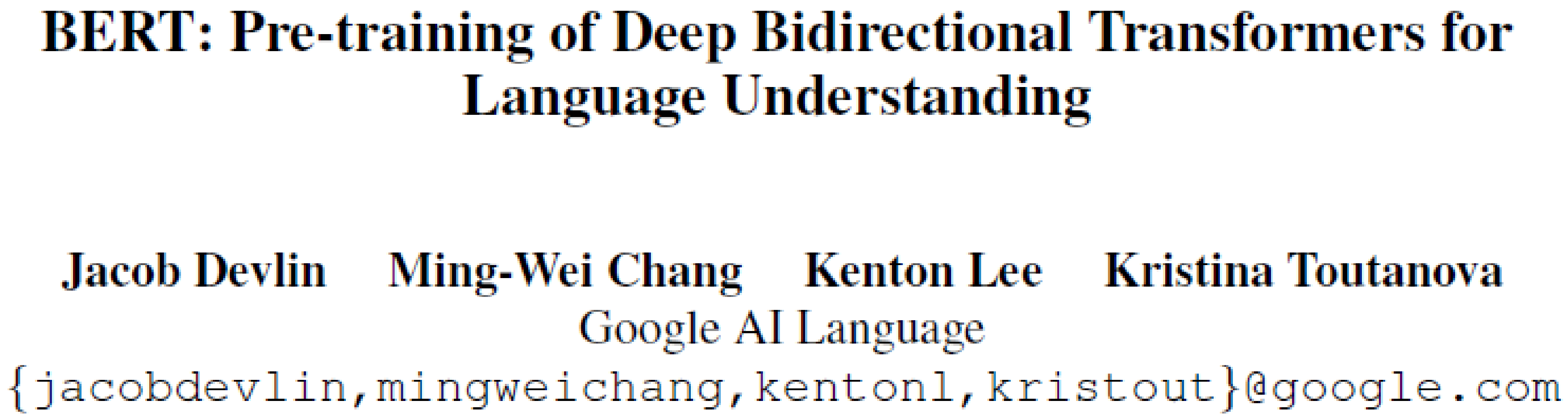


BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

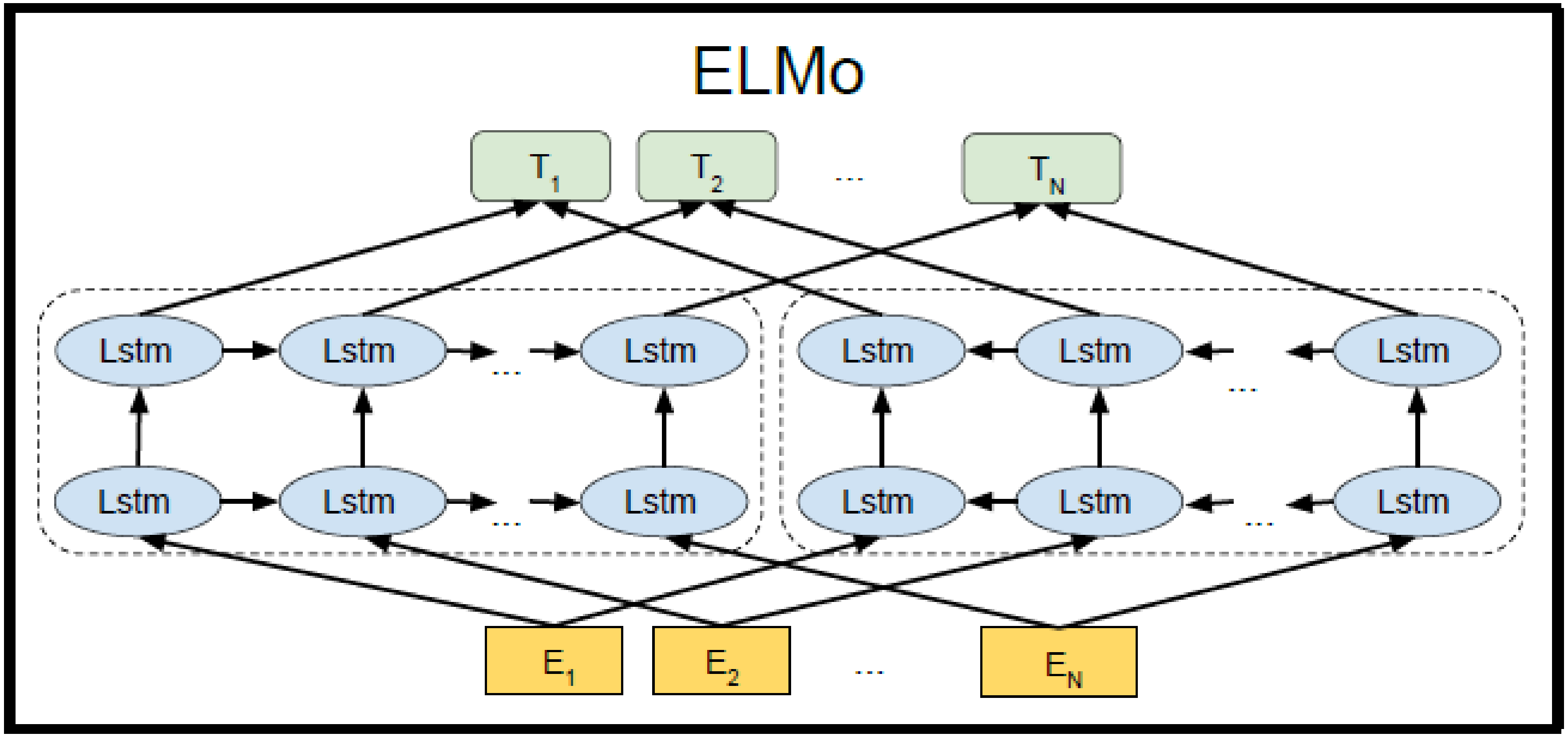
Лиза Корнакова, 24.12.2019

BERT: основная идея

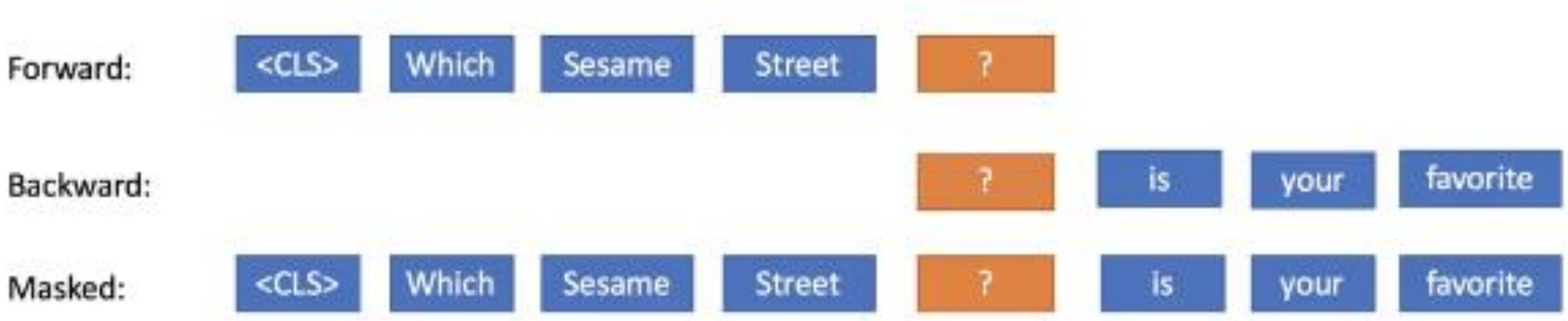
BERT: основная идея

- **B**idirectional **E**ncoder **R**epresentations from **T**ransformers
- Хотим архитектуру, чтобы был общий pre-training
- А затем простой fine-tuning
- Хотим учитывать контекст справа и слева

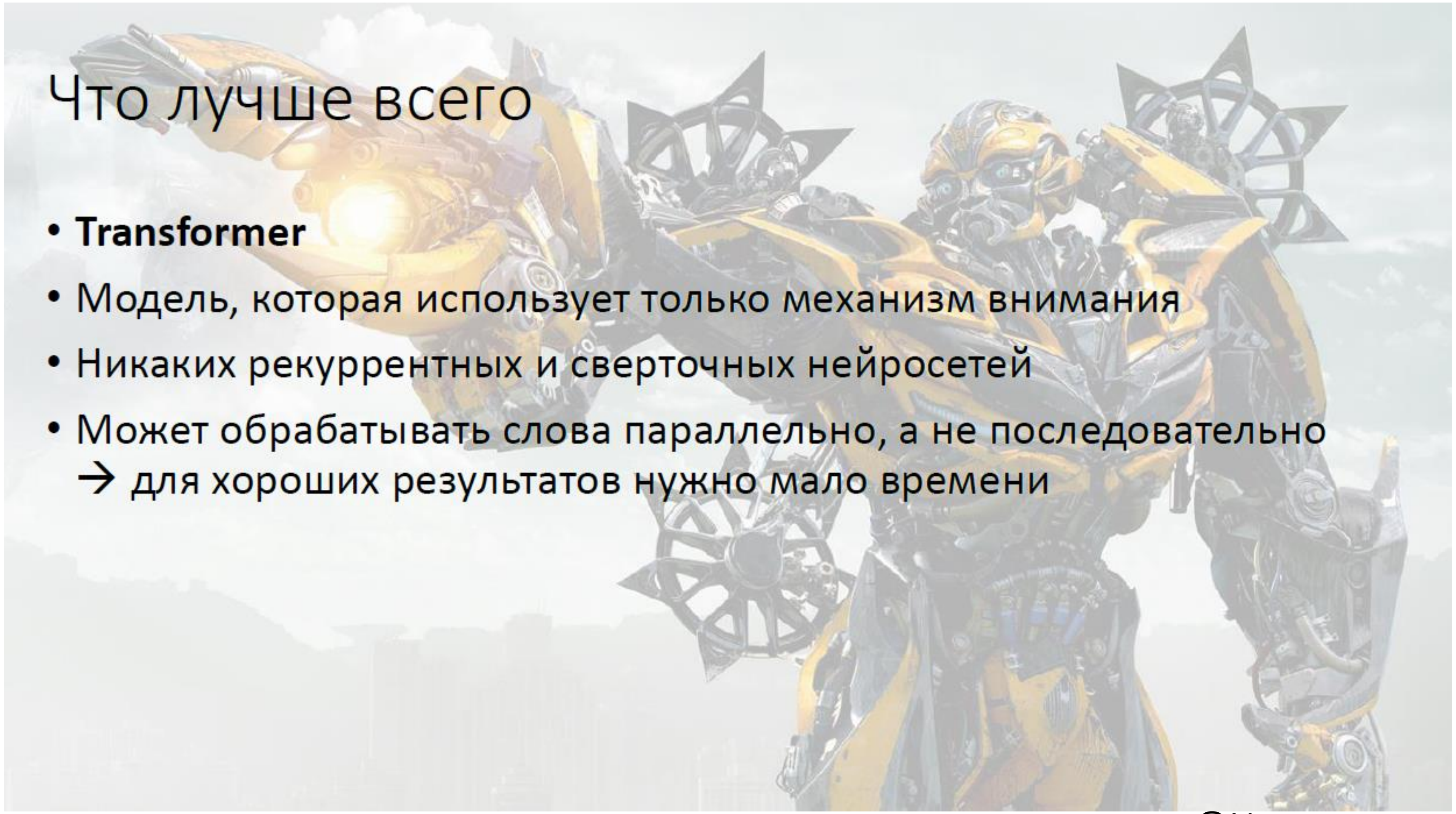
Что умели до BERTa?



Что умели до BERTa?



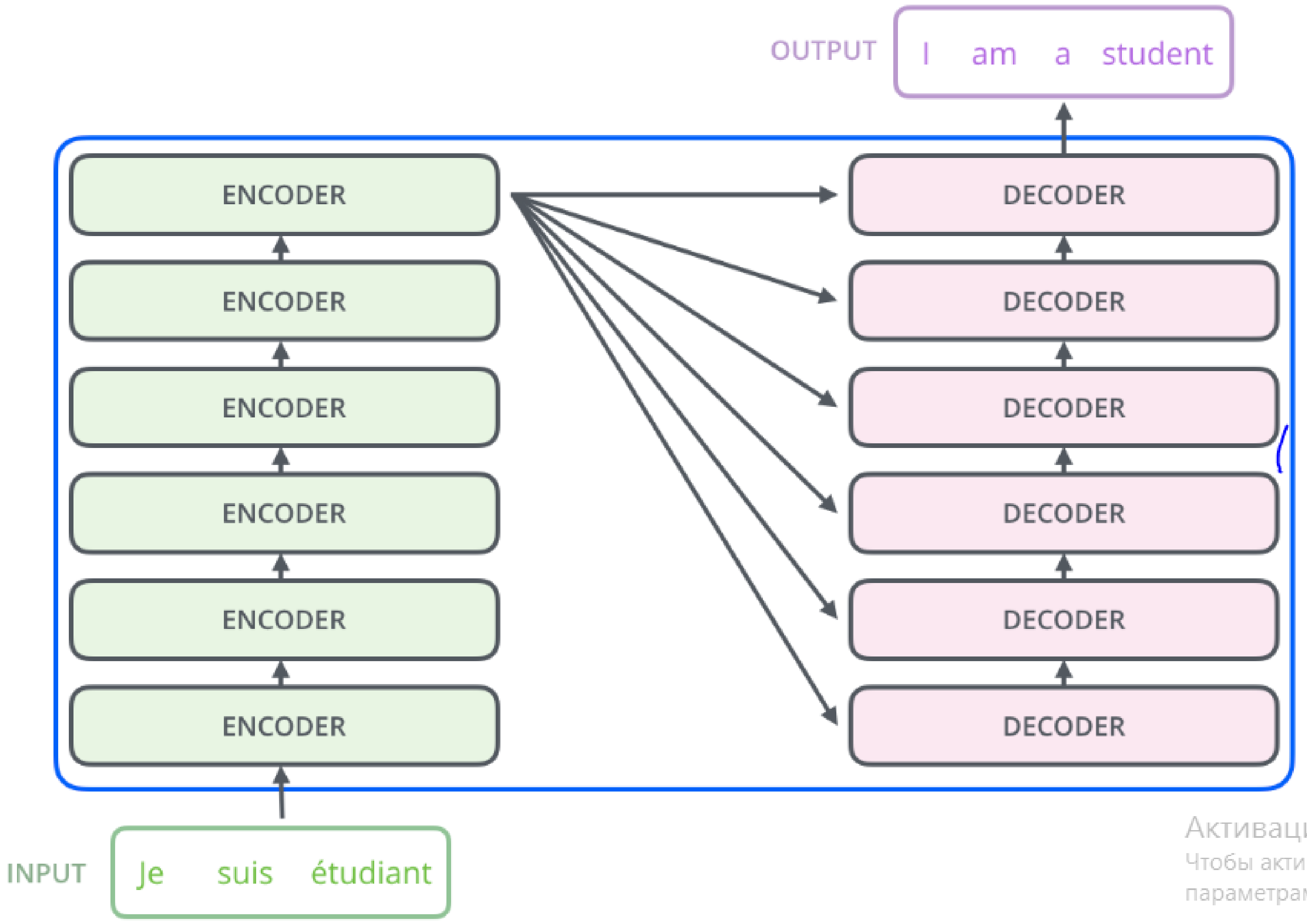
FLASHBACK



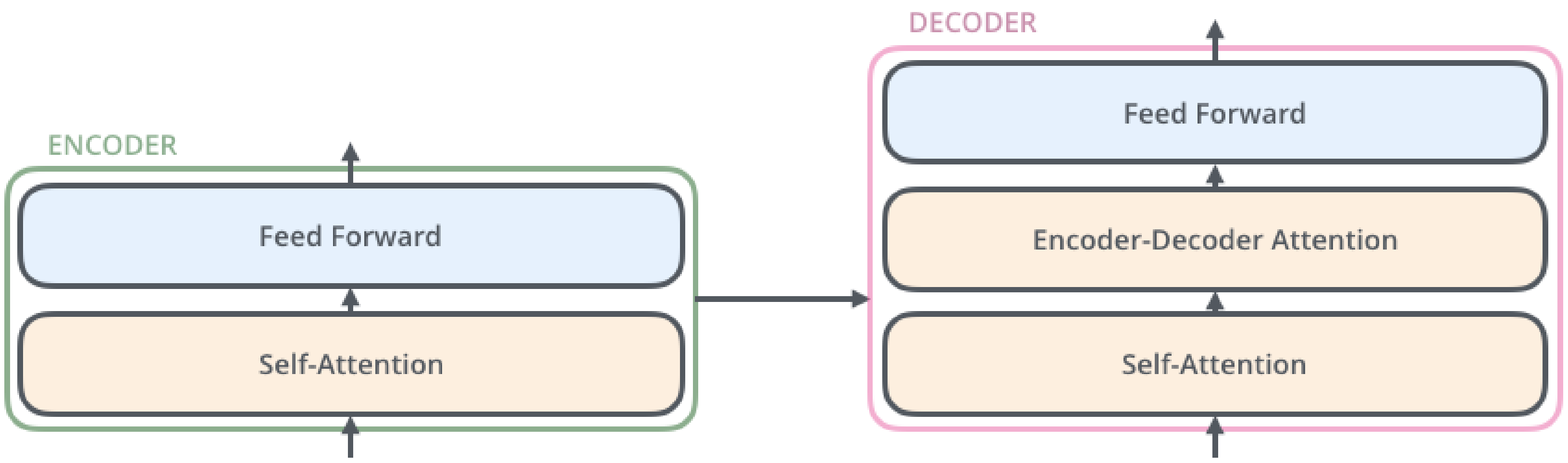
Что лучше всего

- **Transformer**
- Модель, которая использует только механизм внимания
- Никаких рекуррентных и сверточных нейросетей
- Может обрабатывать слова параллельно, а не последовательно
  → для хороших результатов нужно мало времени

Transformer



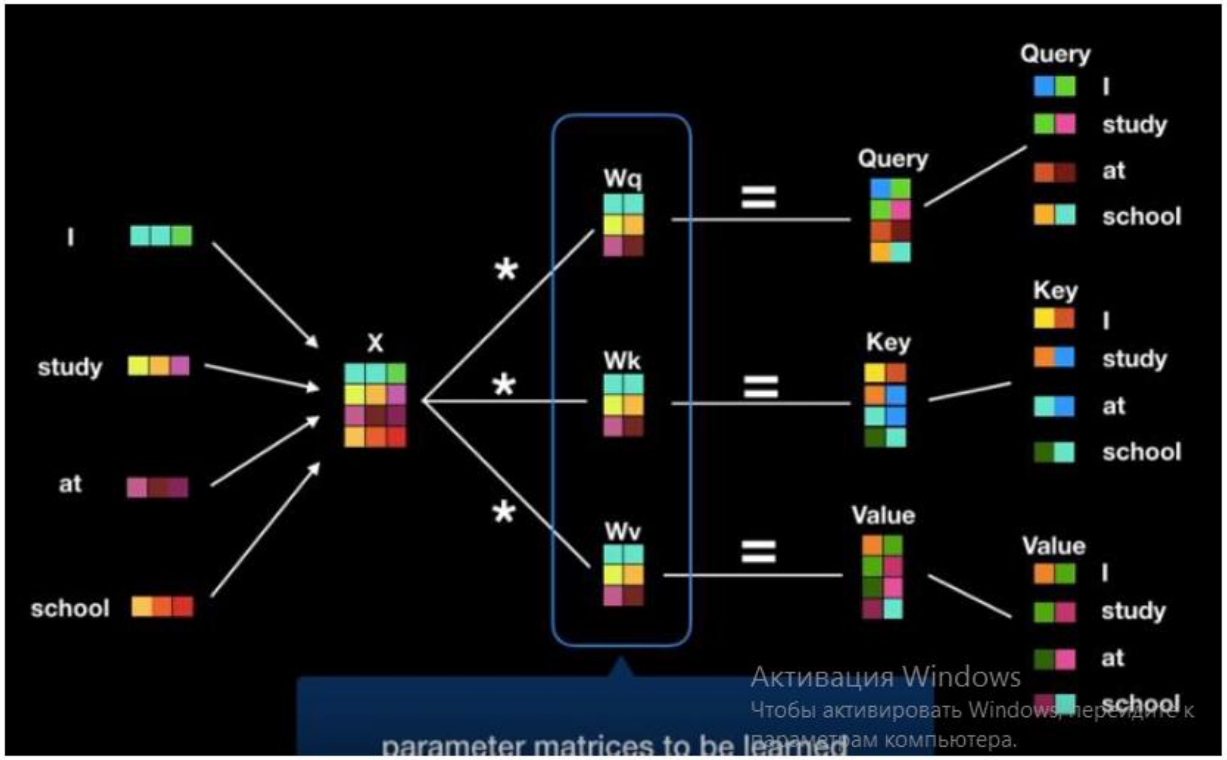
Transformer



FLASHBACK

Как считать self-attention

- Кладем наши эмбеддинги в матрицу X
- В процессе обучения получаем W^Q, W^K, W^V
- Умножаем X на матрицы W^Q, W^K, W^V → получаем Q, K, V



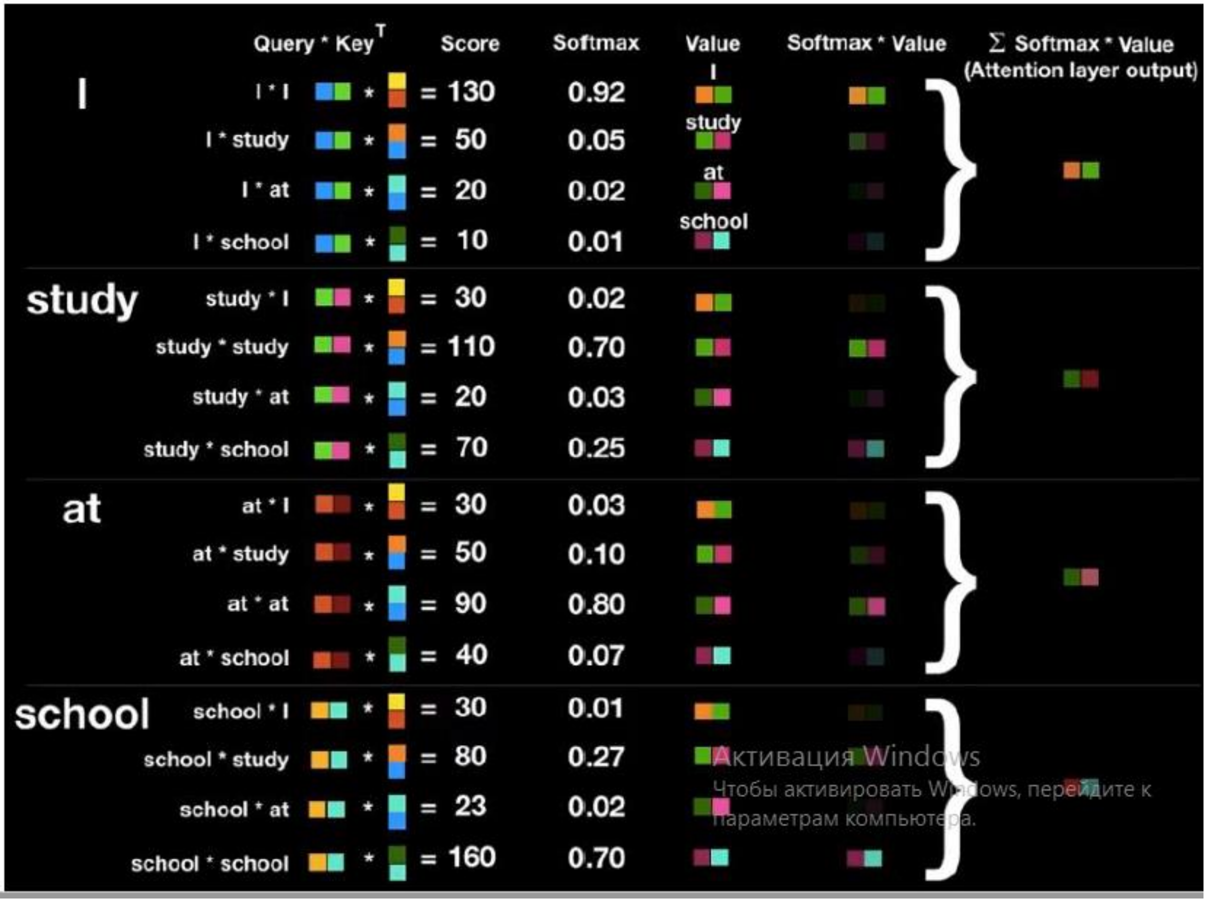
FLASHBACK

Scaled Dot-Product Attention

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

- Take the query
- Find the most similar key
- Get the values that correspond to these similar keys

- Softmax gives the probability distribution over keys, which are peaked at the ones that are similar to a query.

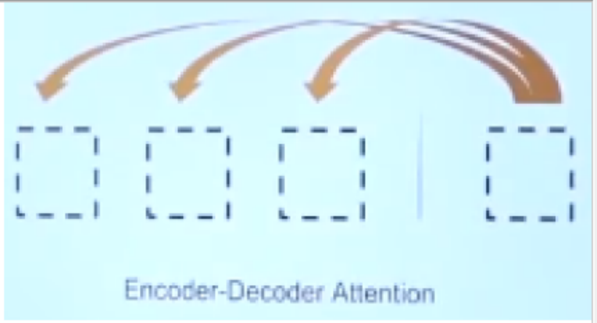


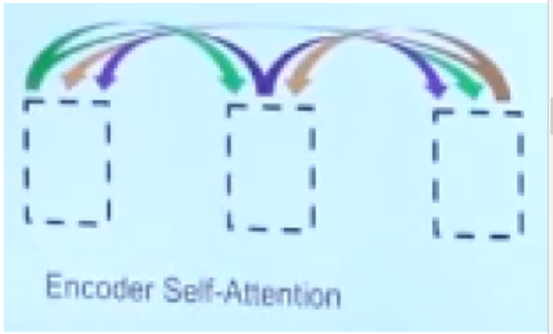
	Query * Key^T	Score	Softmax	Value	Softmax * Value	Σ Softmax * Value (Attention layer output)
I	I * I 🟦🟩 * 🟨🟧	= 130	0.92	I 🟧🟩	🟧🟩	🟧🟩
	I * study 🟦🟩 * 🟧🟦	= 50	0.05	study 🟩🟪	▪▪	
	I * at 🟦🟩 * 🟦🟦	= 20	0.02	at 🟩🟪	▪▪	
	I * school 🟦🟩 * 🟩🟦	= 10	0.01	school 🟪🟦	▪▪	
study	study * I 🟩🟪 * 🟨🟧	= 30	0.02	🟧🟩	▪▪	🟩🟥
	study * study 🟩🟪 * 🟧🟦	= 110	0.70	🟩🟪	🟩🟪	
	study * at 🟩🟪 * 🟦🟦	= 20	0.03	🟩🟪	▪▪	
	study * school 🟩🟪 * 🟩🟦	= 70	0.25	🟪🟦	🟪🟦	
at	at * I 🟥🟥 * 🟨🟧	= 30	0.03	🟧🟩	▪▪	🟩🟪
	at * study 🟥🟥 * 🟧🟦	= 50	0.10	🟩🟪	▪▪	
	at * at 🟥🟥 * 🟦🟦	= 90	0.80	🟩🟪	🟩🟪	
	at * school 🟥🟥 * 🟩🟦	= 40	0.07	🟪🟦	▪▪	
school	school * I 🟧🟦 * 🟨🟧	= 30	0.01	🟧🟩	▪▪	🟥🟦
	school * study 🟧🟦 * 🟧🟦	= 80	0.27	🟩🟪	▪▪	
	school * at 🟧🟦 * 🟦🟦	= 23	0.02	🟩🟪	▪▪	
	school * school 🟧🟦 * 🟩🟦	= 160	0.70	🟪🟦	🟪🟦	

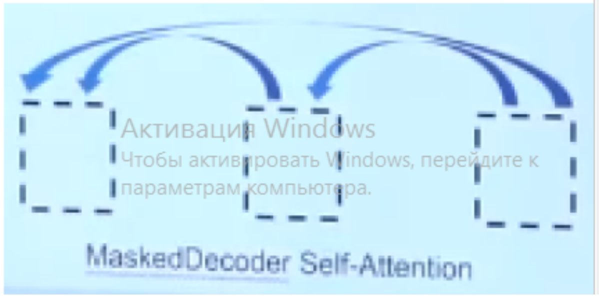
FLASHBACK

Three ways of attention

- **Encoder-decoder attention**: Q из предыдущего слоя декодера, K, V из output'a кодера.
  - Every position attends over all positions in the input sequence. This mimics the typical encoder-decoder attention mechanisms in seq2seq models.
- **Encoder self-attention**: K,V,Q из output'a предыдущего слоя кодера. Each position in the encoder can attend to all positions in the previous layer of the encoder.
- **Decoder self-attention**: each position in the decoder attends to all positions in the decoder up to and including that position.
  - We implement this inside of scaled dot-product attention by masking out (setting to −∞) all values in the input of the softmax which correspond to illegal connections.



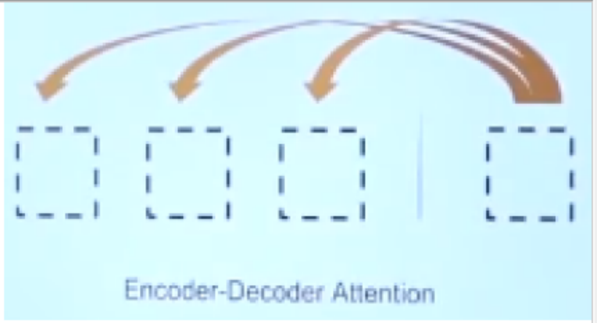


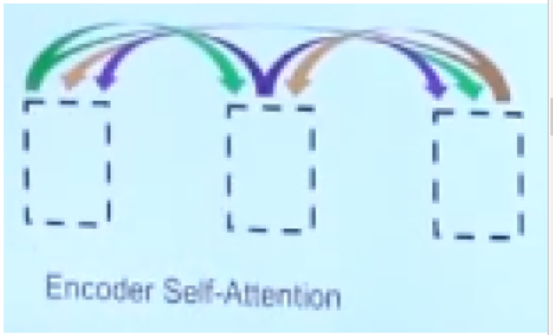


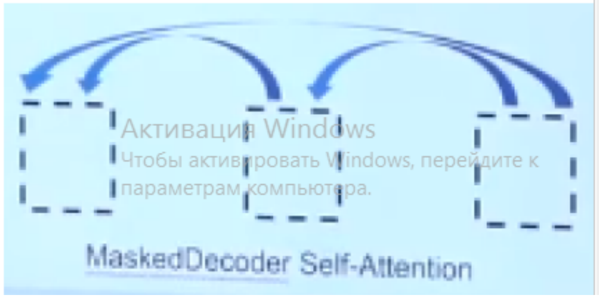
FLASHBACK

Three ways of attention

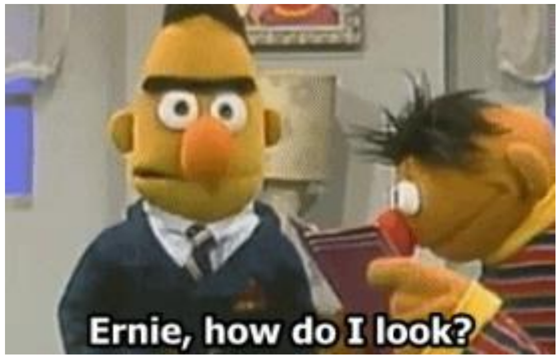
- **Encoder-decoder attention**: Q из предыдущего слоя декодера, K, V из output'а кодера.
  - Every position attends over all positions in the input sequence. This mimics the typical encoder-decoder attention mechanisms in seq2seq models.
- **Encoder self-attention**: K,V,Q из output'a предыдущего слоя кодера. Each position in the encoder can attend to all positions in the previous layer of the encoder.
- **Decoder self-attention**: each position in the decoder attends to all positions in the decoder up to and including that position.
  - We implement this inside of scaled dot-product attention by masking out (setting to −∞) all values in the input of the softmax which correspond to illegal connections.

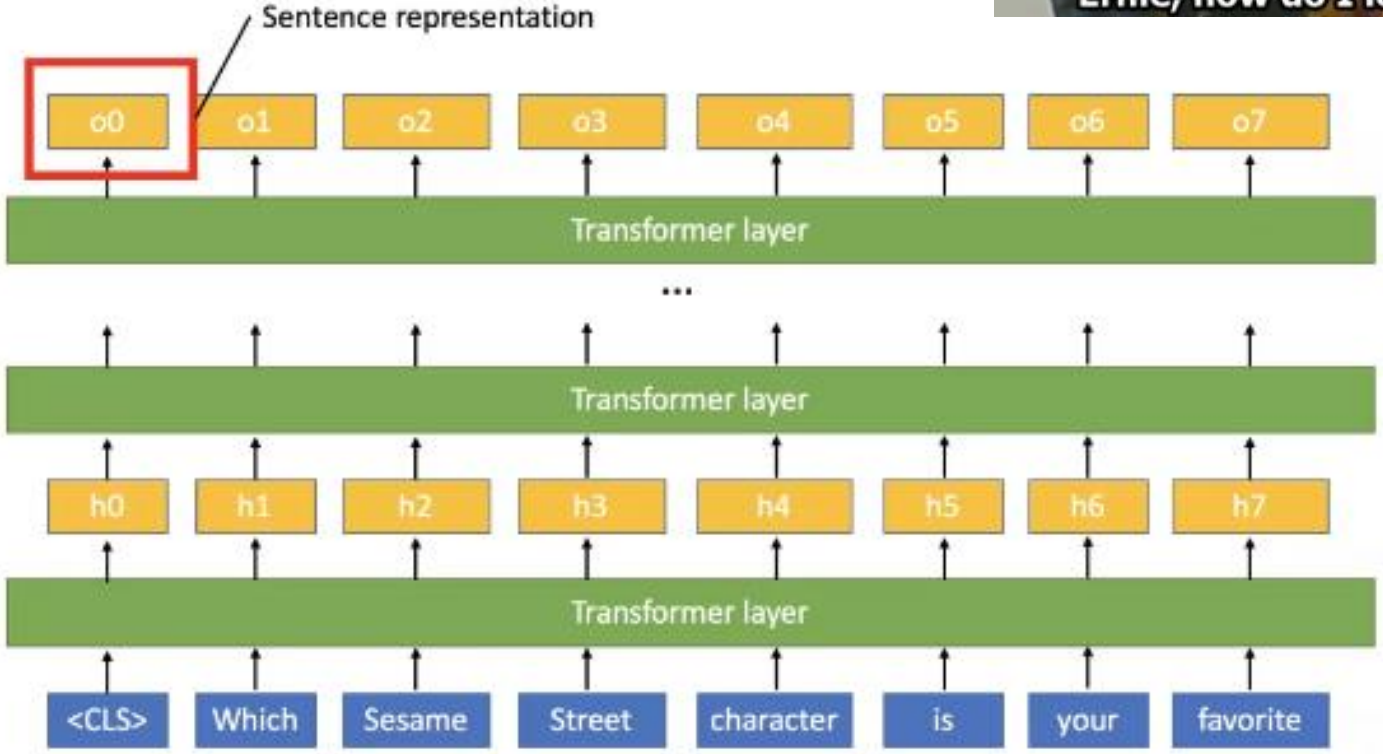






BERT

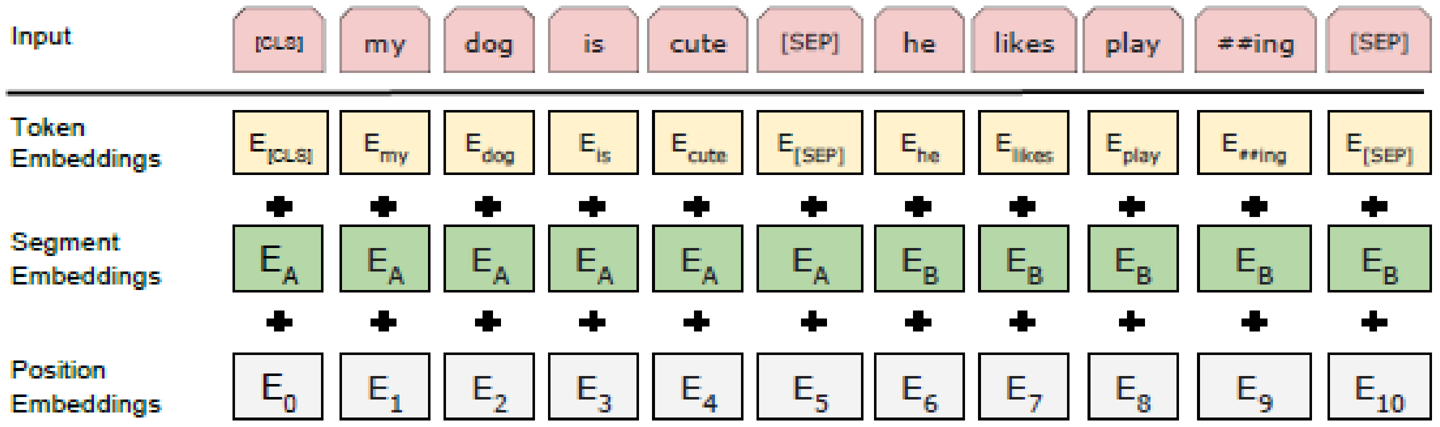




A basic conceptual diagram representing how BERT works

BERT

- WordPiece embeddings with a 30,000 token vocabulary.
- The first token of every sequence is always a special classification token ([CLS]).



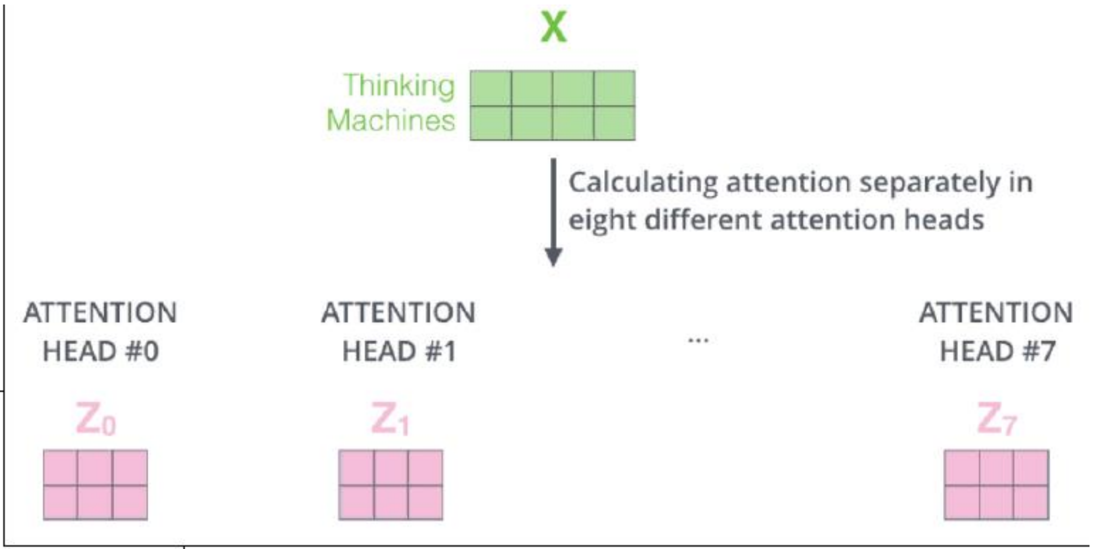
Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8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

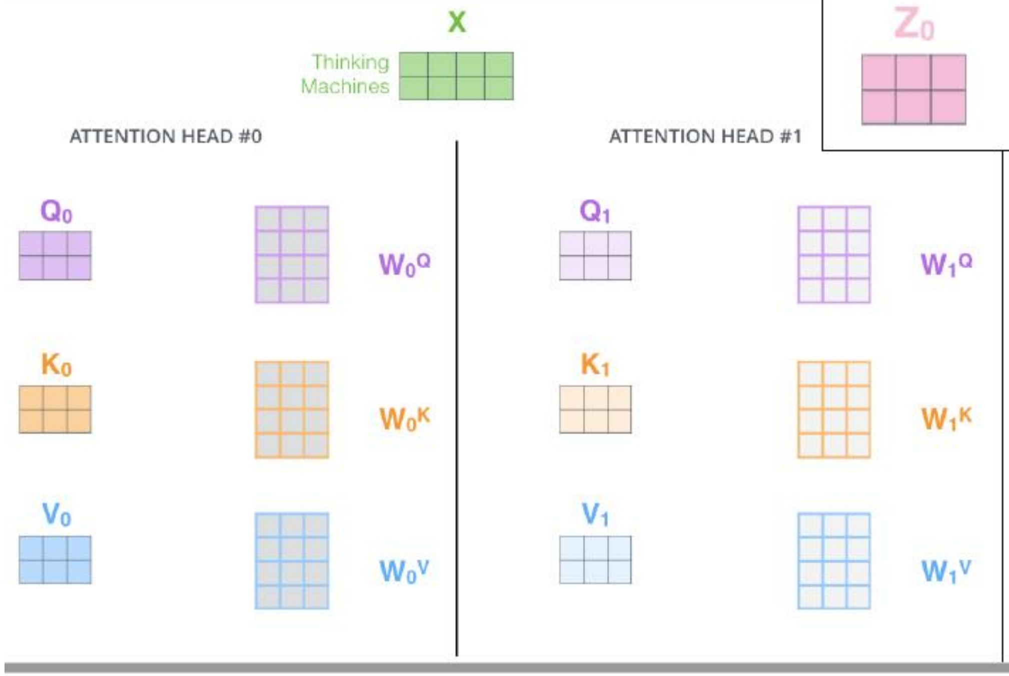
Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

BERT

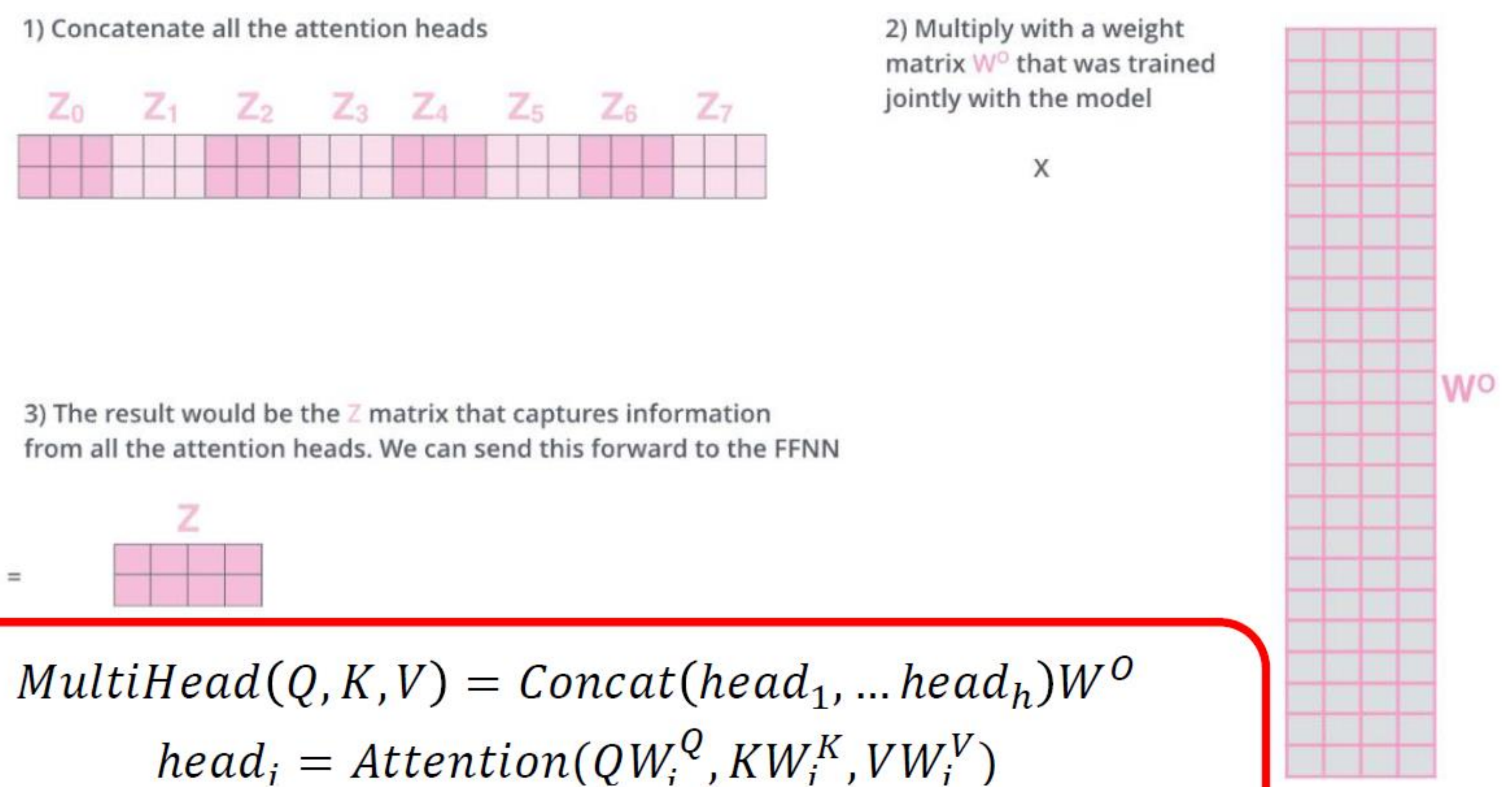
- BERT BASE (L=12, H=768, A=12, Total Parameters=110M)
- BERT LARGE (L=24, H=1024, A=16, Total Parameters=340M)
- number of layers (i.e., Transformer blocks) as L, the hidden size as H, and the number of self-attention heads as A

FLASHBACK


X
Thinking
Machines
Calculating attention separately in eight different attention heads
ATTENTION HEAD #0
ATTENTION HEAD #1
...
ATTENTION HEAD #7
Z_0
Z_1
Z_7


X
Thinking
Machines
ATTENTION HEAD #0
ATTENTION HEAD #1
Q_0
W_0^Q
K_0
W_0^K
V_0
W_0^V
Q_1
W_1^Q
K_1
W_1^K
V_1
W_1^V

FLASHBACK



1) Concatenate all the attention heads

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

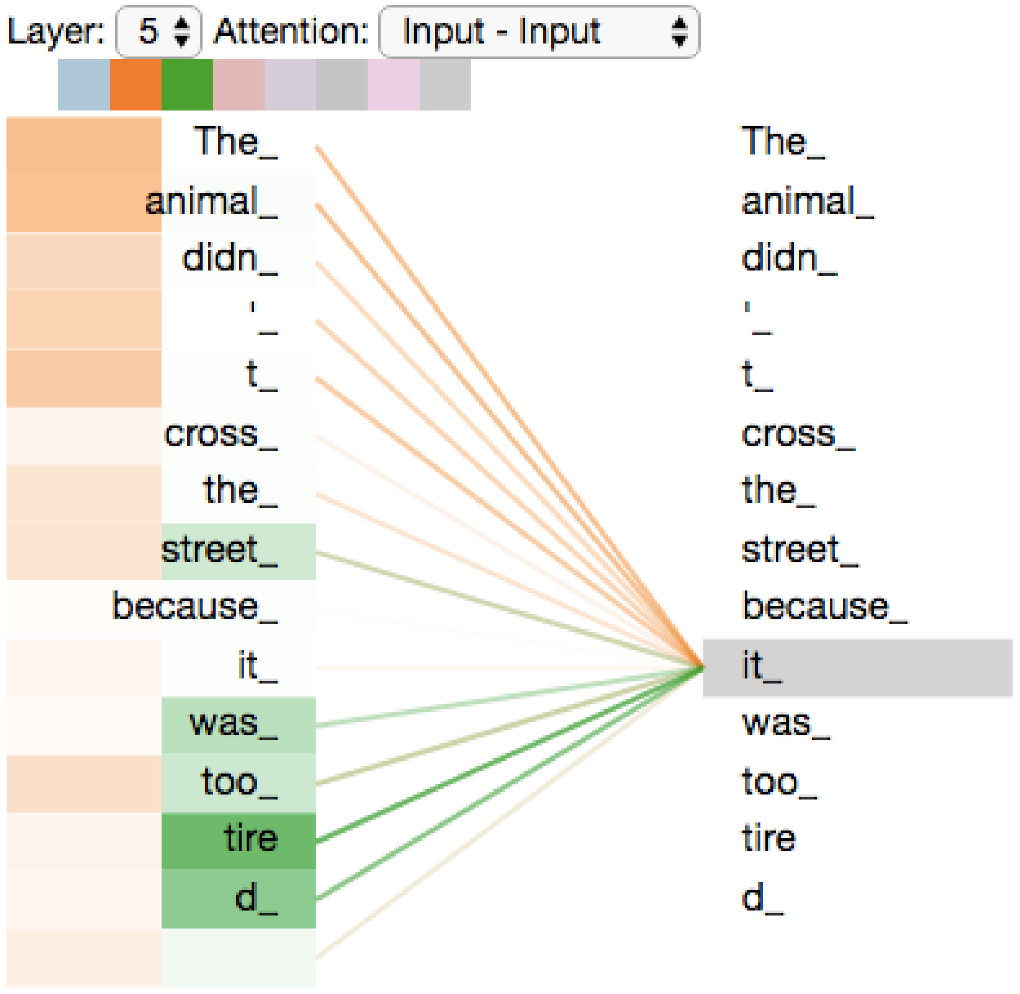
3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

$$MultiHead(Q,K,V) = Concat(head_1, \dots head_h)W^O$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

FLASHBACK


Layer: 5
Attention: Input - Input
The_
animal_
didn_
'_
t_
cross_
the_
street_
because_
it_
was_
too_
tire
d_
The_
animal_
didn_
'_
t_
cross_
the_
street_
because_
it_
was_
too_
tire
d_

BERT

- Нет смысла тренировать модель на обычной языковой модели.
- Будем использовать:

✓ Masked Language Model

✓ Next sentence prediction

Masked Language Model

- Просим модель предсказать не следующее слово, а случайное слово последовательности
- 15% заменяем на [MASK]

Input = `[CLS] the man went to [MASK] store [SEP]`

Masked Language Model

- If the model had been trained on only predicting '<MASK>' tokens and then never saw this token during fine-tuning, it would have thought that there was no need to predict anything and this would have hampered performance
- 80% заменяются на <MASK>

*"My dog is **<MASK>**"*

- 10% на случайное слово

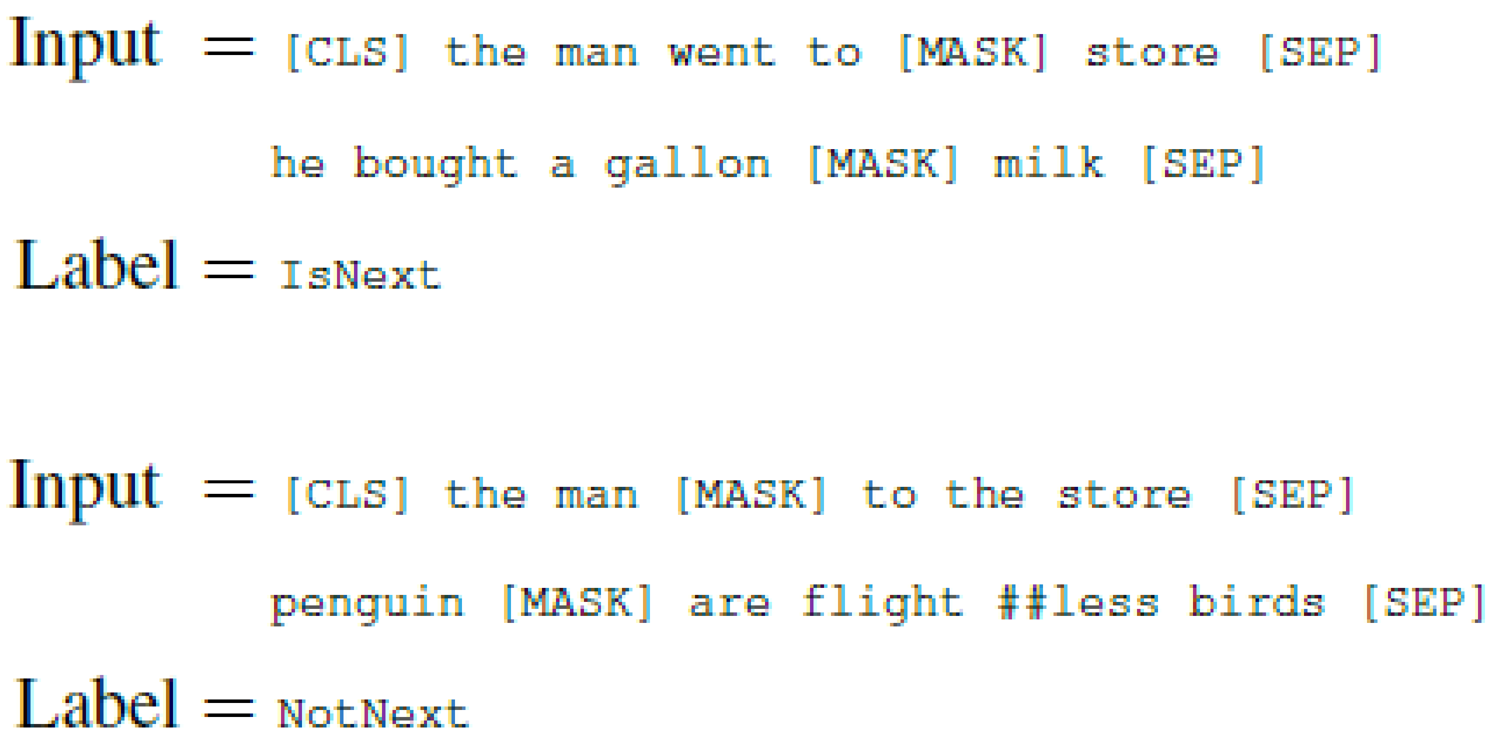
"My dog is **apple***"*

- 10% остаются на месте

"My dog is **hairy***"*

Next Sentence Prediction

- Хотим Question Answering (QA) and Natural Language Inference (NLI)
- Нужно понимать отношения между предложениями



Input = [CLS] the man went to [MASK] store [SEP]

he bought a gallon [MASK] milk [SEP]

Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]

penguin [MASK] are flight ##less birds [SEP]

Label = NotNext

Pre-training

- The pre-training corpus was built from

❑ BookCorpus (800M words)

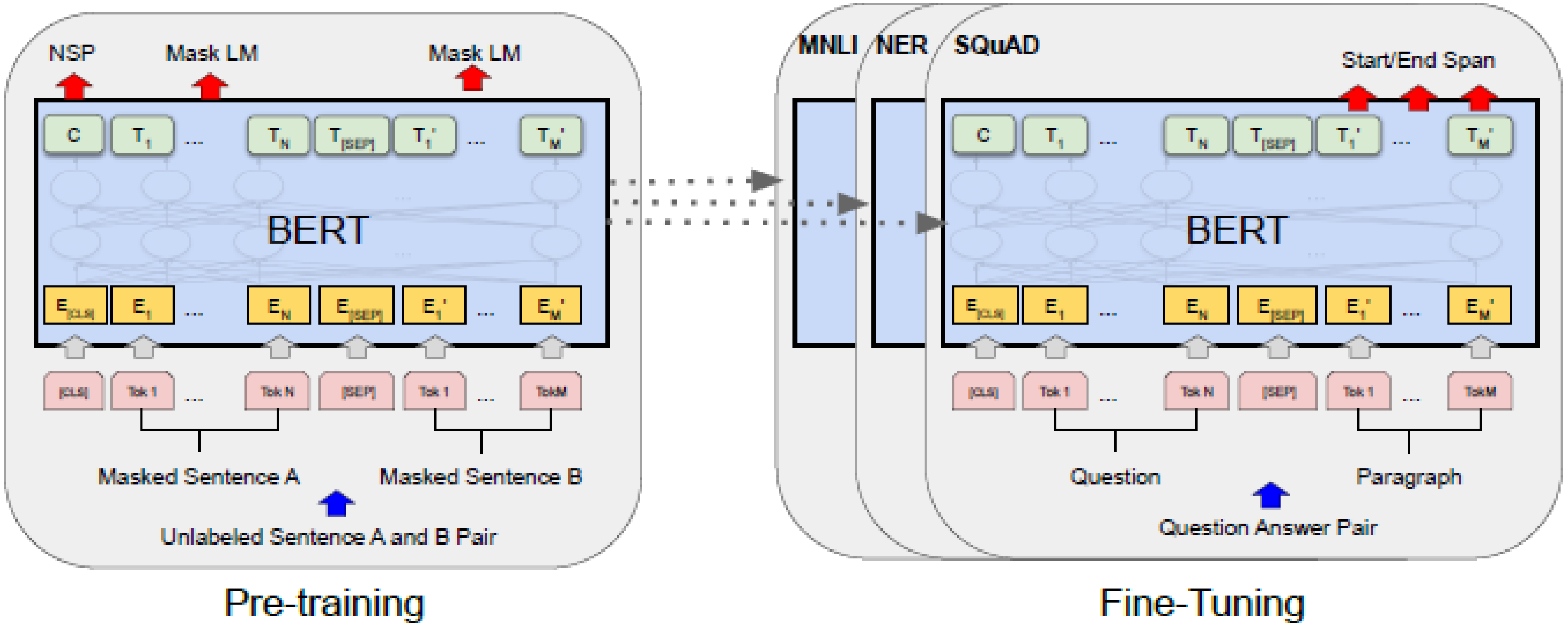
❑ English Wikipedia (2,500M words).

❑ Tokens were tokenized using 30,000 WordPiece tokens.

Fine-tuning

- For each task, we simply plug in the taskspecific inputs and outputs into BERT and finetune all the parameters end-to-end.
- At the output, the token representations are fed into an output layer for token level tasks, such as sequence tagging or question answering, and the [CLS] representation is fed into an output layer for classification, such as entailment or sentiment analysis.

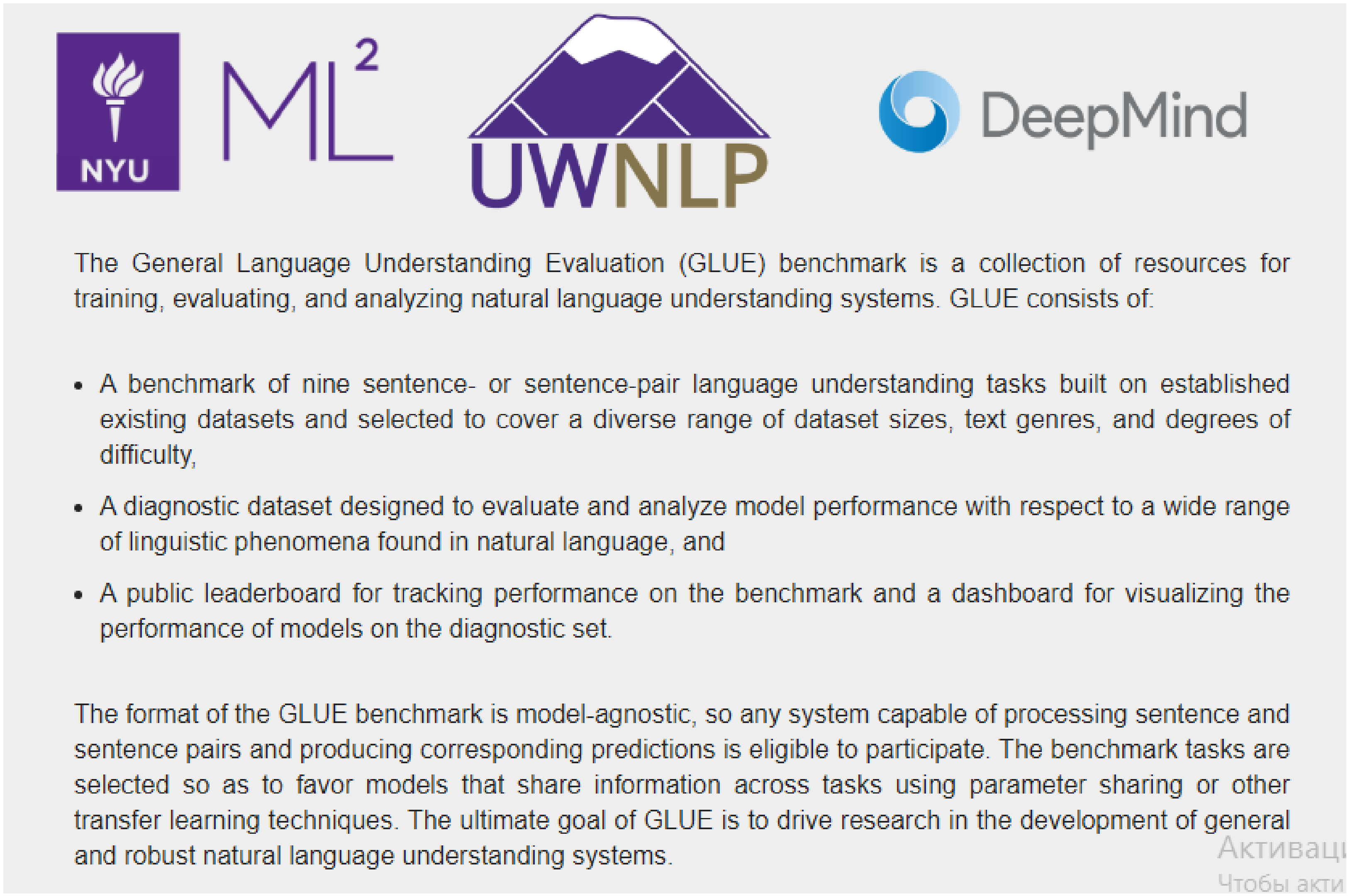
Fine-tuning


NSP
Mask LM
Mask LM
C
T_1
...
T_N
$T_{[SEP]}$
T_1'
...
T_M'
BERT
$E_{[CLS]}$
E_1
...
E_N
$E_{[SEP]}$
E_1'
...
E_M'
[CLS]
Tok 1
...
Tok N
[SEP]
Tok 1
...
TokM
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
Pre-training
MNLI
NER
SQuAD
Start/End Span
BERT
Question
Paragraph
Question Answer Pair
Fine-Tuning

И как работает?

- 11 NLP tasks
- The General Language Understanding Evaluation (GLUE) benchmark
- Collection of diverse natural language understanding tasks

И как работает?

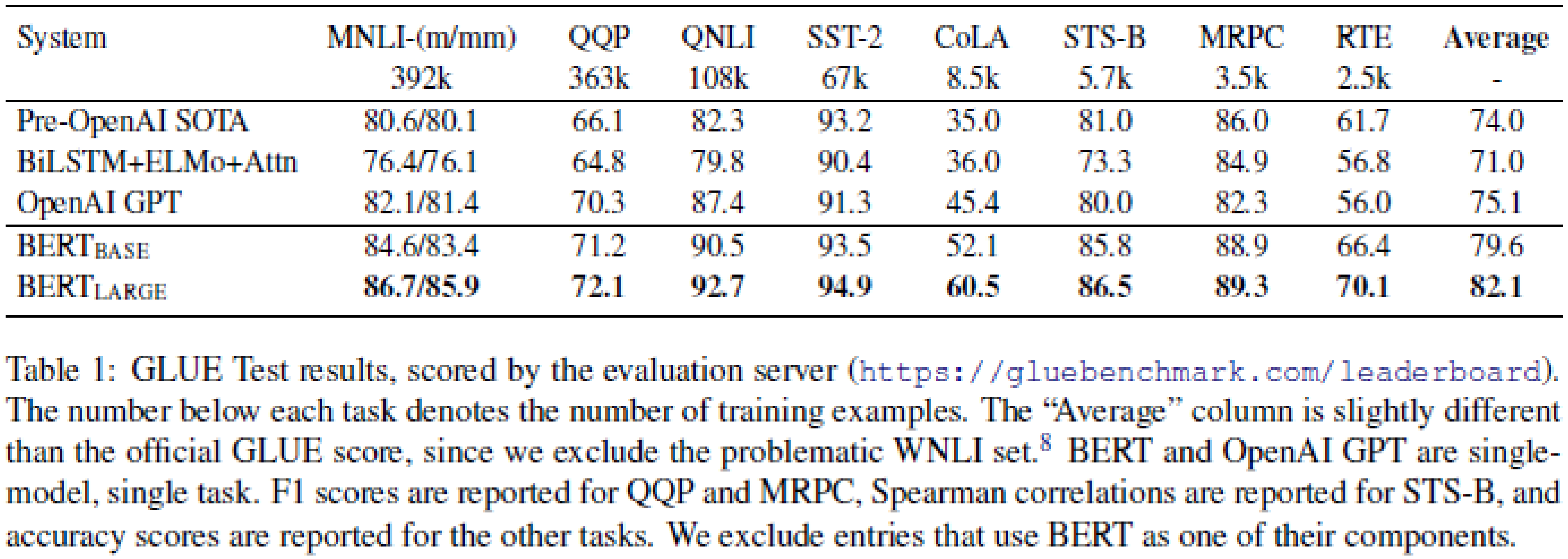


The General Language Understanding Evaluation (GLUE) benchmark is a collection of resources for training, evaluating, and analyzing natural language understanding systems. GLUE consists of:

- A benchmark of nine sentence- or sentence-pair language understanding tasks built on established existing datasets and selected to cover a diverse range of dataset sizes, text genres, and degrees of difficulty,
- A diagnostic dataset designed to evaluate and analyze model performance with respect to a wide range of linguistic phenomena found in natural language, and
- A public leaderboard for tracking performance on the benchmark and a dashboard for visualizing the performance of models on the diagnostic set.

The format of the GLUE benchmark is model-agnostic, so any system capable of processing sentence and sentence pairs and producing corresponding predictions is eligible to participate. The benchmark tasks are selected so as to favor models that share information across tasks using parameter sharing or other transfer learning techniques. The ultimate goal of GLUE is to drive research in the development of general and robust natural language understanding systems.

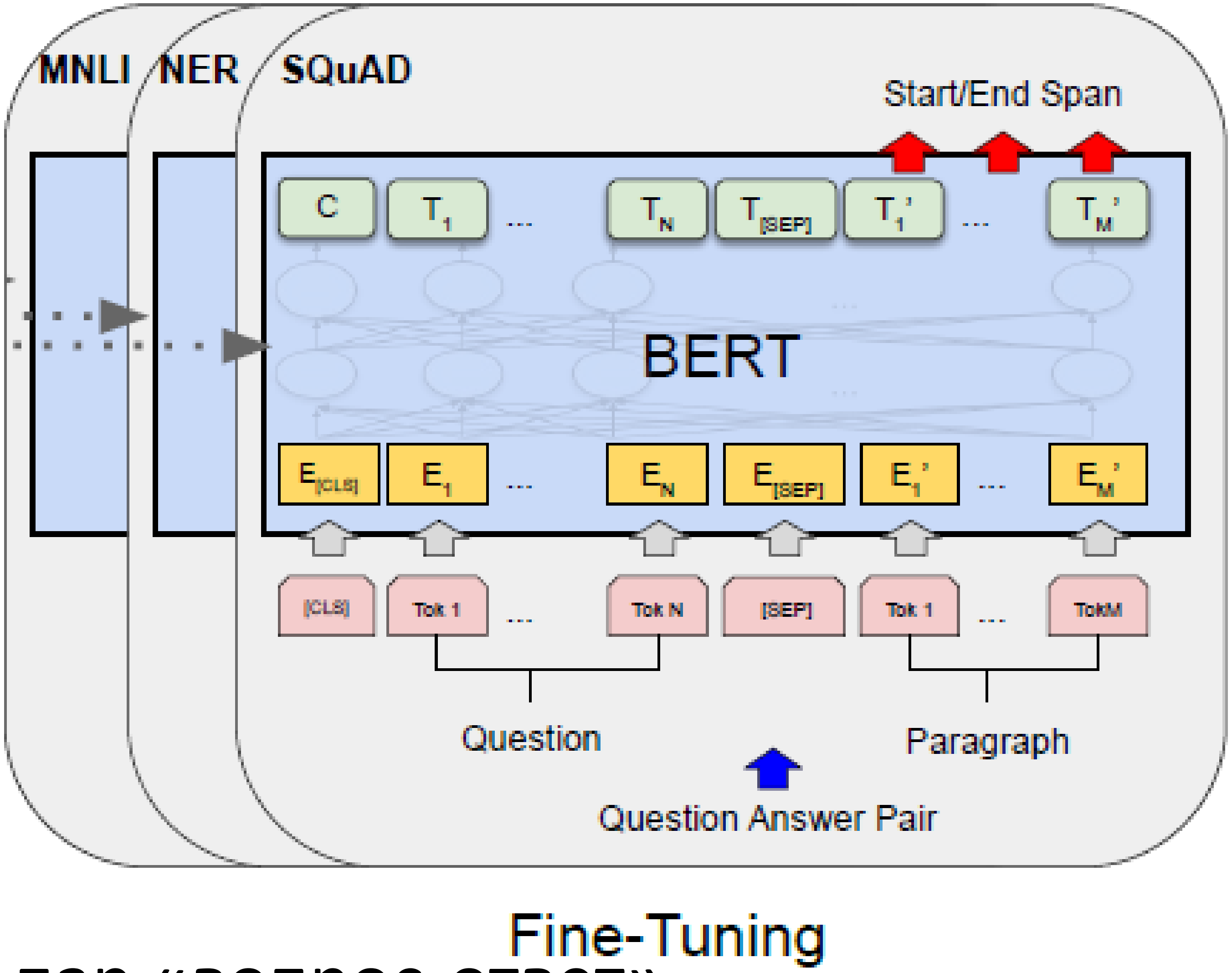
И как работает?



System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Average
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT$_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 1: GLUE Test results, scored by the evaluation server (`https://gluebenchmark.com/leaderboard`). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

SQUAD

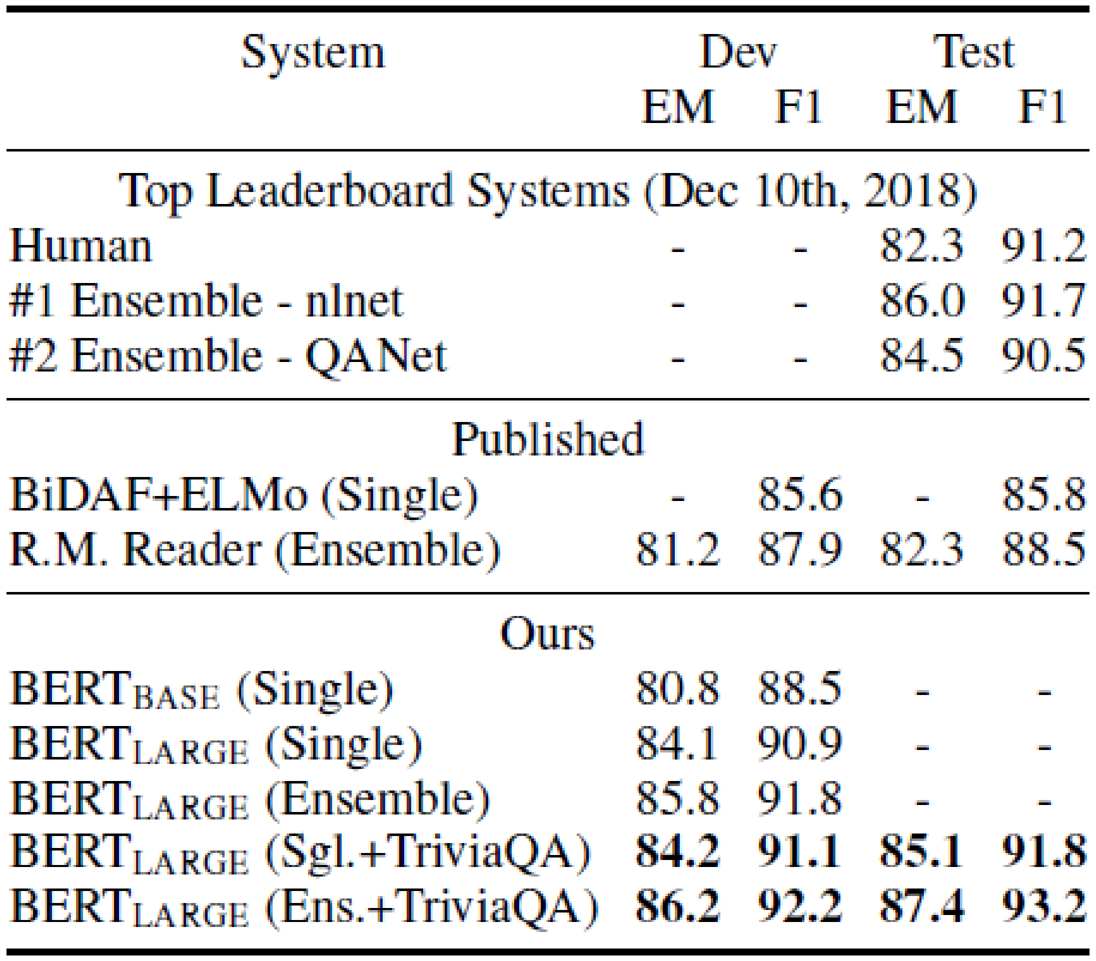


- 100 тысяч пар «вопрос-ответ»
- Дается вопрос и кусок текста из Wiki
- Нужно найти ответ в тексте

SQUAD

- We only introduce a start vector $S \in R^H$ and an end vector $E \in R^H$ during fine-tuning.
- The probability of word *i* being the start of the answer span $$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}.$$
- The analogous formula is used for the end of the
- answer span.
- The score of a candidate span from
- position *i* to position *j* is defined as $S*T_i + E*T_j$, and the maximum scoring span where $j \geq i$ is used as a prediction.

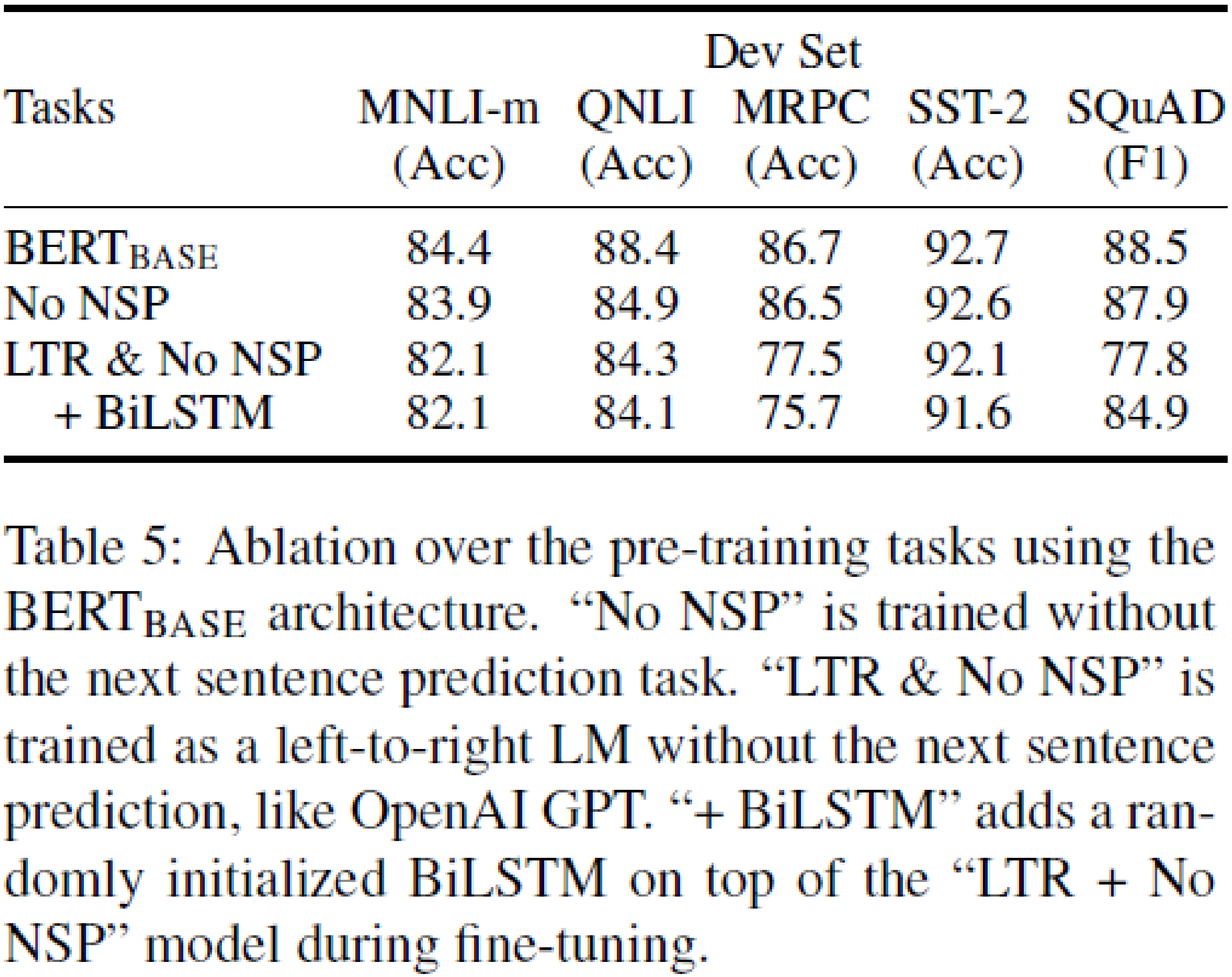
SQUAD



System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Table 2: SQuAD 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

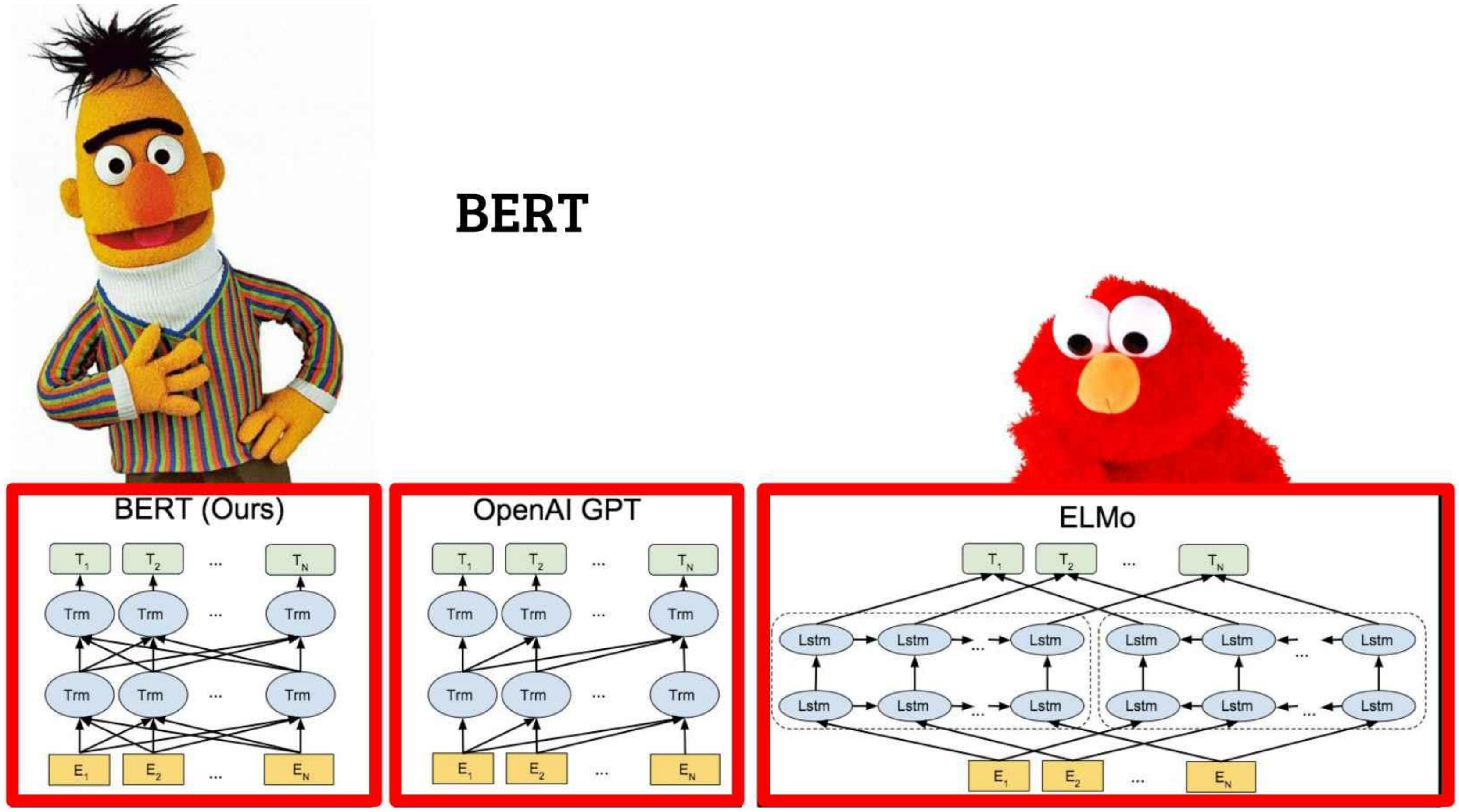
Ablation Studies



	Dev Set				
Tasks	MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
BERT$_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the BERT$_{\text{BASE}}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

Спасибо!


BERT
BERT (Ours)
OpenAI GPT
ELMo
T1 T2 ... TN
Trm
Lstm
E1 E2 ... EN