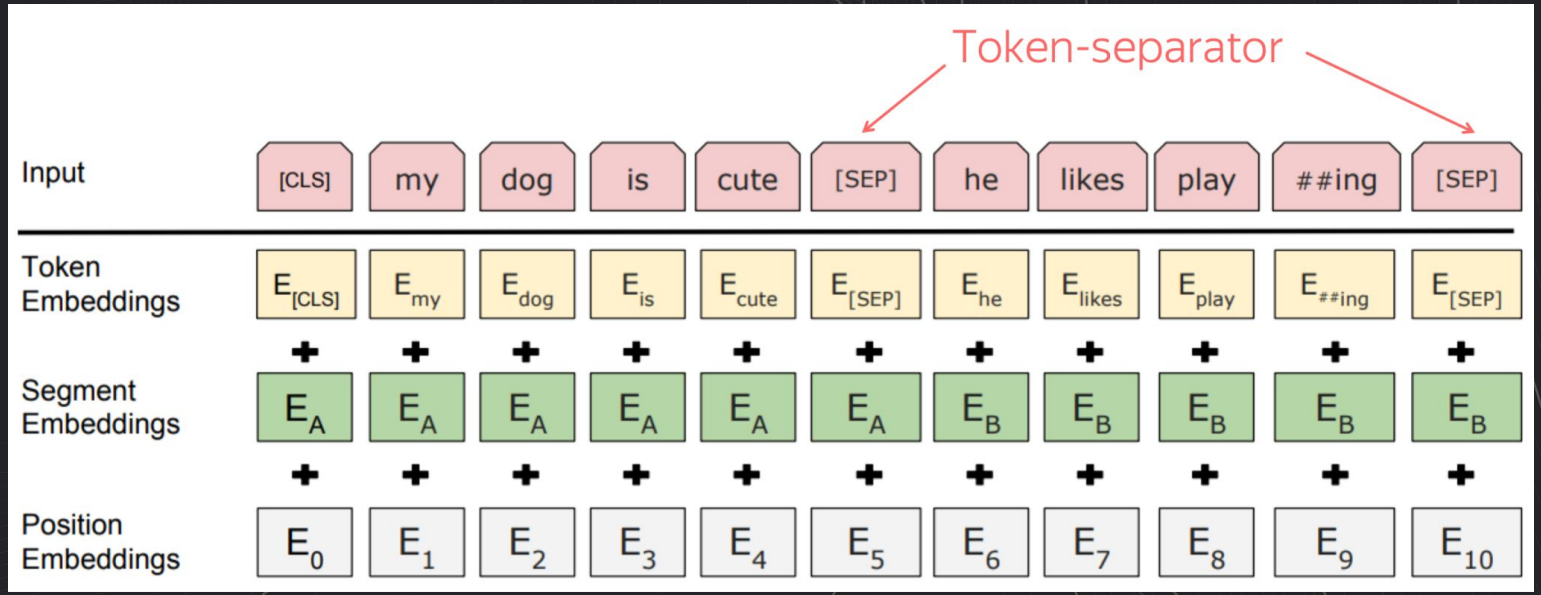
- pick randomly 15% of tokens
- replace each of chose tokens with:
  - [MASK] with 80% prob
  - random token with 10% prob
  - self with 10% prob
- predict original tokens (only chosen ones!)



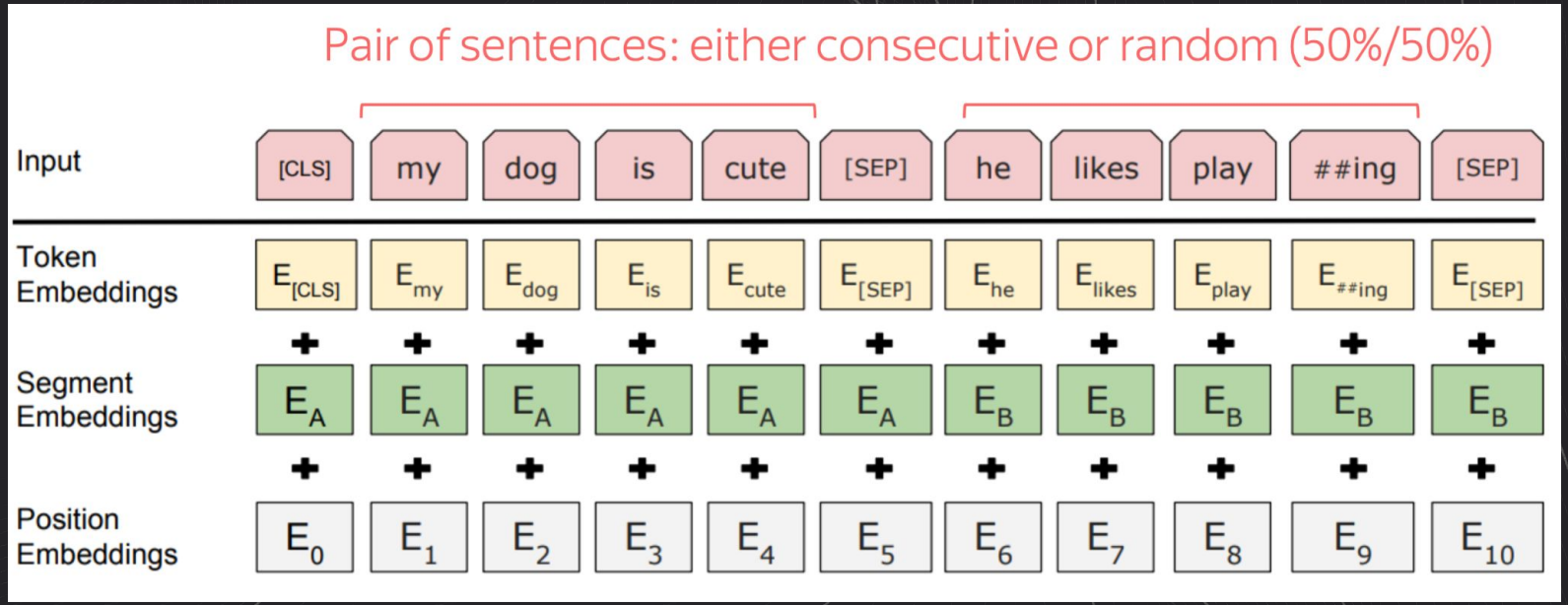
BERT training (NSP)



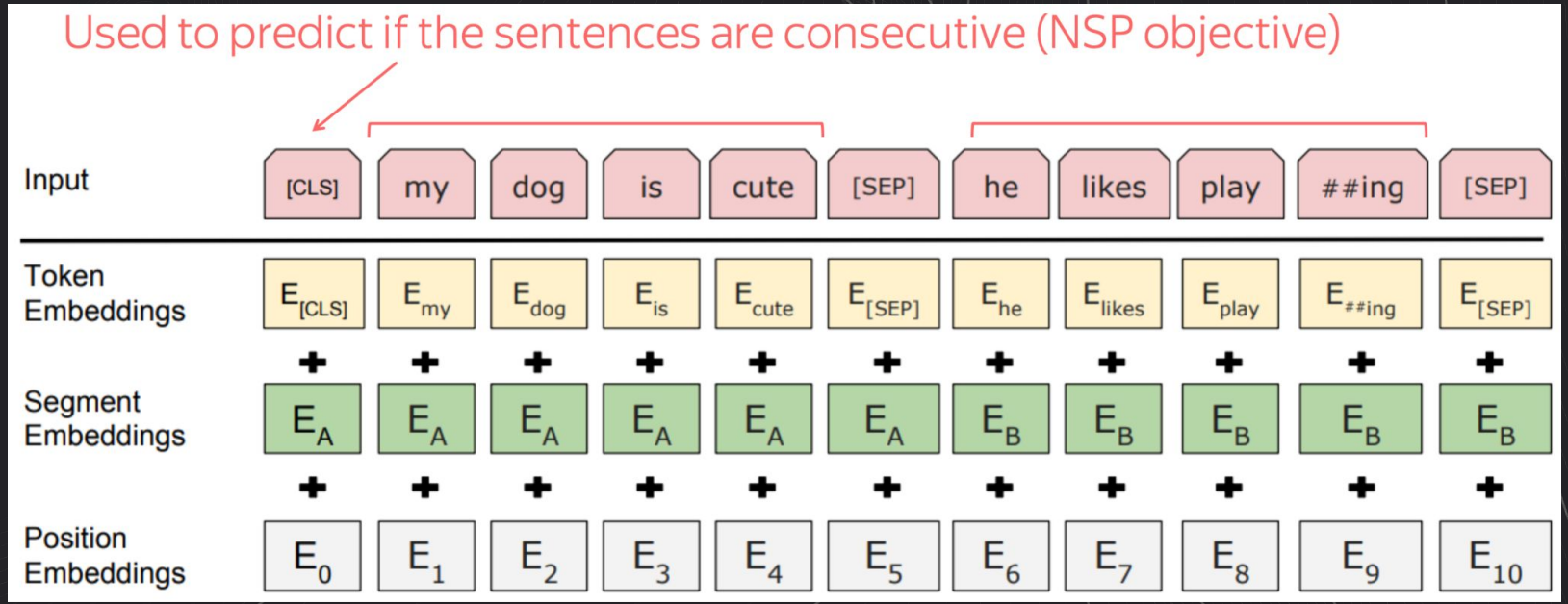
BERT training (NSP)



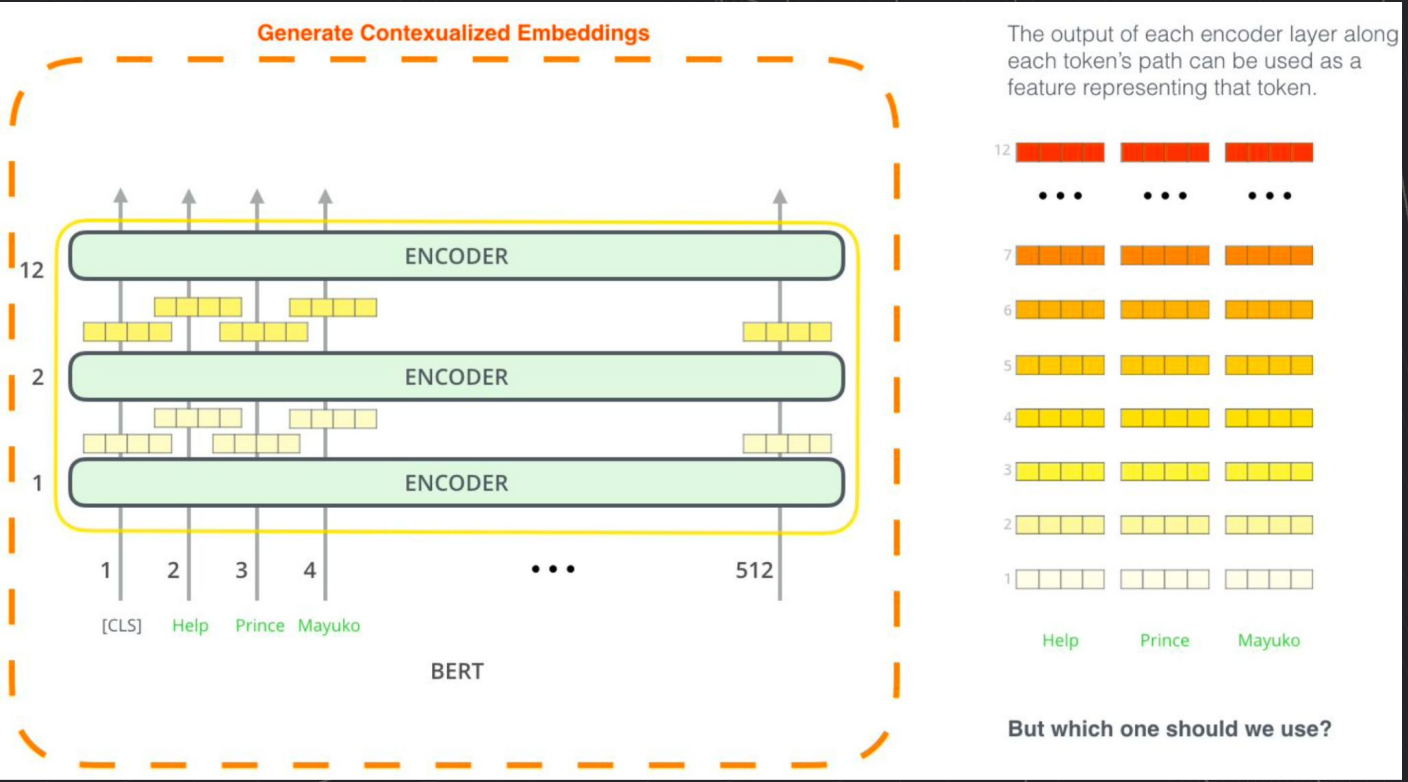
BERT training (NSP)



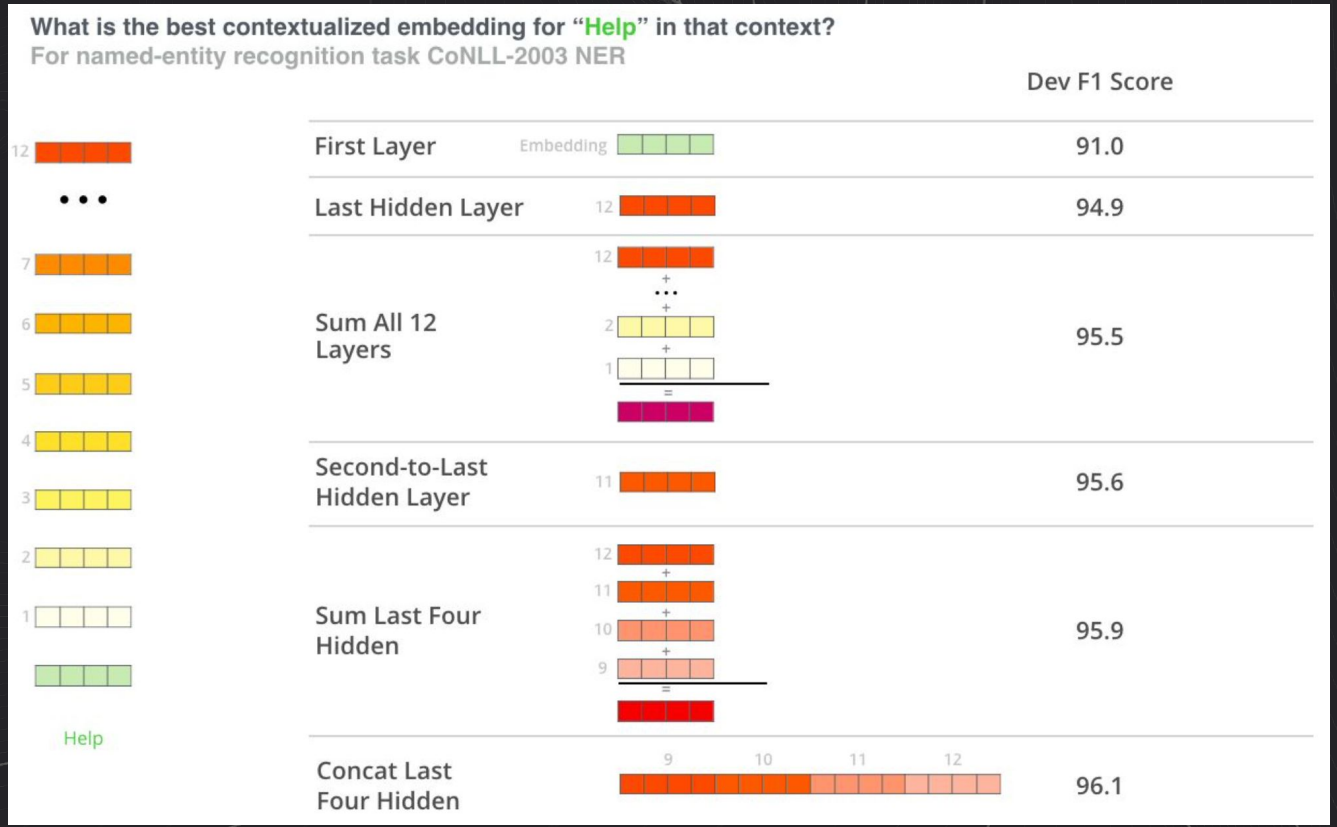
BERT training (NSP)



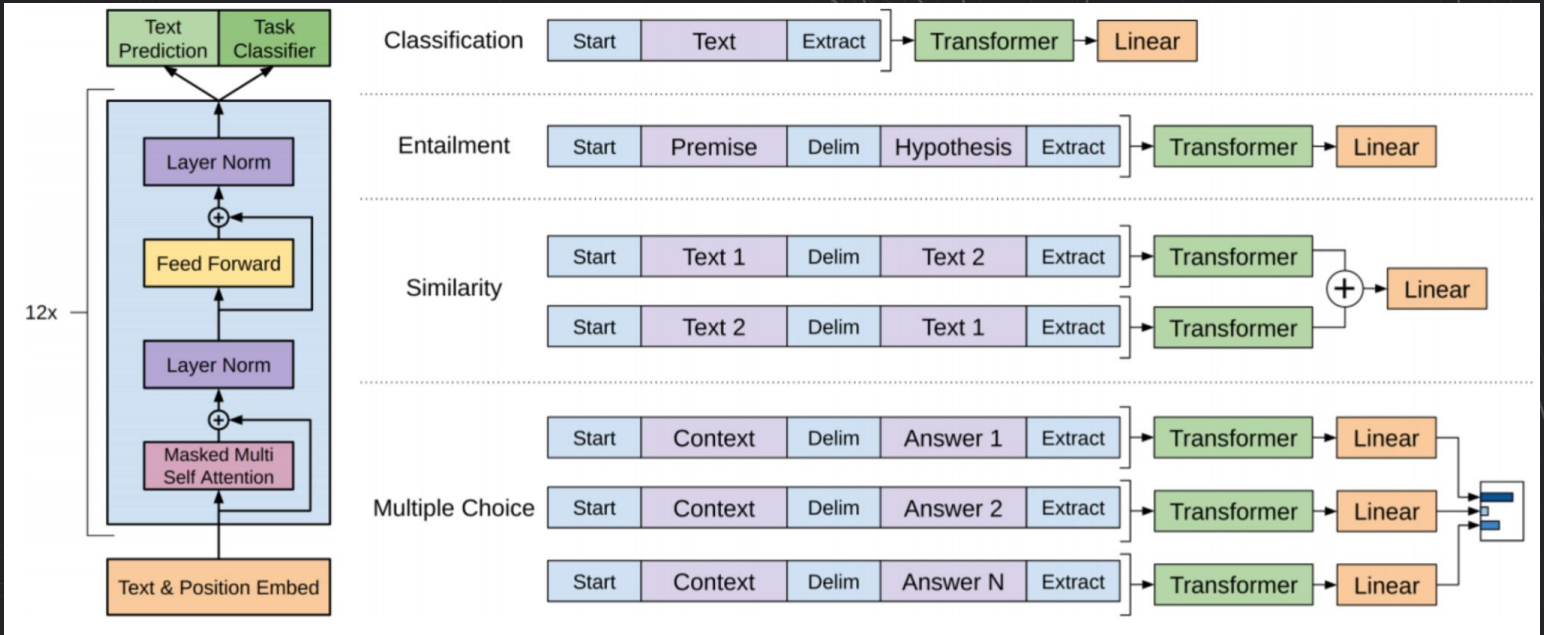
BERT for feature extraction



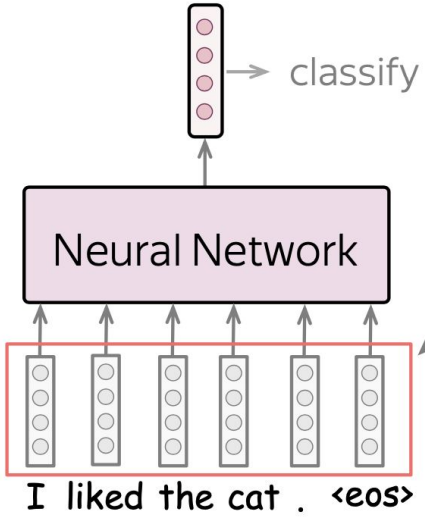
BERT for feature extraction



GPT (1-2-3): Transformer Decoder



Text classification approaches



Input word embeddings:

- Train from scratch
- Take pretrained (Word2Vec, GloVe)
- Initialize with pretrained, then fine-tune

Text classification approaches

Train from scratch

What they will know:

May be not enough to learn relationships between words

Take pretrained (Word2vec, Glove)

Initialize with pretrain, then fine-tune

Text classification approaches

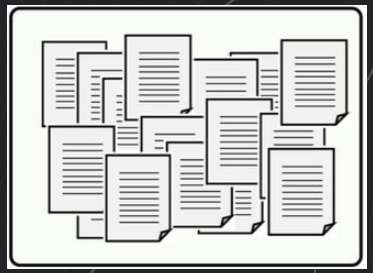
Train from scratch

What they will know:

May be not enough to learn relationships between words

Take pretrained (Word2vec, Glove)

What they will know:



Know relationships between words,

but are not specific to the task

Initialize with pretrain, then fine-tune

Text classification approaches

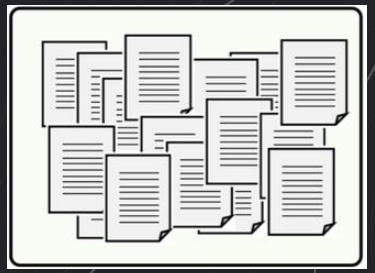
Train from scratch

What they will know:

May be not enough to learn relationships between words

Take pretrained (Word2vec, Glove)

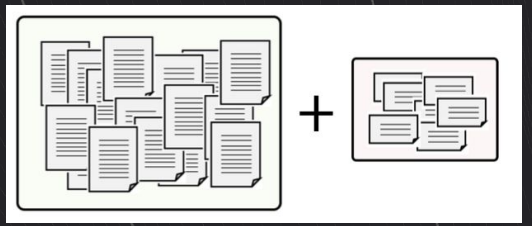
What they will know:



Know relationships between words, but are not specific to the task

Initialize with pretrain, then fine-tune

What they will know:



Know relationships between words and adapted for the task

Text classification approaches

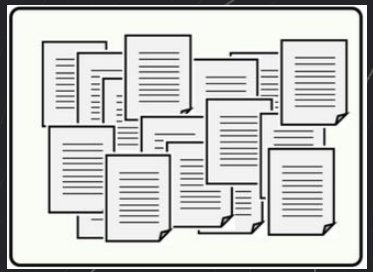
Train from scratch

What they will know:

May be not enough to learn relationships between words

Take pretrained (Word2vec, Glove)

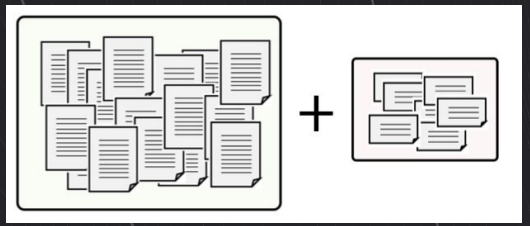
What they will know:



Know relationships between words, but are not specific to the task

Initialize with pretrain, then fine-tune

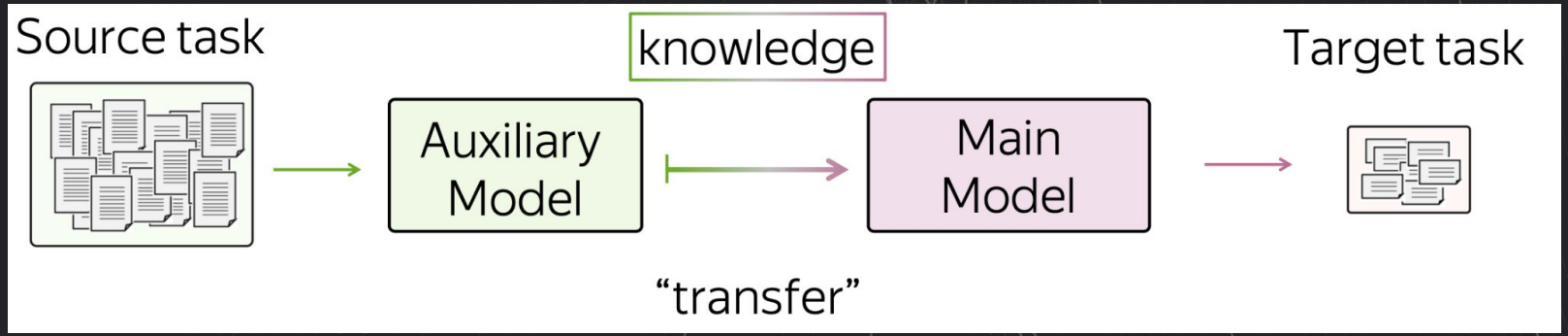
What they will know:



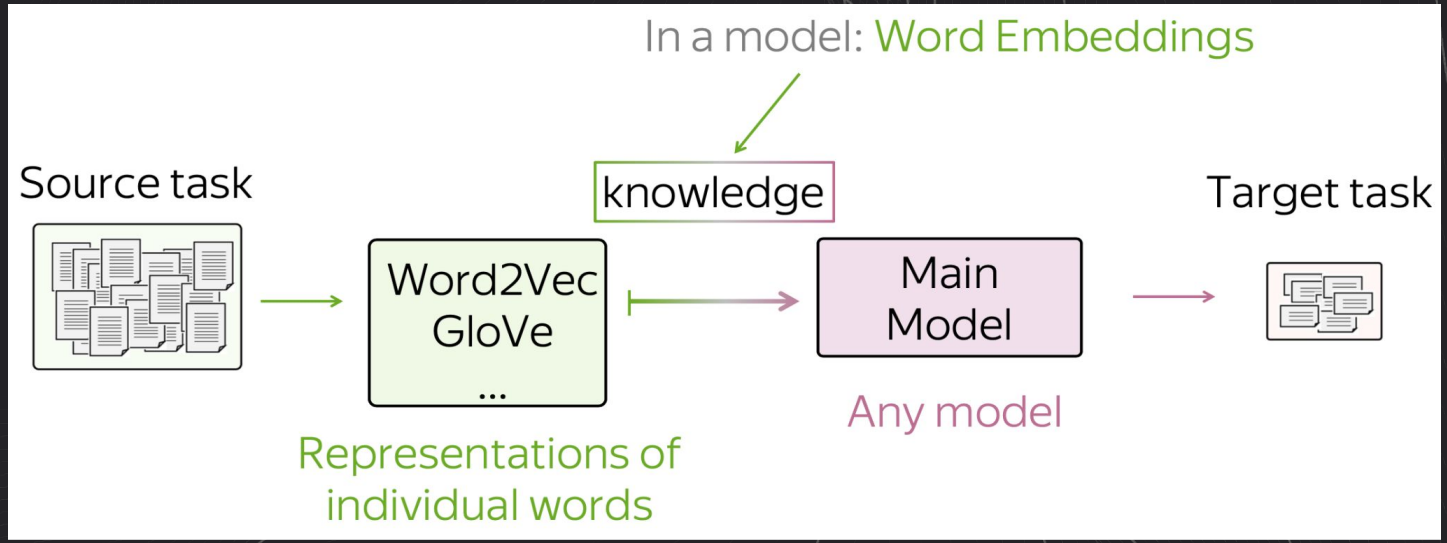
Know relationships between words and adapted for the task

“Transfer” knowledge

Transfer learning idea

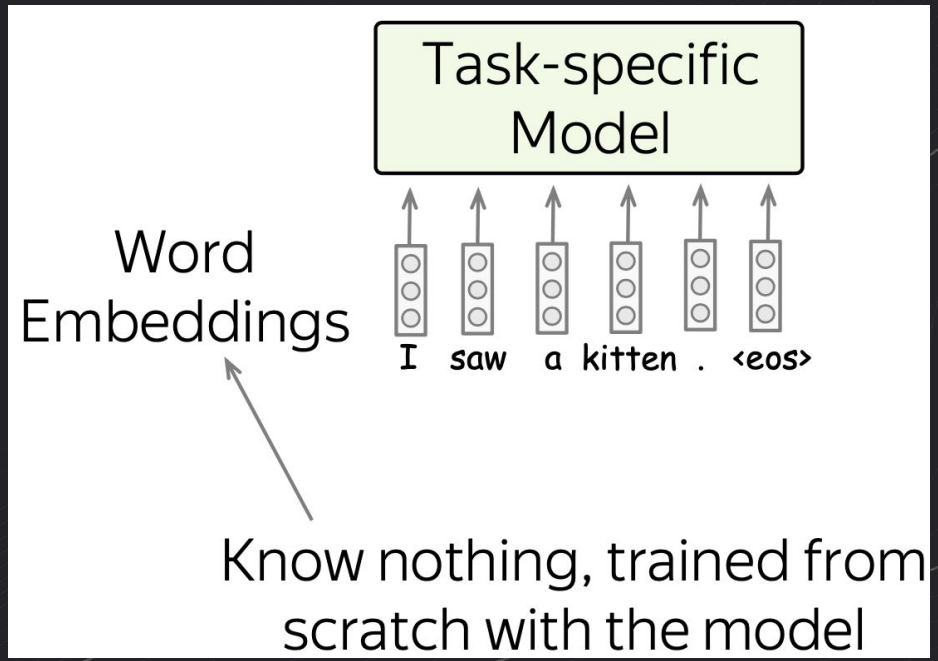


Transfer learning approach



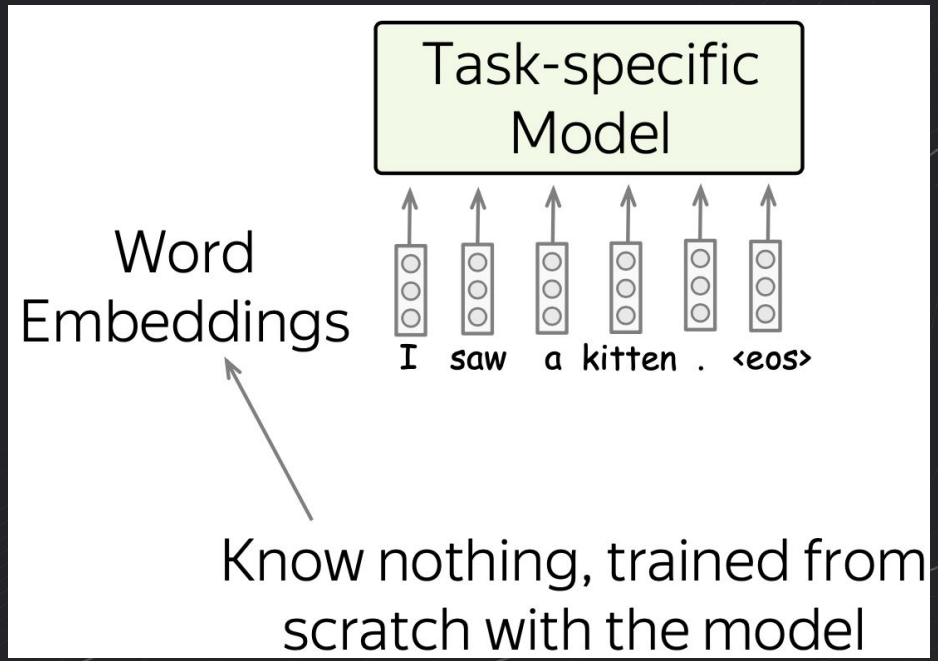
Transfer learning through Word Embedding

BEFORE:

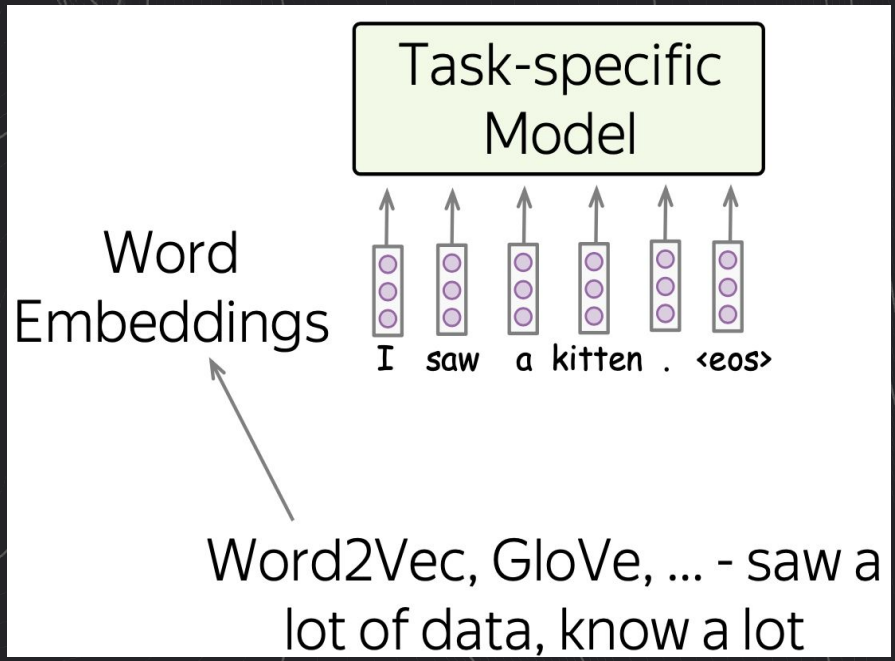


Transfer learning through Word Embedding

BEFORE:

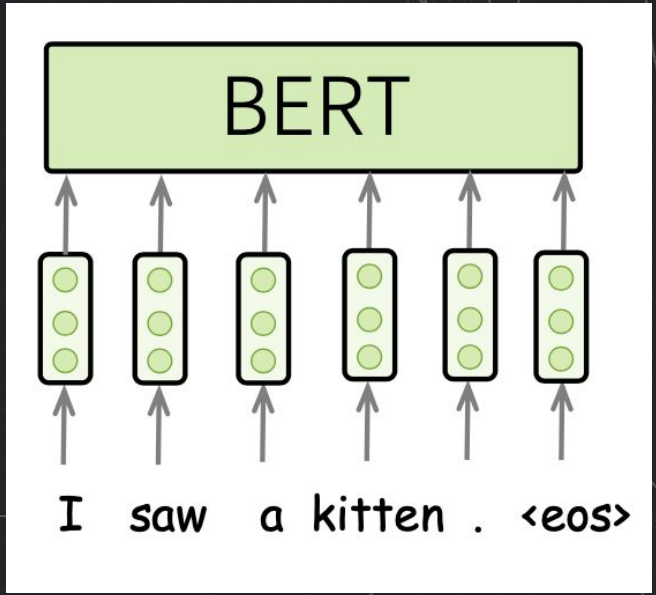


AFTER:



Transfer learning after BERT

No task specific model at at all!



Terminology

- word embeddings
- Encoder
- Decoder
- self-attention
- attention
- multi-head attention
- masked multi-head attention
- positional encoding
- residual connections
- layer normalization
- Transformer
- BERT
- GPT
- transfer learning