BERT: model, analysis and modifications

Ekaterina Lobacheva

NATIONAL RESEARCH UNIVERSITY

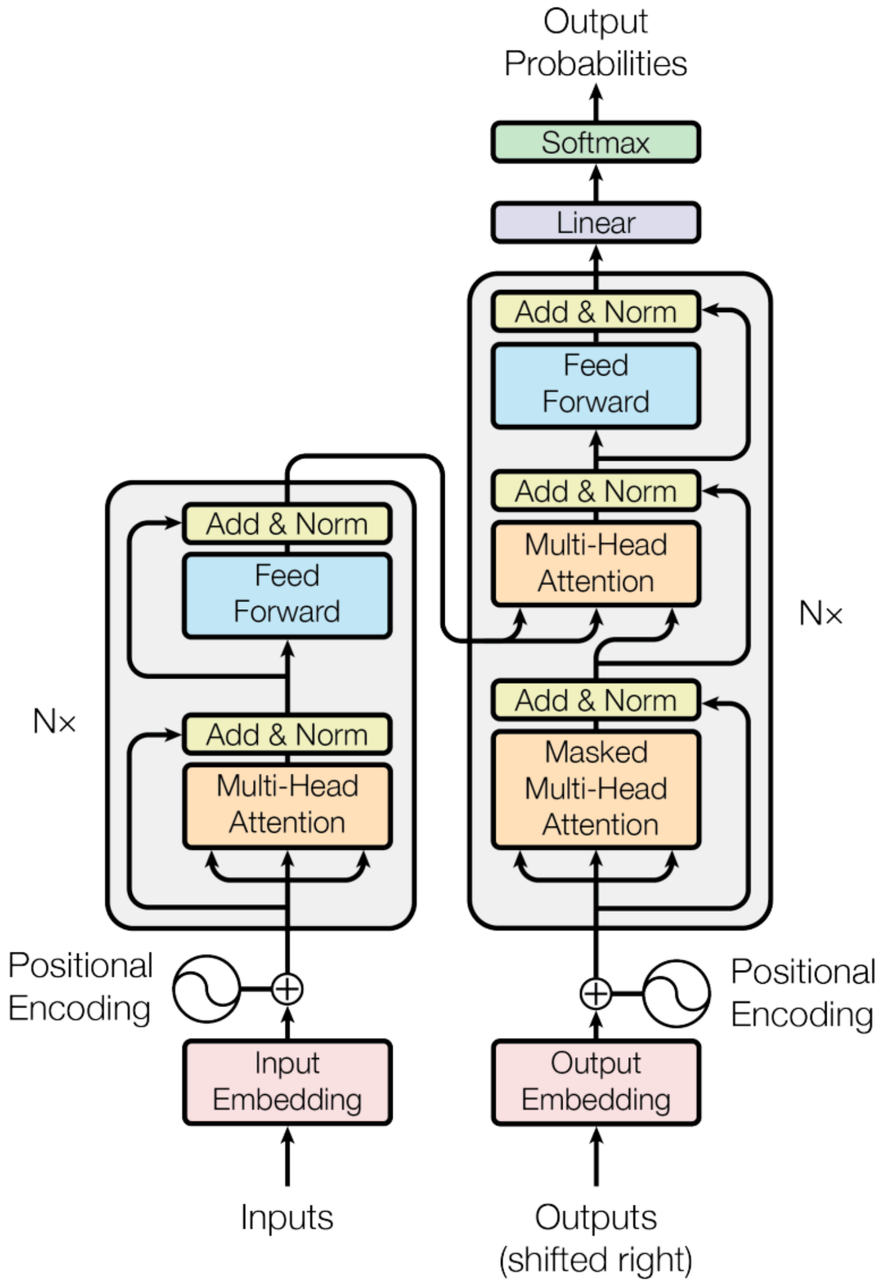
SAMSUNG Research

p(B|A)yesgroup.ru

Basics: Transformer

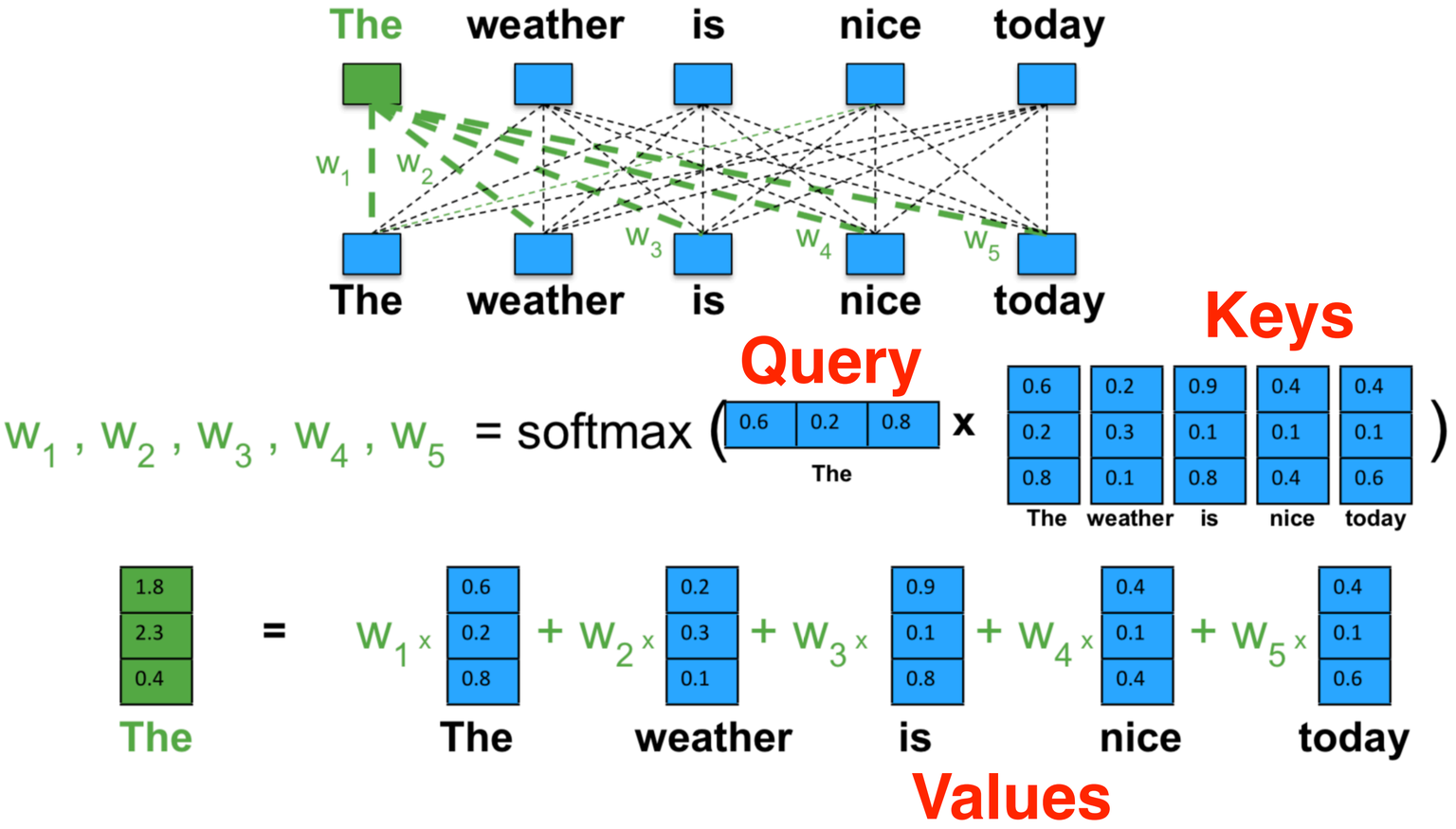
Transformer

- Seq2seq task (NMT)
- Encoder-decorder
- Attention:
  - Multi-head attention
  - Self-attention
  - Enc-dec attention
  - Mask in the decoder
- Position-wise ff-layers
- Residual connections
- Layer norm
- Positional encoding



Transformer

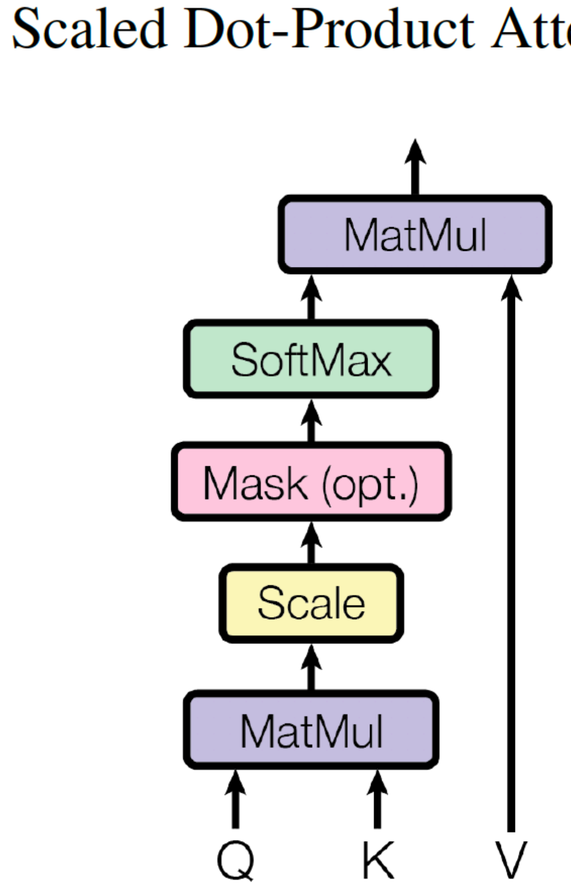
- Seq2seq task (NMT)
- Encoder-decorder
- **Attention**:
  - Multi-head attention
  - Self-attention
  - Enc-dec attention
  - Mask in the decoder
- Position-wise ff-layers
- Residual connections
- Layer norm
- Positional encoding



http://fuyw.top/NLP_02_QANet/

Transformer

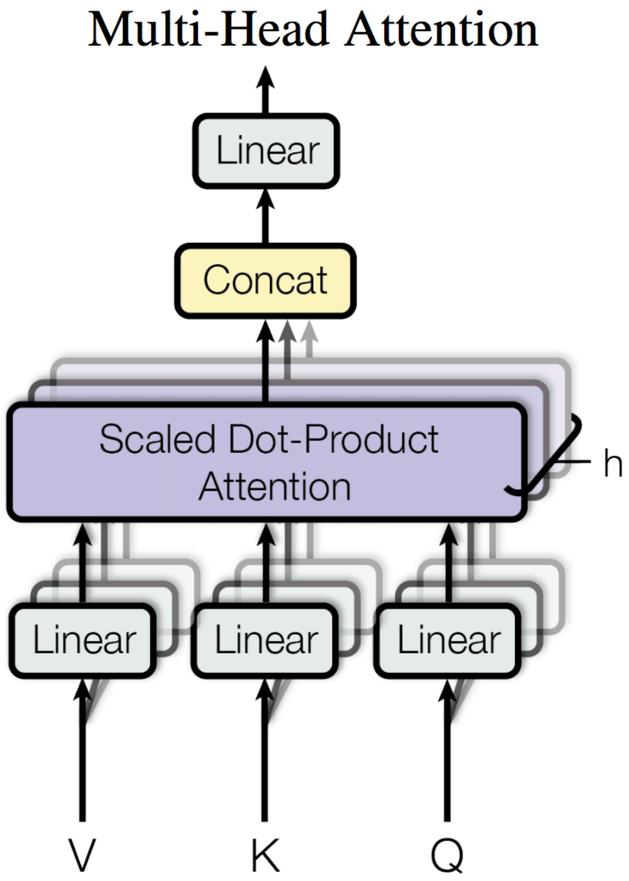
- Seq2seq task (NMT)
- Encoder-decorder
- **Attention**:
  - Multi-head attention
  - Self-attention
  - Enc-dec attention
  - Mask in the decoder
- Position-wise ff-layers
- Residual connections
- Layer norm
- Positional encoding



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

Transformer

- Seq2seq task (NMT)
- Encoder-decorder
- Attention:
  - **Multi-head attention**
  - Self-attention
  - Enc-dec attention
  - Mask in the decoder
- Position-wise ff-layers
- Residual connections
- Layer norm
- Positional encoding

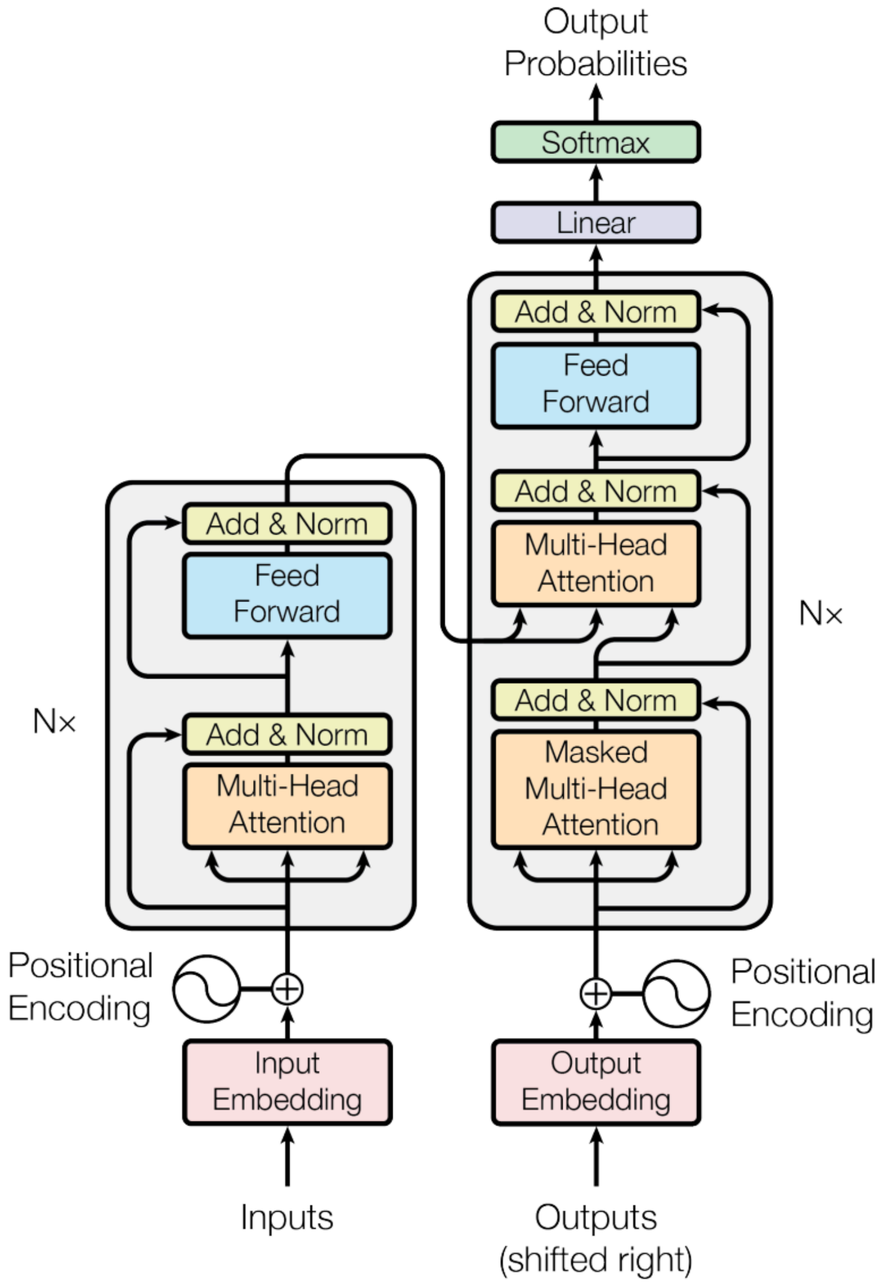


$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$

$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

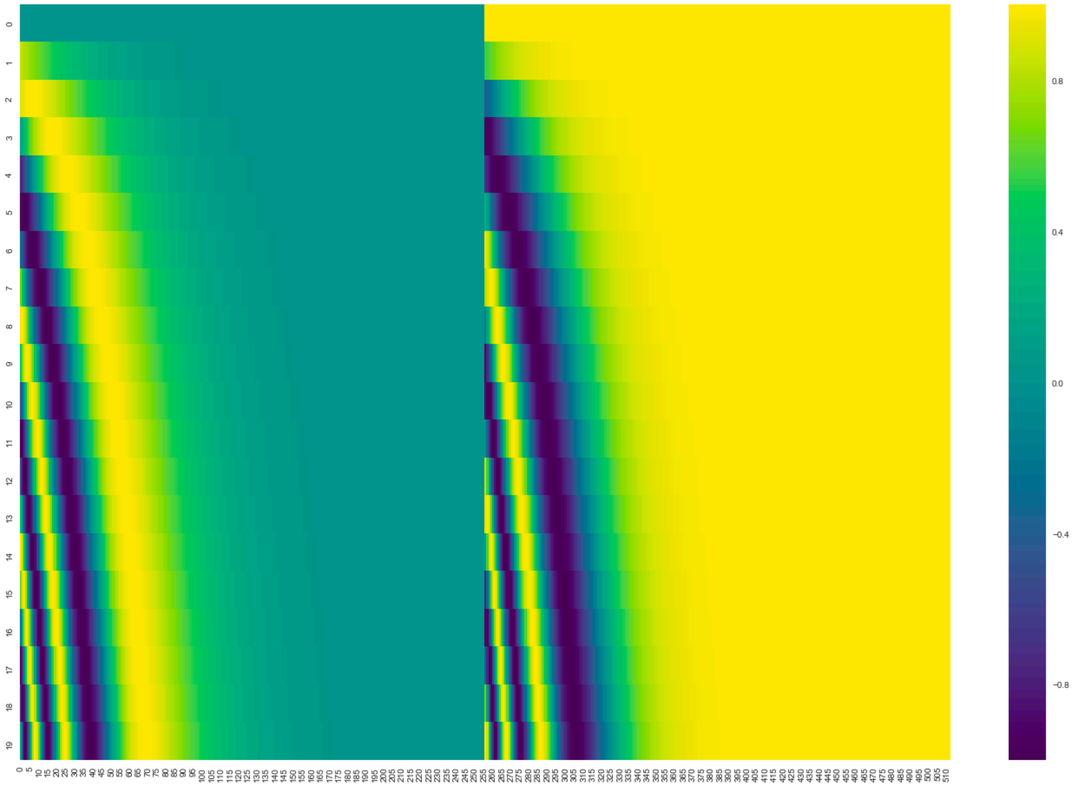
Transformer

- Seq2seq task (NMT)
- Encoder-decorder
- Attention:
  - Multi-head attention
  - **Self-attention**
  - **Enc-dec attention**
  - **Mask in the decoder**
- Position-wise ff-layers
- Residual connections
- Layer norm
- Positional encoding



Transformer

- Seq2seq task (NMT)
- Encoder-decorder
- Attention:
  - Multi-head attention
  - Self-attention
  - Enc-dec attention
  - Mask in the decoder
- Position-wise ff-layers
- Residual connections
- Layer norm
- **Positional encoding**



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

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

http://jalammar.github.io/illustrated-transformer/

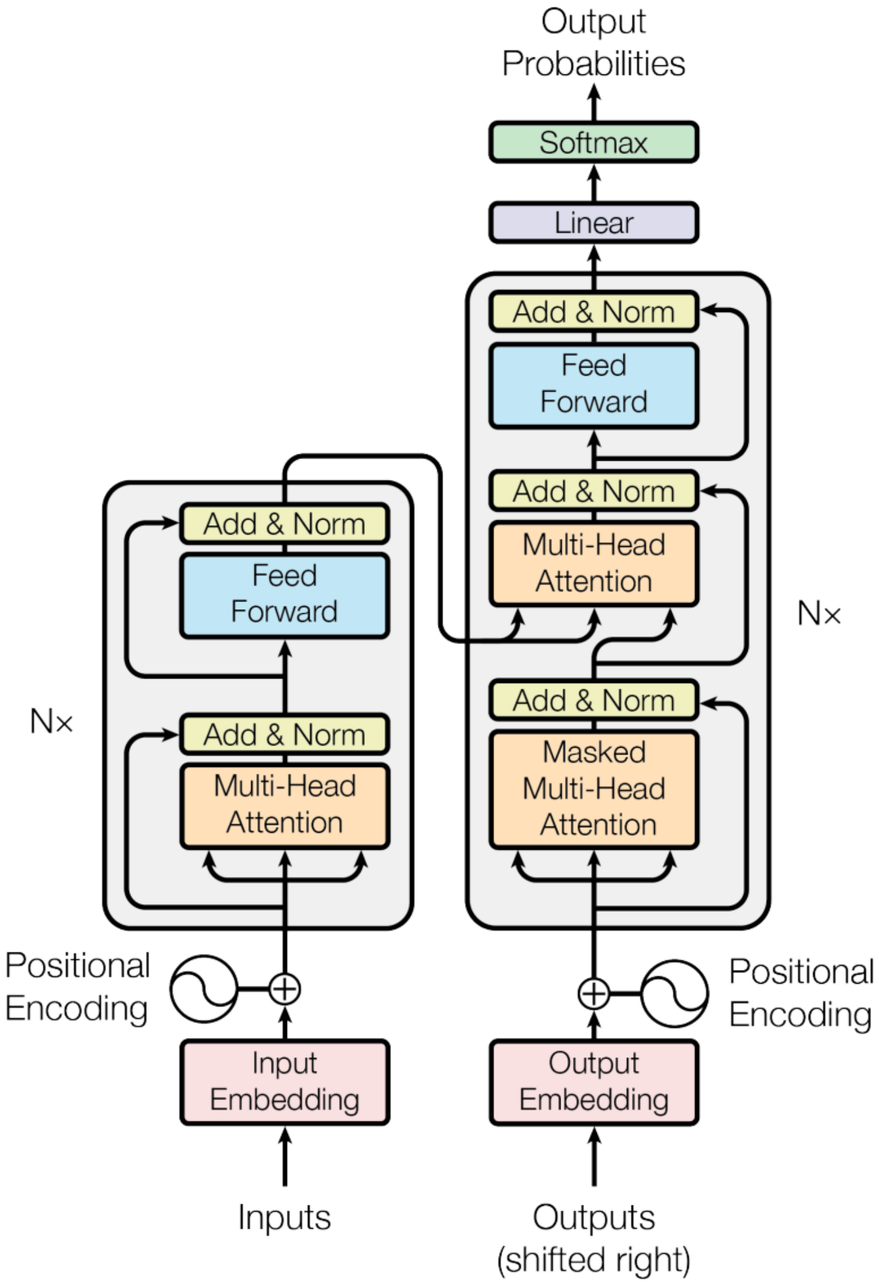
Transformer size

- Number of layers (blocks) in encoder/ decoder
- Hidden dimensionality
- Number of heads
- Dimensionality of inner feed-forward layers (usually 4 x hidden dimensionality)

Standard model:

6+6 layers, H = 512, 8 heads

65M params



BERT

Bidirectional Encoder Representations from Transformers

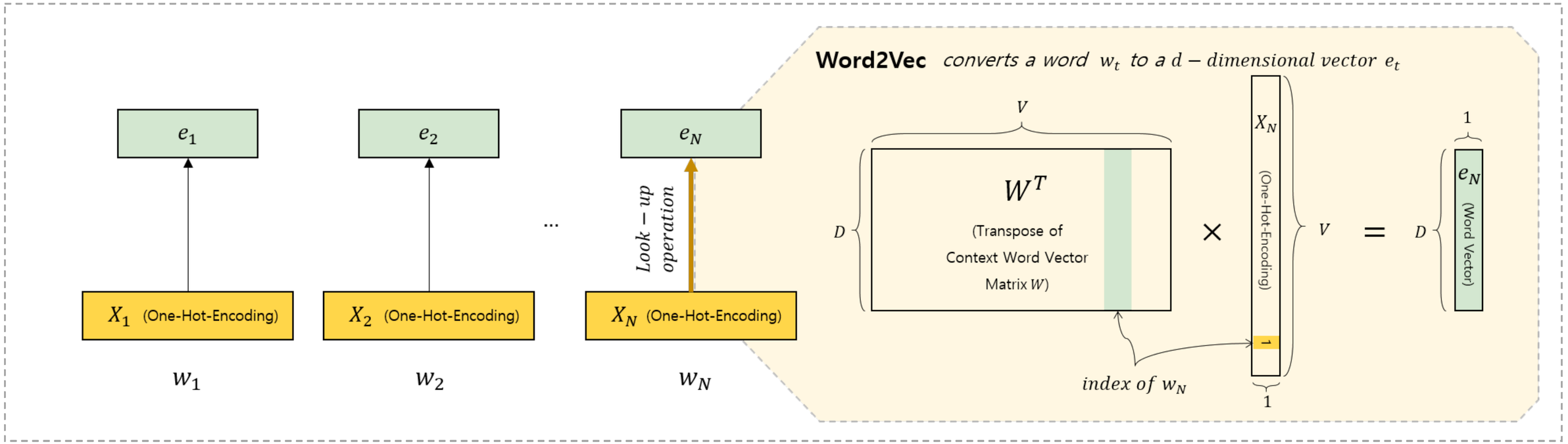
Embeddings in NLP

- Pretraining of token embeddings (unsupervised as LM or supervised on the task with a lot of data)
- Incorporating embeddings into a model for a target task
- Training of the target task model:
  - feature-based: use embeddings as additional features
  - fine-tuning: fine-tune embeddings

Important for fine-tuning:

- enough data for the target task
- architecture need to be applicable both to LM and to the target task

Individual embeddings



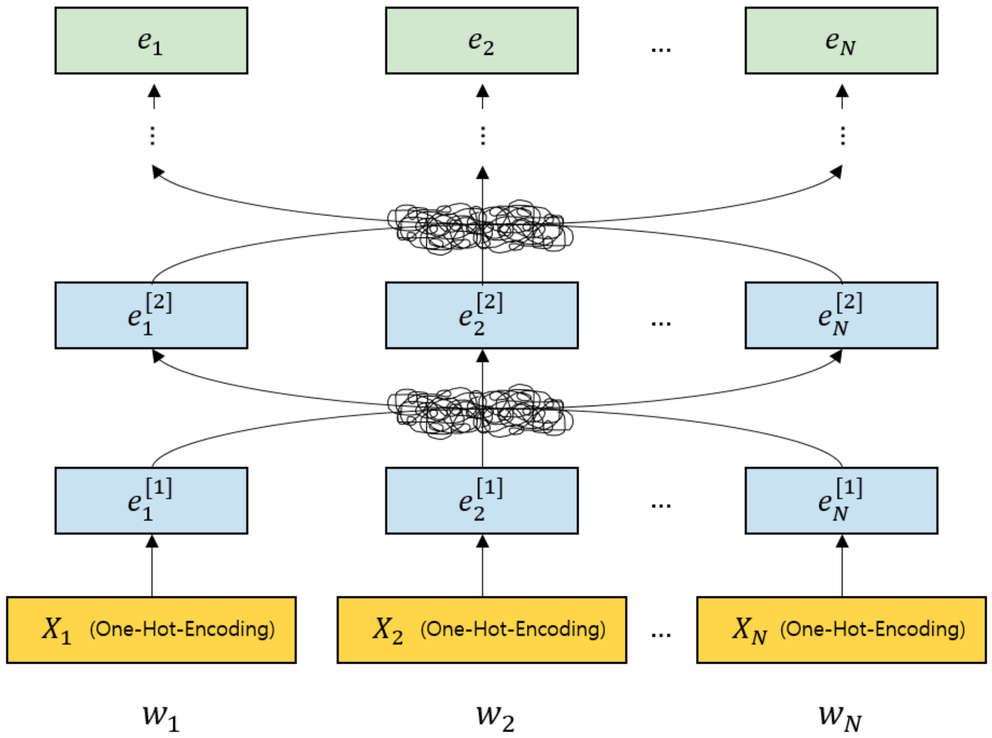
Wor2Vec, Glove, fastText …

http://intelligence.korea.ac.kr/members/wschoi/nlp/deeplearning/paperreview/Contextualized-Word-Embedding/

Contextualized embeddings

Contextualized Word Representations

converts a sequence of words (more precisely, tokens) $w_1 w_2 \dots w_N$ *to a series of* $d-$ *dimensional vectors* $e_1 e_2, \dots e_N$



Output:
series of $d-$ *dimensional vectors* $e_1 e_2, \dots e_N$

deep neural network
(e.g. deep bi-LSTM or deep-self-attention Nets)
for extracting features

simple word (or token) embedding
+ (optional) positional embedding

CoVe, ELMo, BERT, GPT/GPT-2 …

http://intelligence.korea.ac.kr/members/wschoi/nlp/deeplearning/paperreview/Contextualized-Word-Embedding/

Contextualised embeddings

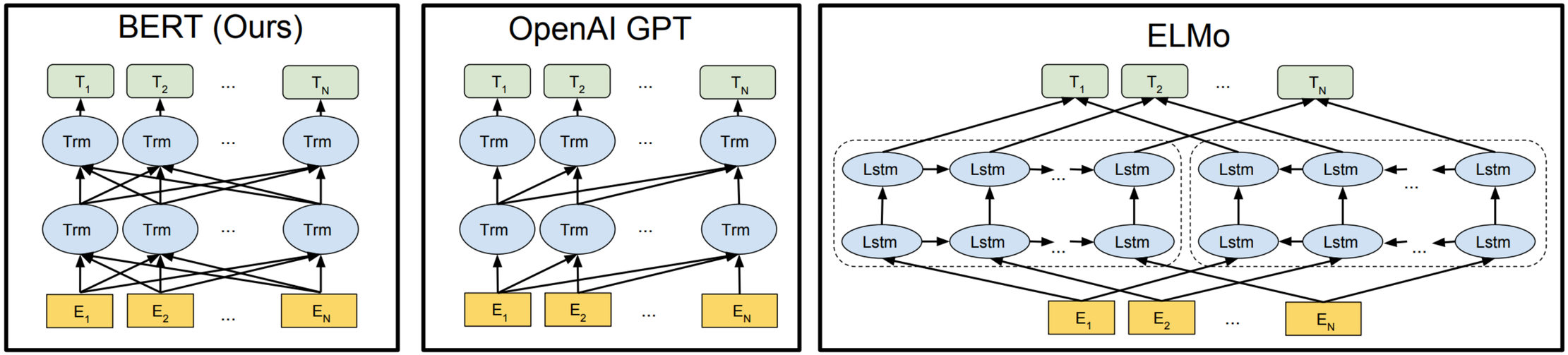
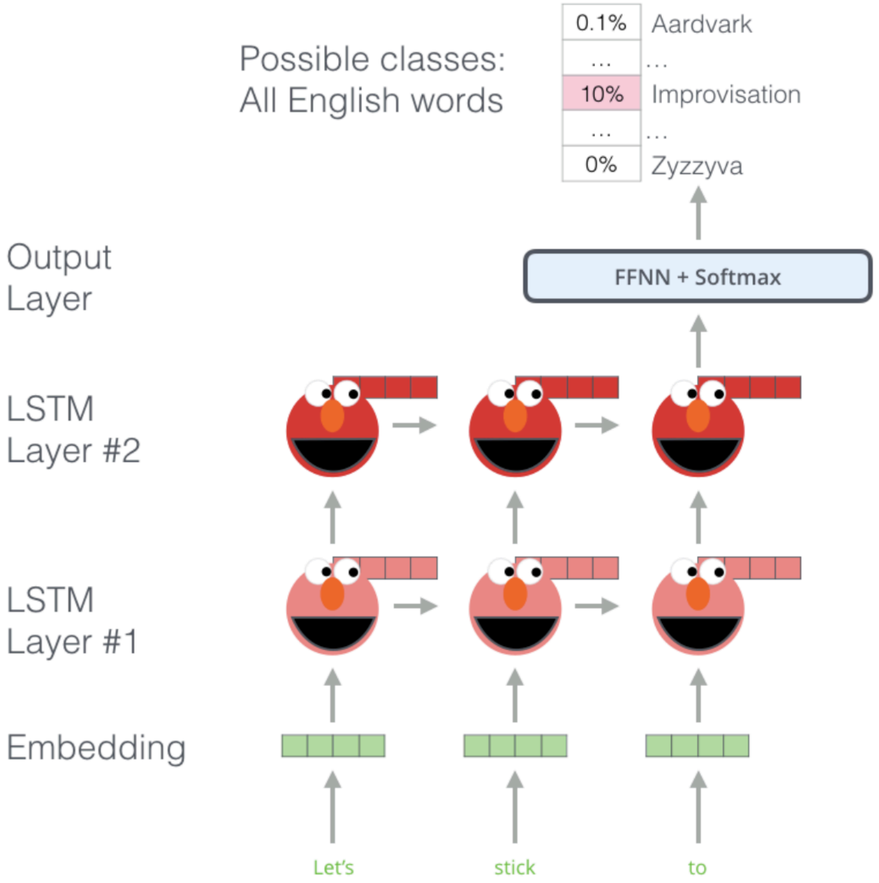


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT representations are jointly conditioned on both left and right context in all layers. In addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

ELMO: pre-training



- Use 2 LSTMs: forward and backward
- Params of embeddings and softmax are shared
- Optimize log likelihood:

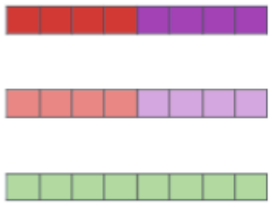
$$\sum_{k=1}^{N} (\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s))$$

http://jalammar.github.io/illustrated-bert/

ELMO: embeddings

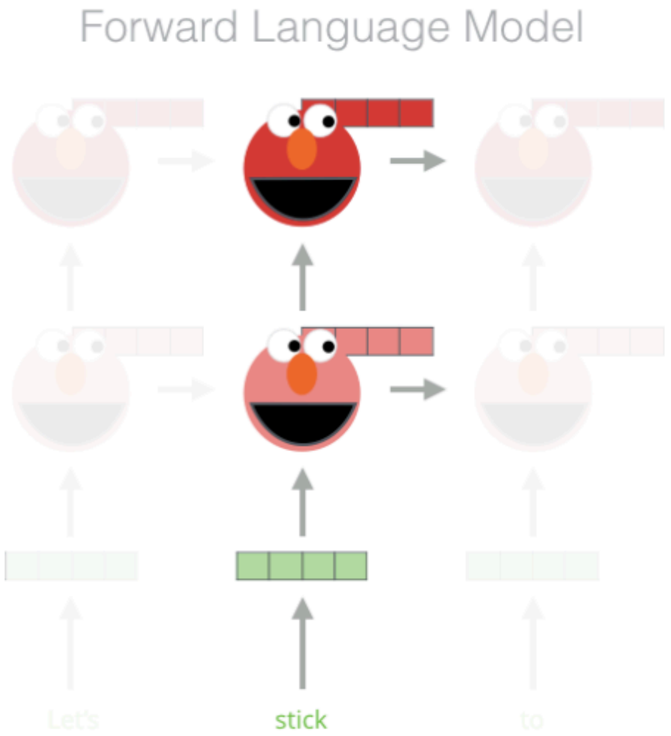
1- Concatenate hidden layers

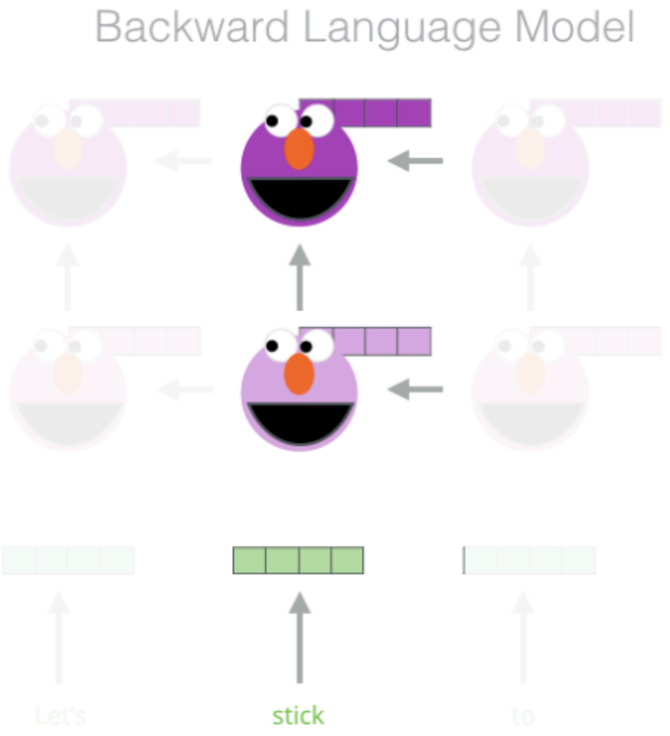
2- Multiply each vector by a weight based on the task



3- Sum the (now weighted) vectors

ELMo embedding of "stick" for this task in this context





http://jalammar.github.io/illustrated-bert/

ELMO

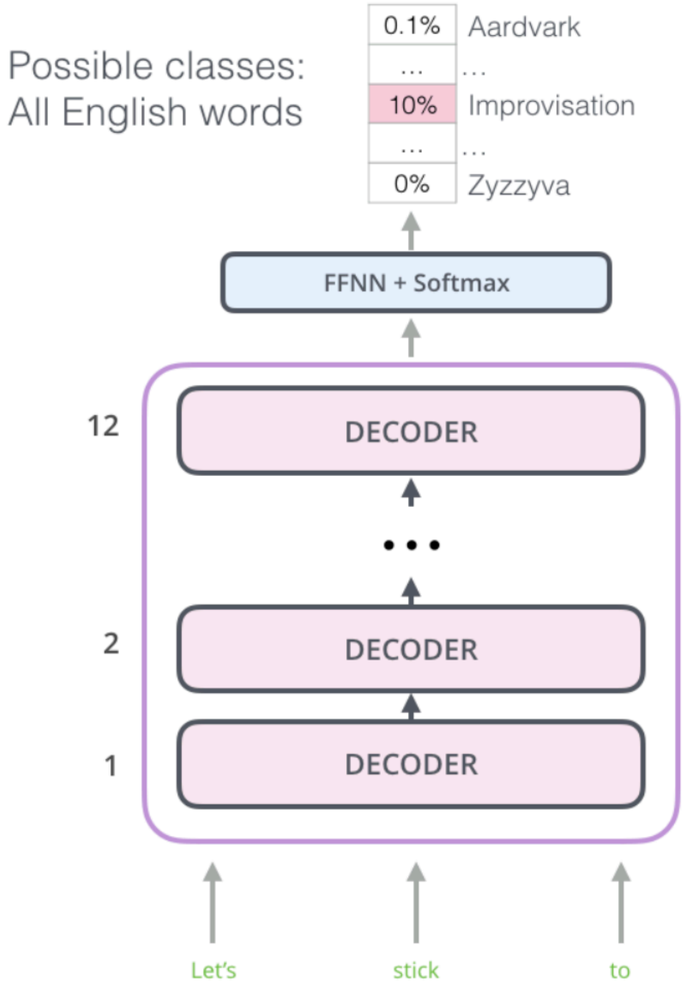
Pros:

- new SOTA at the time
- left and right context

Cons:

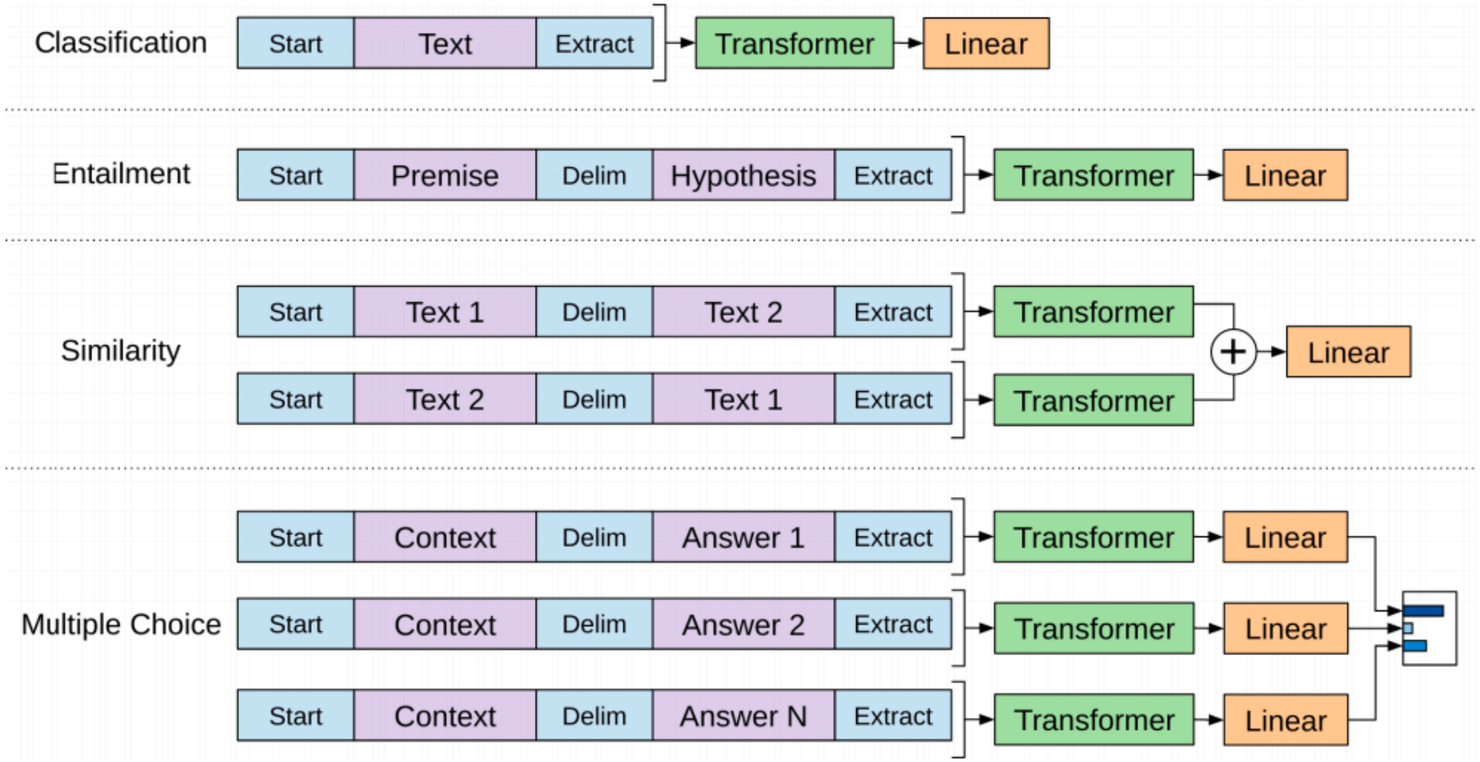
- RNN
- left and right context act almost independently
- mostly feature-based

GPT: pre-training



- Transformer decoder
- Train standard left-to-right LM

GPT: fine-tuning



Extra parameters:

- Last linear layer
- embeddings for delimiter tokens

GPT

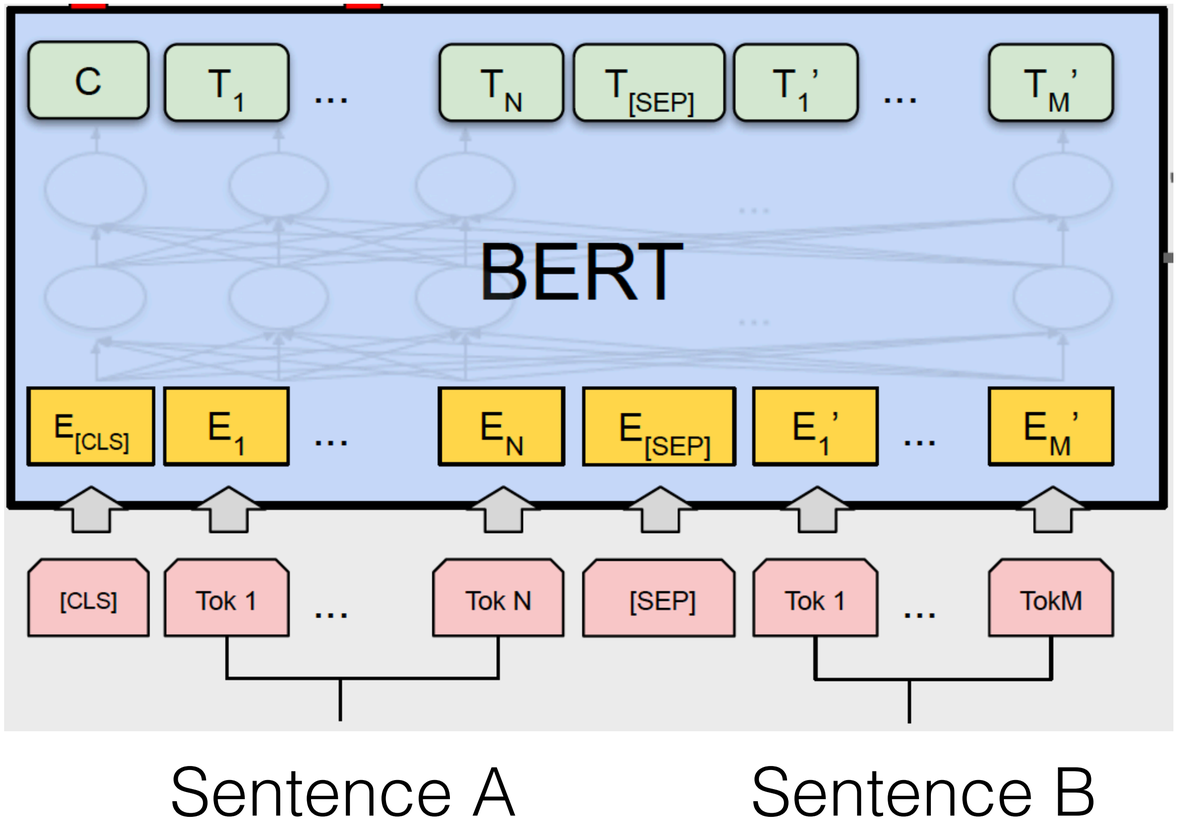
Pros:

- Transformer
- fine-tuning -> more like transfer learning
- strong LM

Cons:

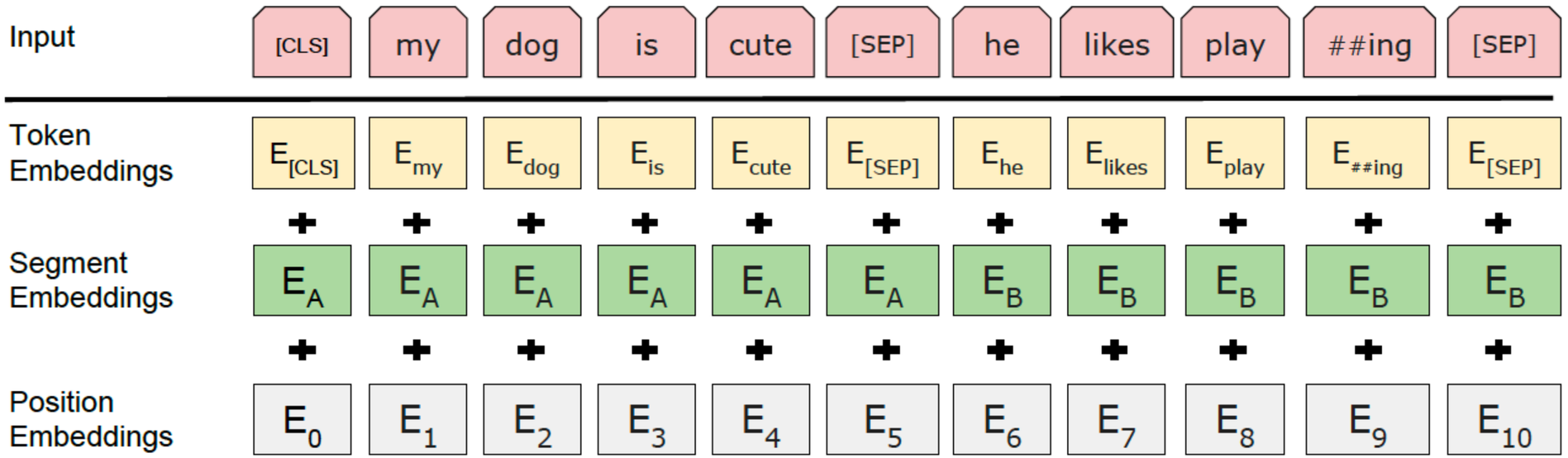
- only left context

BERT: architecture



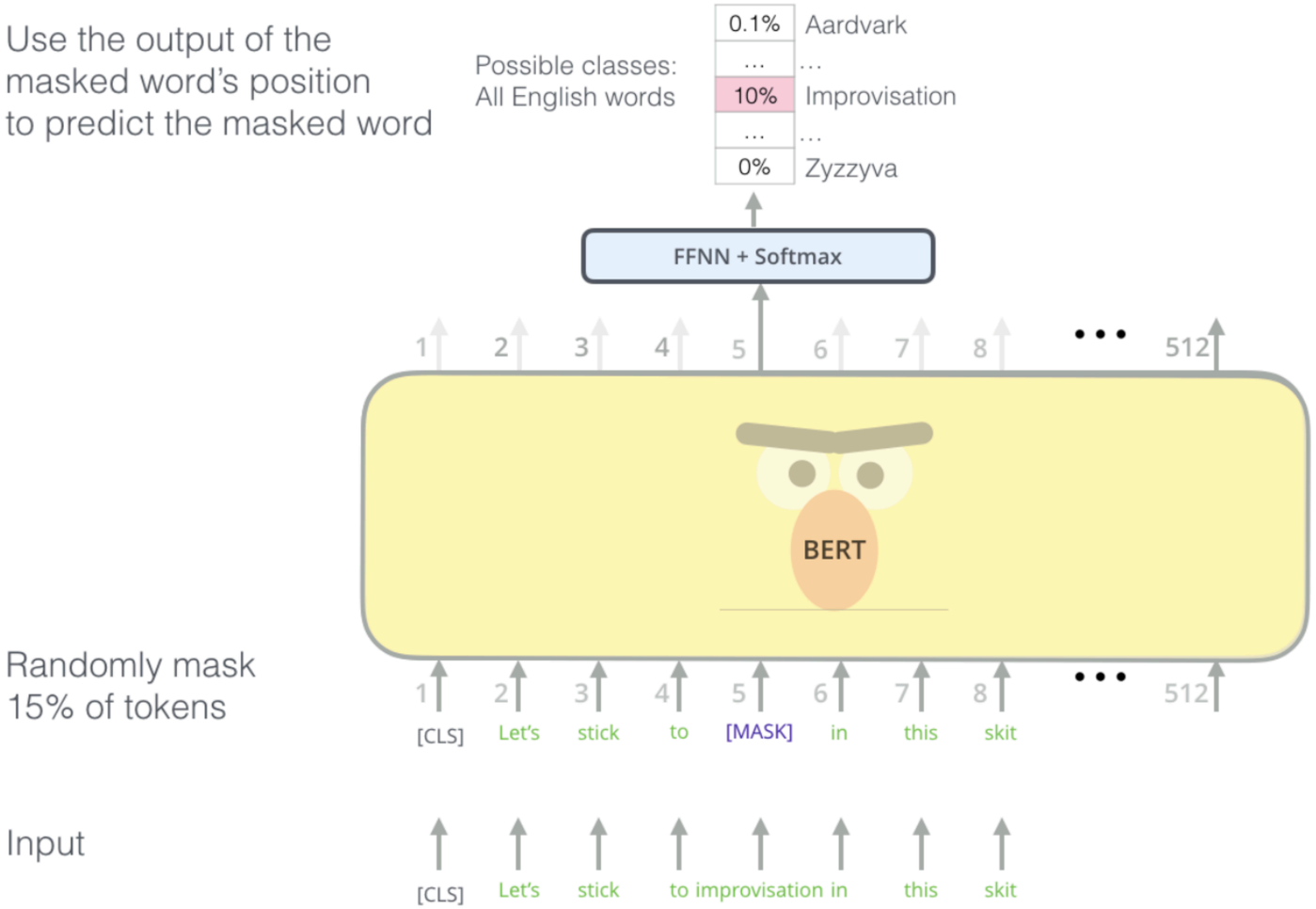
- Transformer encoder
- Two sentences in the input
- CLS token
- SEP token

BERT: input representation



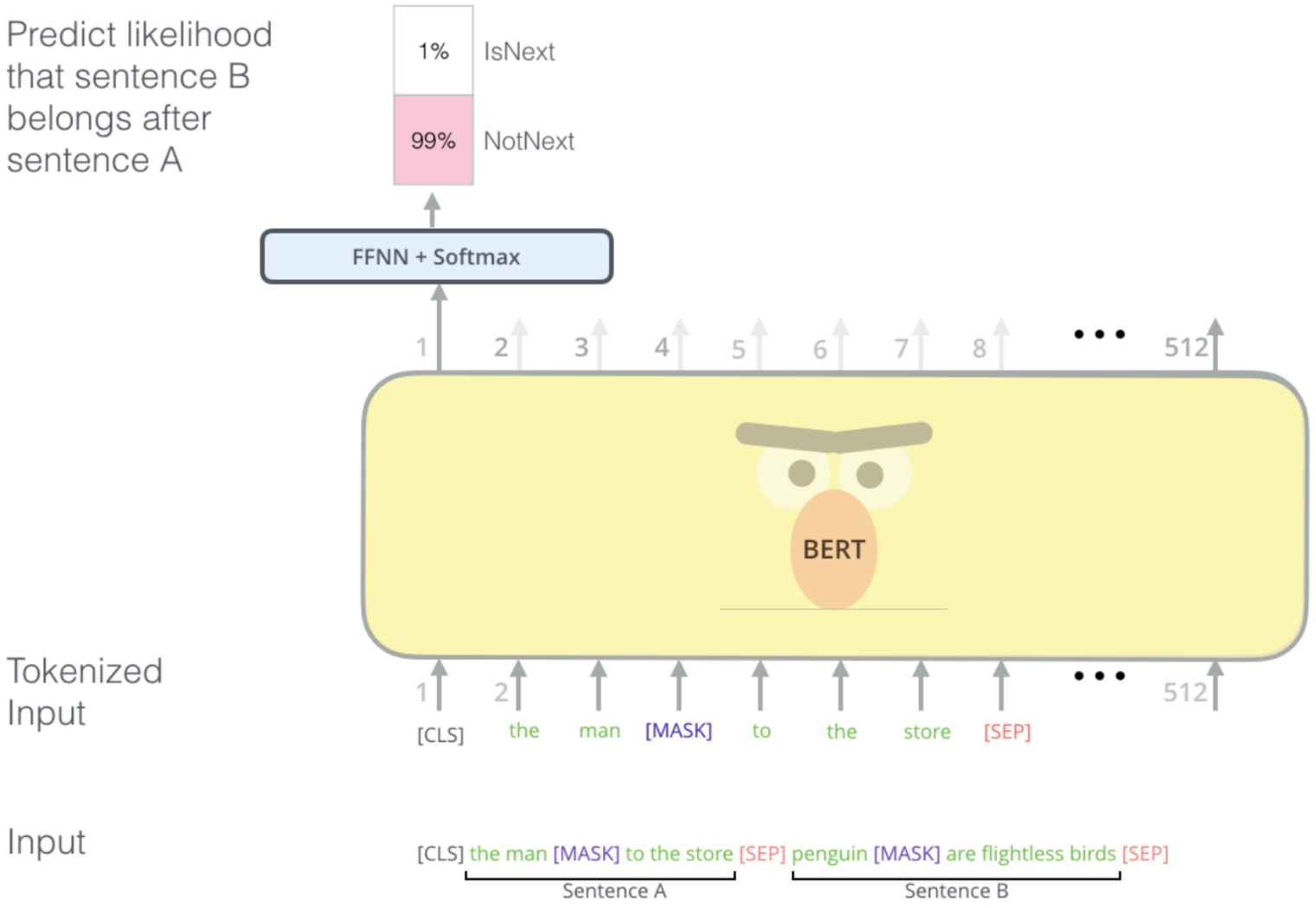
All three types of embeddings are learned

BERT: pre-training - Masked LM



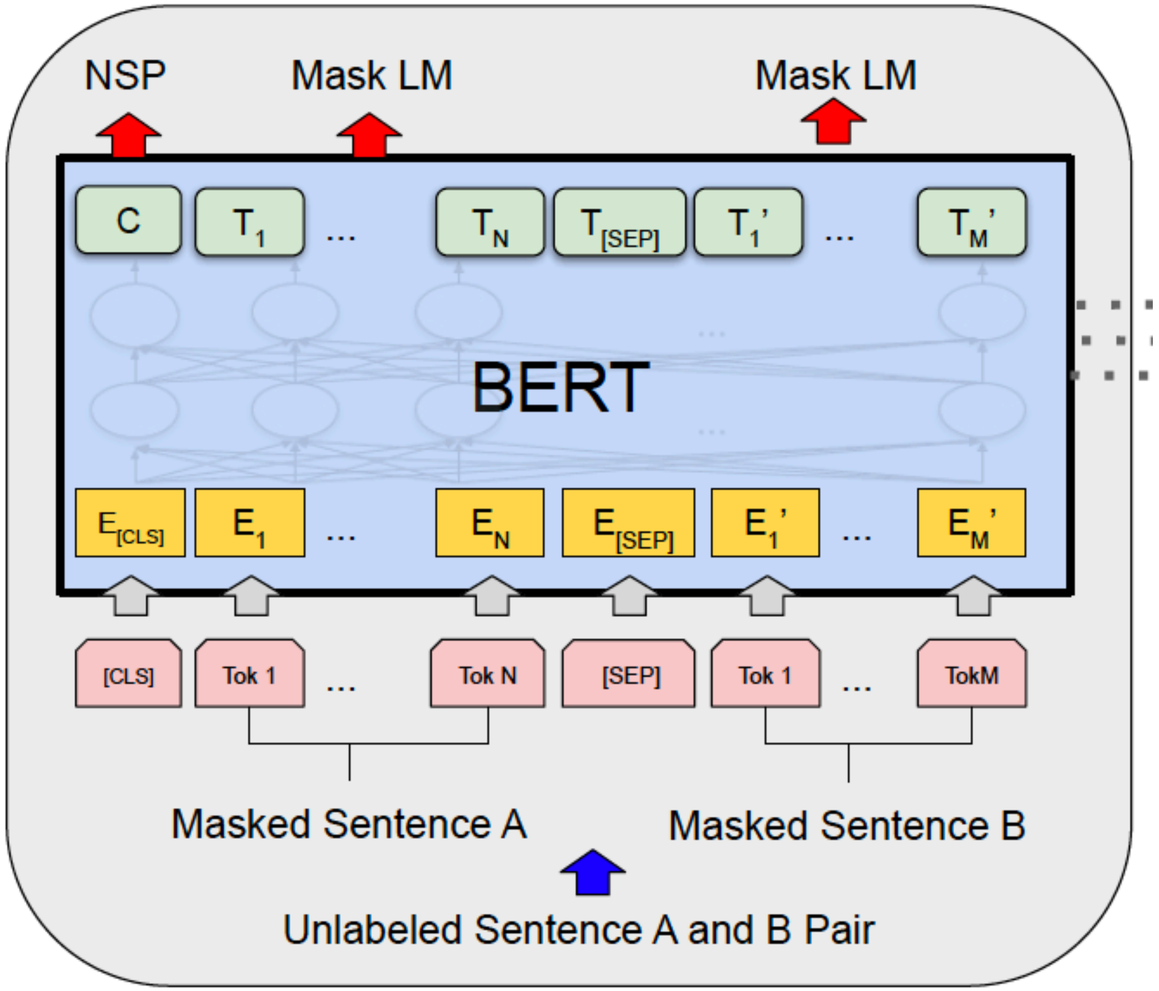
- No information leakage
- Predict only masked tokens
- Mismatch between pre-training and fine-tuning -> for each token from these 15%:
  - 80% - [MASK]
  - 10% - random token
  - 10% - original token

BERT: pre-training - Next Sentence Prediction



- 50% - IsNext
- 50% - NotNext
- Use CLS output
- Beneficial for QA and NLI because it helps to understand the relationship between two sentences

BERT: pre-training



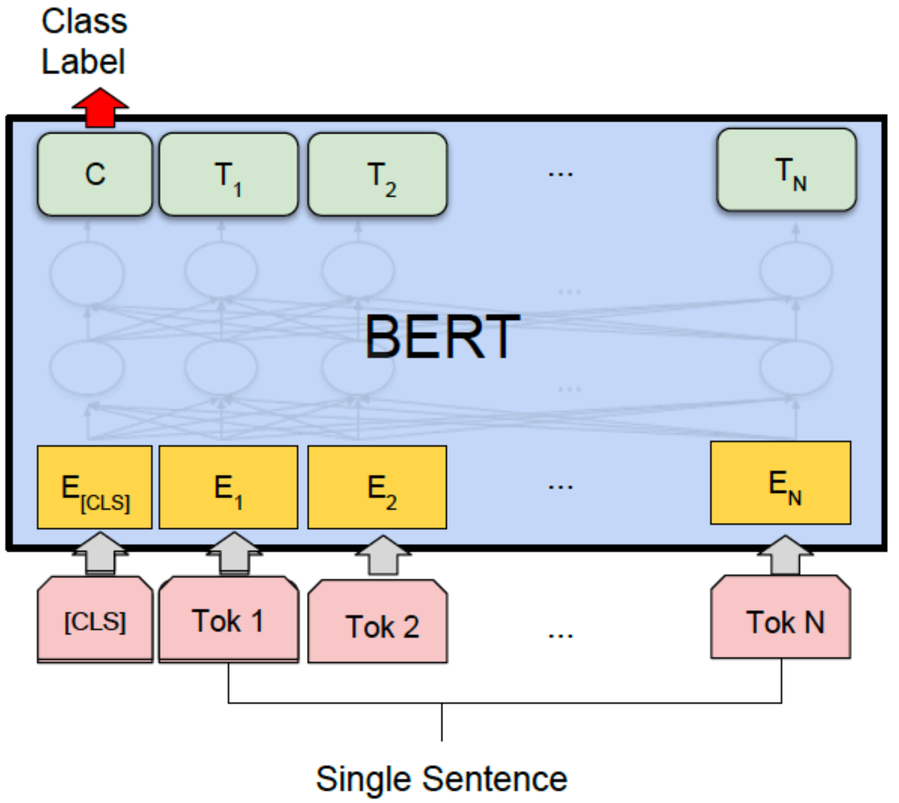
Models:

- BERT base: L=12, H=768, A=12, Total Parameters=110M
- BERT large: L=24, H=1024, A=16, Total Parameters=340M
- sequence length: 128 for 90% of the time and 512 for the last 10% of the time

Data:

- BooksCorpus (800M words)
- English Wikipedia (2,500M words) - only the text passages

BERT: fine-tuning

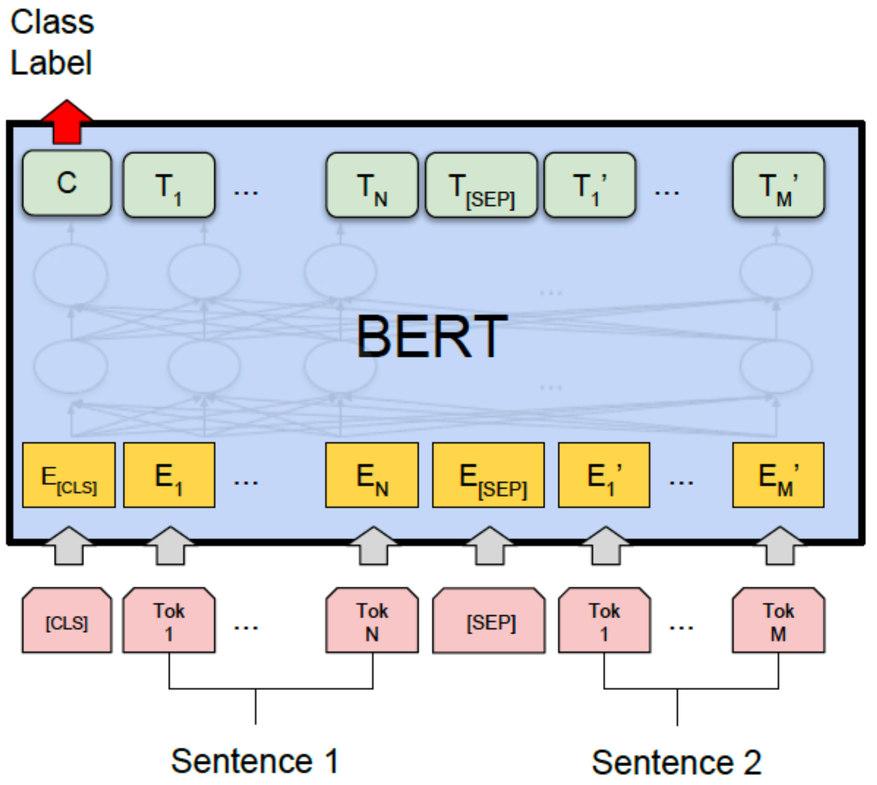


(b) Single Sentence Classification Tasks: SST-2, CoLA

SST-2 The Stanford Sentiment Treebank is a binary single-sentence classification task consisting of sentences extracted from movie reviews with human annotations of their sentiment (Socher et al., 2013).

CoLA The Corpus of Linguistic Acceptability is a binary single-sentence classification task, where the goal is to predict whether an English sentence is linguistically "acceptable" or not (Warstadt et al., 2018).

BERT: fine-tuning



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

MNLI Multi-Genre Natural Language Inference is a large-scale, crowdsourced entailment classification task (Williams et al., 2018). Given a pair of sentences, the goal is to predict whether the second sentence is an *entailment*, *contradiction*, or *neutral* with respect to the first one.

QQP Quora Question Pairs is a binary classification task where the goal is to determine if two questions asked on Quora are semantically equivalent (Chen et al., 2018).

BERT: results on GLUE

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**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.

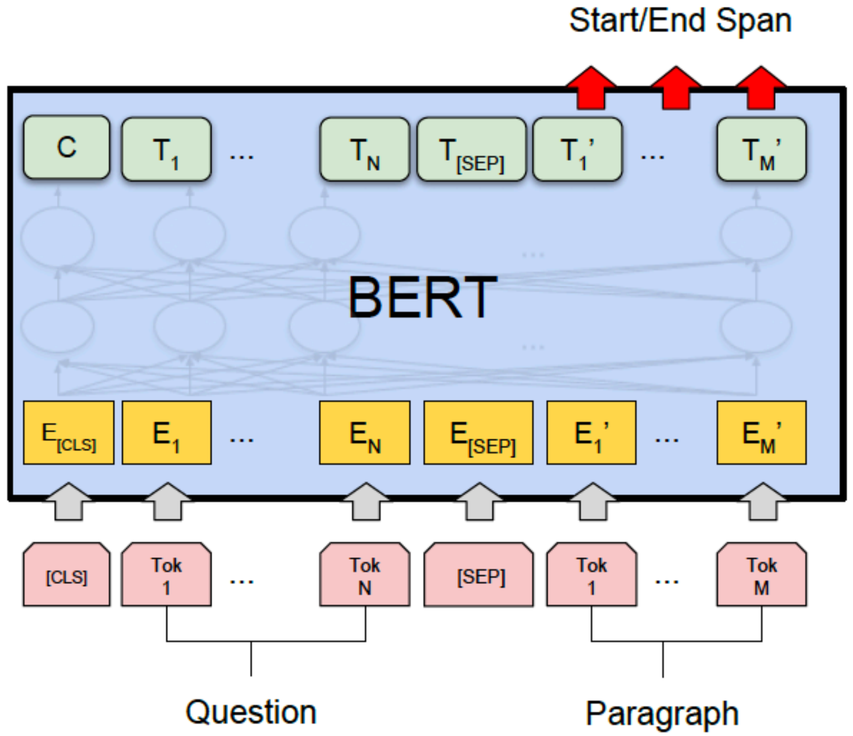
BERT: results on SWAG

Given a sentence, the task is to choose the most plausible continuation among four choices.

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

Table 4: SWAG Dev and Test accuracies. †Human performance is measured with 100 samples, as reported in the SWAG paper.

BERT: fine-tuning



(c) Question Answering Tasks: SQuAD v1.1

SQuAD v1.1

Given a question and a passage from Wikipedia containing the answer, the task is to predict the answer text span in the passage.

BERT: results on SQuAD v1.1 and v2.0

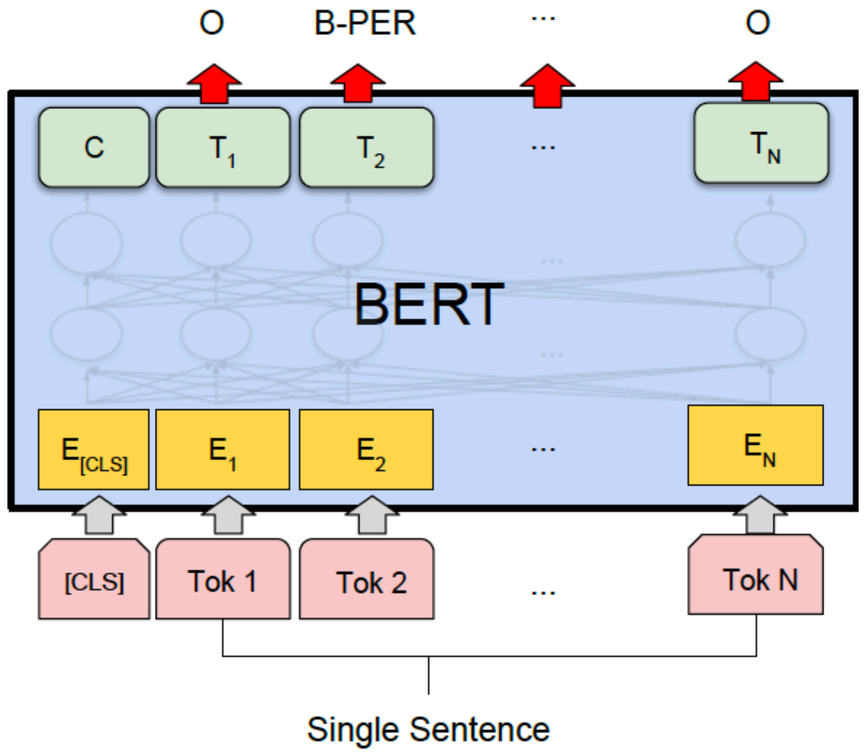
System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	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.

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
$\text{BERT}_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

Table 3: SQuAD 2.0 results. We exclude entries that use BERT as one of their components.

Feature-based Approach with BERT



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

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

Table 7: CoNLL-2003 Named Entity Recognition results. Hyperparameters were selected using the Dev set. The reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

BERT: training

Pre-training

BERT base - 4 Cloud TPUs in Pod configuration (16 TPU chips total)
BERT large -16 Cloud TPUs (64 TPU chips total)

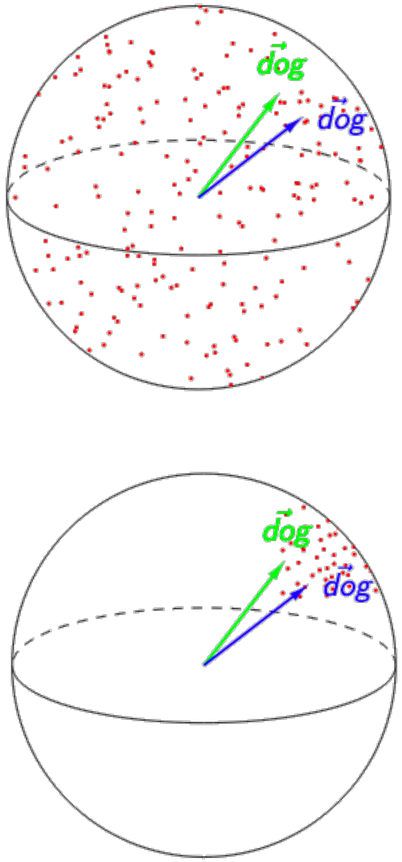
Each pretraining took 4 days to complete.

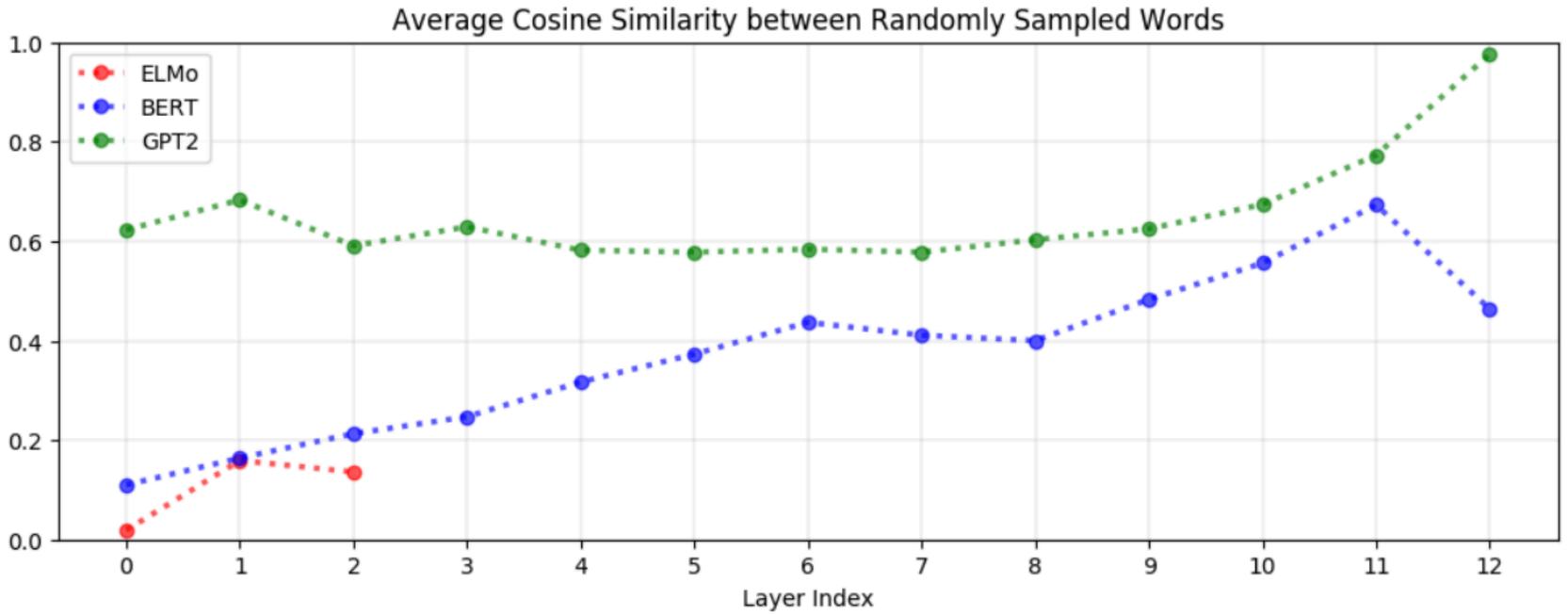
Fine-tuning

at most 1 hour on a single Cloud TPU, or a few hours on a GPU

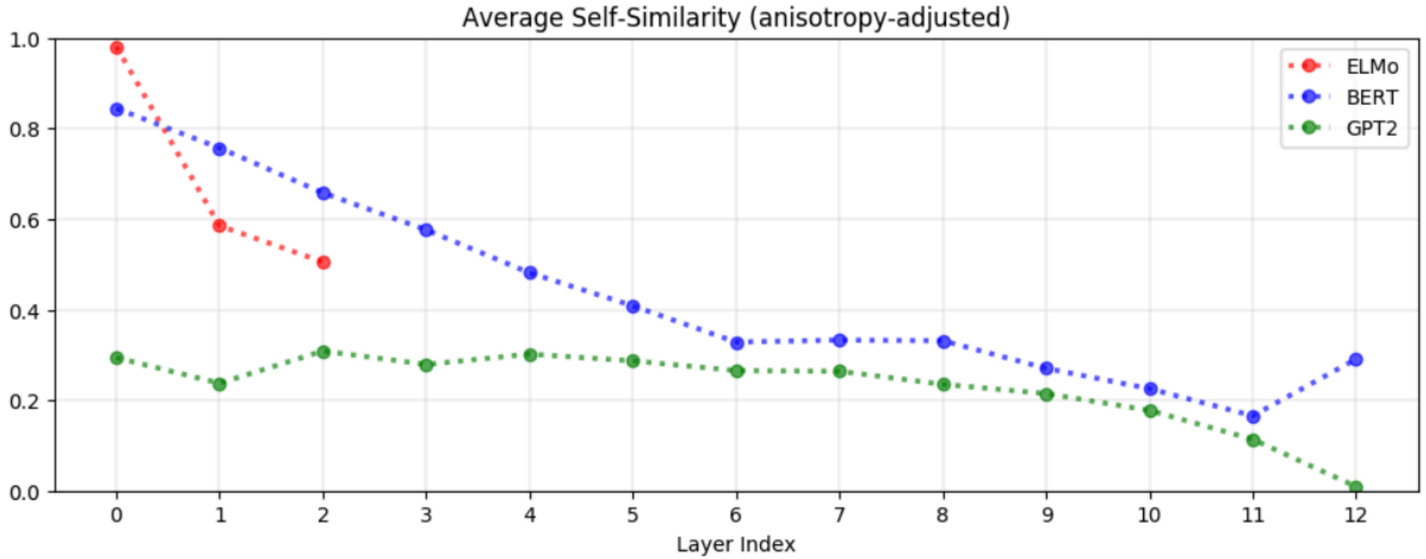
How Contextual are Contextualized Word Representations?

Anisotropy of contextualized embeddings





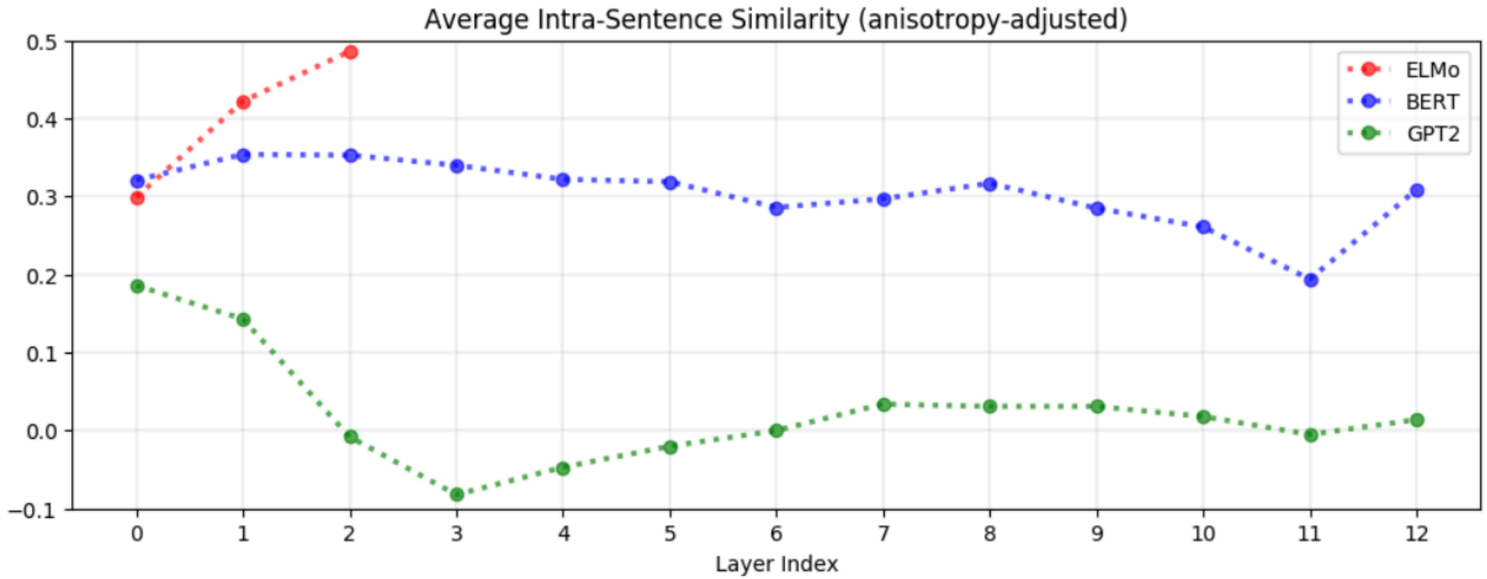
Context-Specificity



$$SelfSim_\ell(w) = \frac{1}{n^2 - n} \sum_j \sum_{k \neq j} \cos(f_\ell(s_j, i_j), f_\ell(s_k, i_k))$$

- Contextualized word representations are more context-specific in higher layers
- Stopwords (e.g., ‘the’, ‘of’, ‘to’) have among the most context-specific representation
- On average, less than 5% of the variance in a word’s contextualized representations can be explained by a static embedding

Context-Specificity



$$IntraSim_\ell(s) = \frac{1}{n}\sum_i \cos(\vec{s_\ell}, f_\ell(s,i))$$

$$\text{where } \vec{s_\ell} = \frac{1}{n}\sum_i f_\ell(s,i)$$

ELMO -> BERT -> GPT

Representation of words in the same sentence are similar -> two words in the same sentence do not necessarily have a similar meaning simply because they share the same context

Word Sense Disambiguation

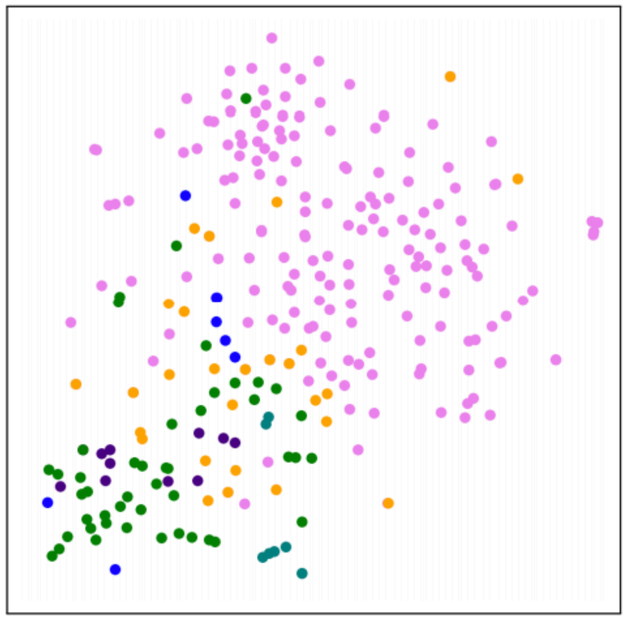
Model	SE-2	SE-3	S7-T7 (coarse)		S7-T17 (fine)	
			SemCor	WNGT	SemCor	WNGT
Flair	65.27	68.75	69.24	78.68	45.92	50.99
ELMo	67.57	70.70	70.80	79.12	52.61	50.11
BERT	**76.10**	**78.62**	73.61	**81.11**	**59.82**	55.16

Table 2: kNN with $k = 1$ WSD performance (F1%). Best results for each testset are marked bold.

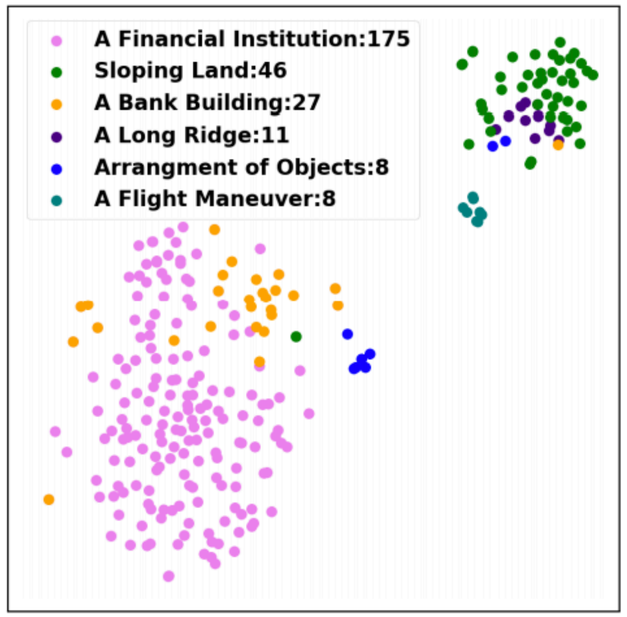
ELMO: linear combination of embedding, LSTM1 and LSTM2

BERT: concatenation of the averaged wordpiece vectors of the last four layers

T-SNE plots of different senses of 'bank'

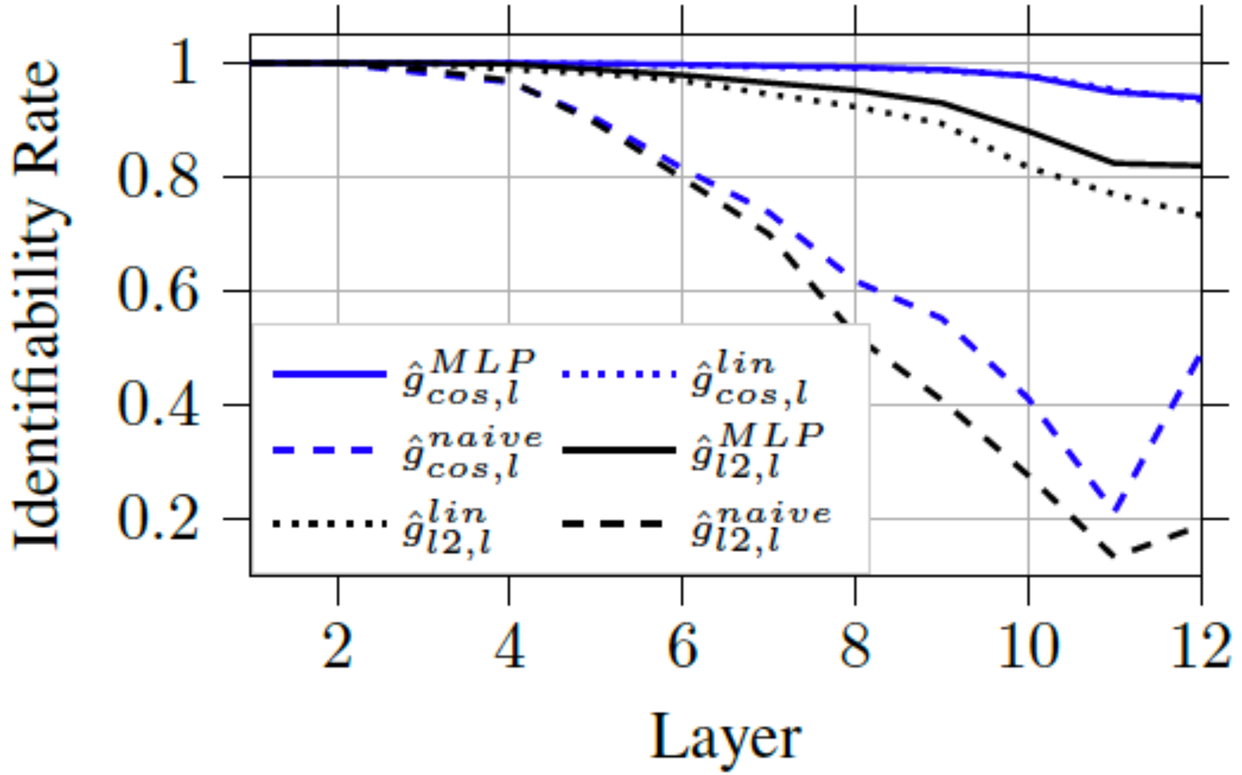


(c) ELMo



(a) BERT

Token identifiability



Token mixing

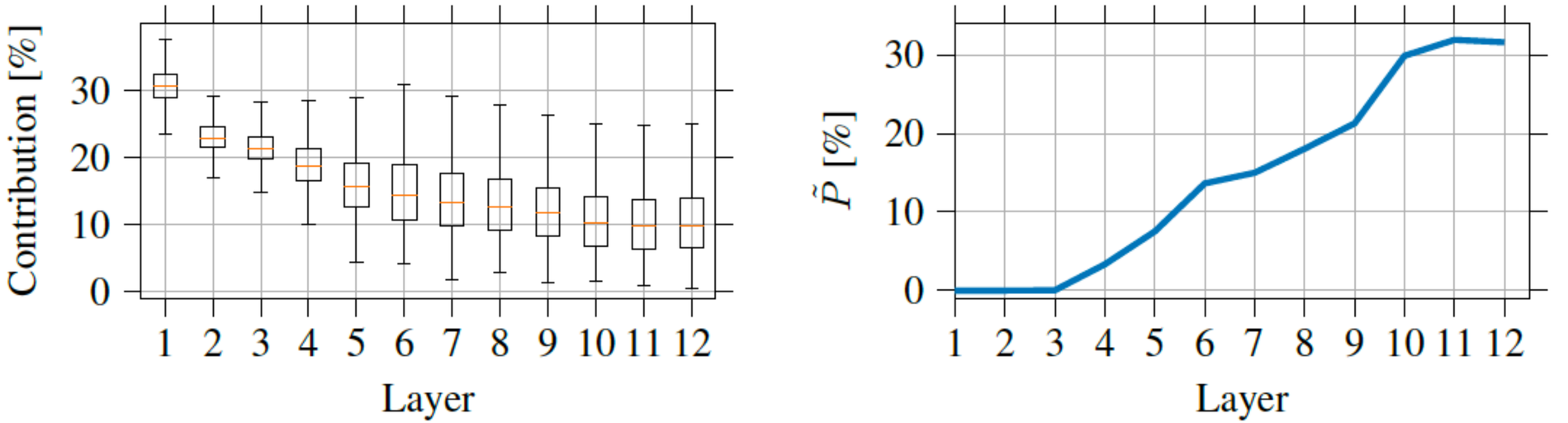


Figure 3: (a) Contribution of the input token to the embedding at the same position. The orange line represents the median value and outliers are not shown. (b) Percentage of tokens $\tilde{P}$ that are *not* the main contributors to their corresponding contextual embedding at each layer.

$$c_{i,j}^{l} = \frac{||\nabla_{i,j}^{l}||_2}{\sum_{k=0}^{d_s}||\nabla_{k,j}^{l}||_2} \qquad \text{with} \quad \nabla_{i,j}^{l} = \frac{\delta \boldsymbol{e}_j^{l}}{\delta \boldsymbol{x}_i}$$

Context contribution

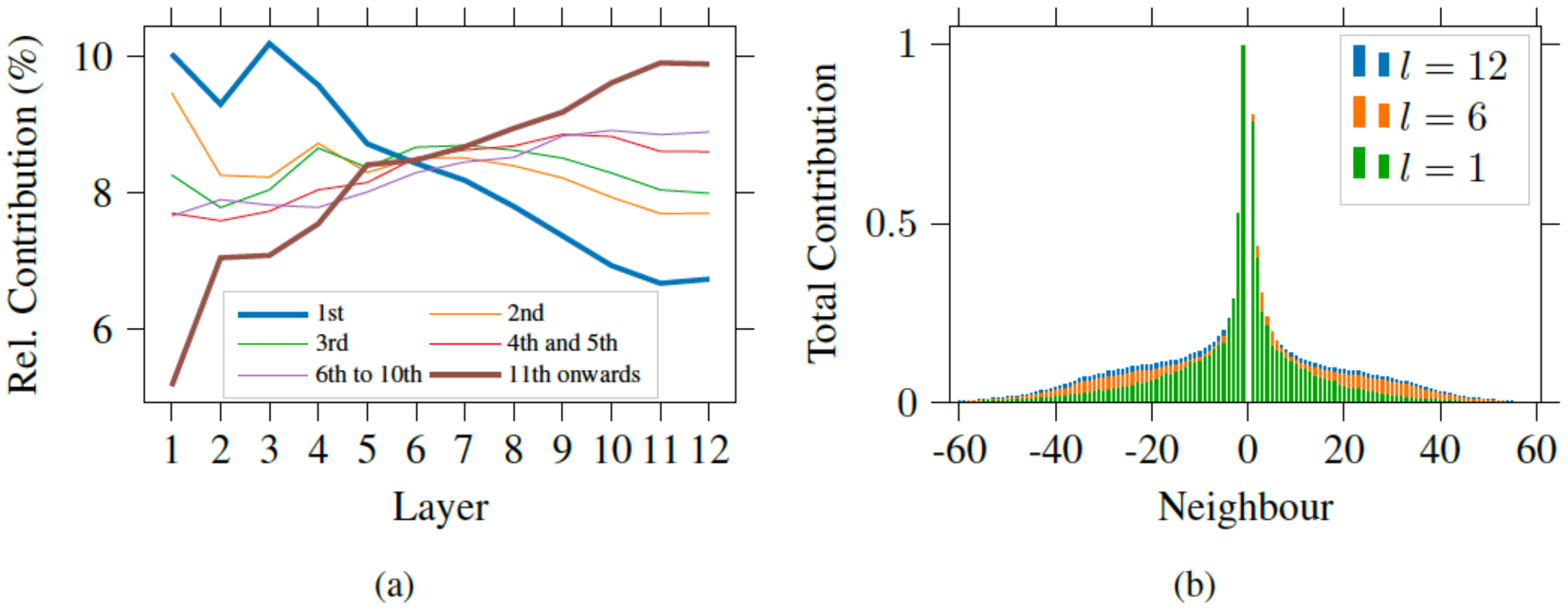
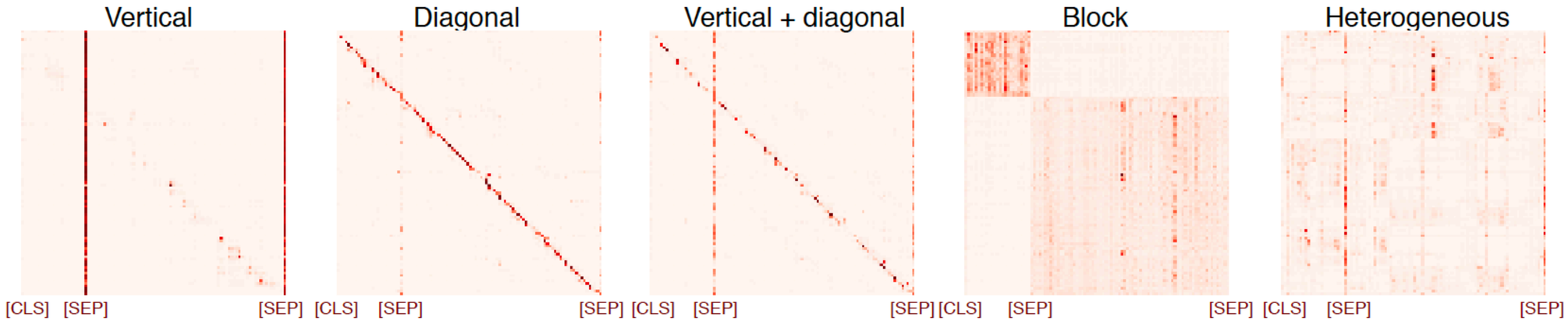


Figure 4: (a) Relative contribution per layer of neighbours at different positions. (b) Total contribution per neighbour for the first, middle and last layers.

BERT’s self-attention patterns

BERT's self-attention patterns



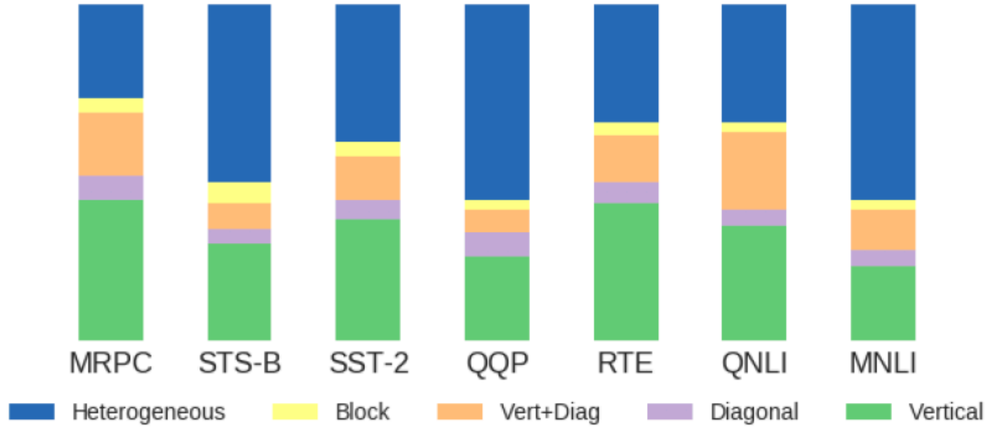
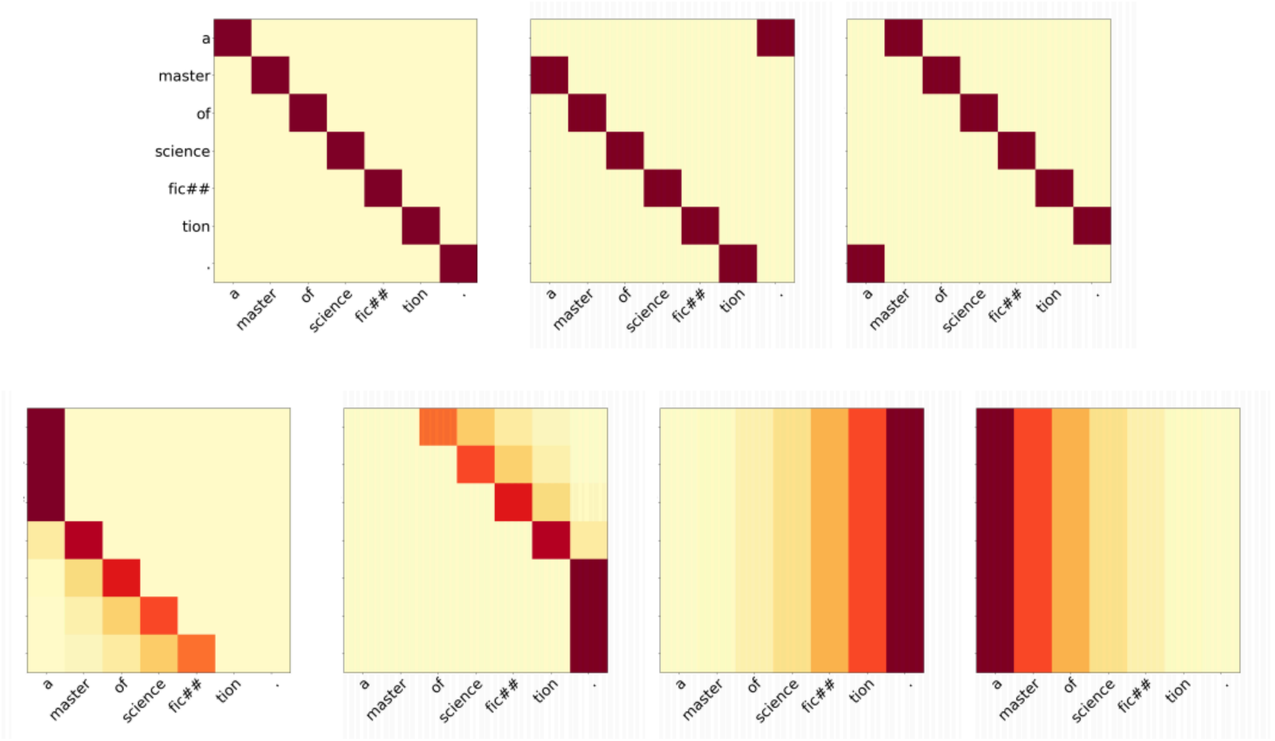


Figure 2: Estimated percentages of the identified self-attention classes for each of the selected GLUE tasks.

A lot of self-attention heads do not directly encode any nontrivial linguistic information

Fixed Encoder Self-Attention in Transformer



+ 1 trainable head

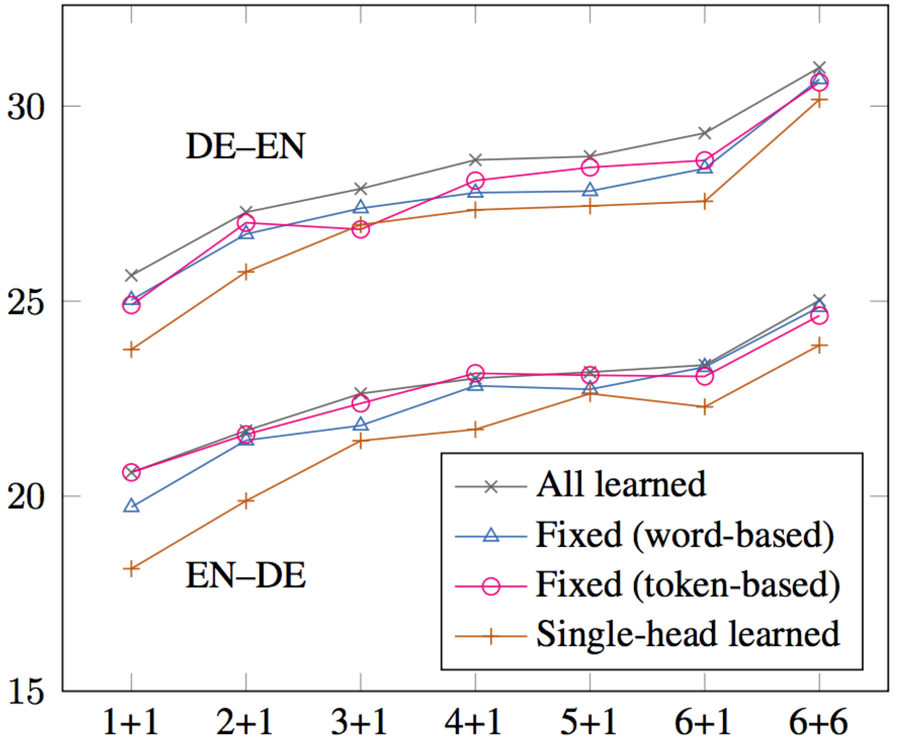
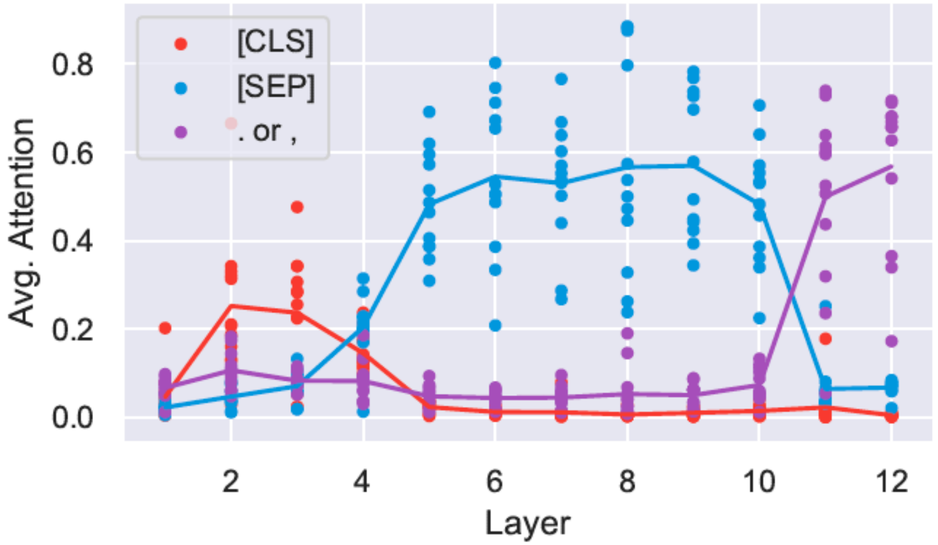


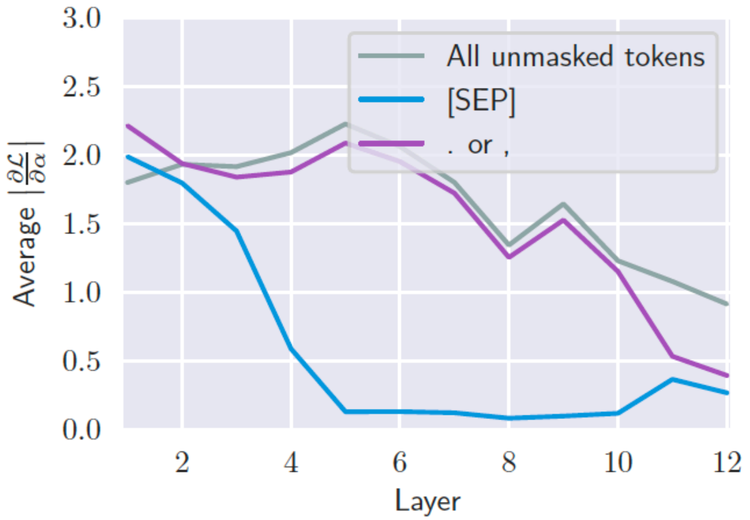
Figure 2: BLEU scores for the German ↔ English standard scenario. The x-axis shows different configurations of encoder and decoder layers.

BERT's self-attention patterns

A lot of attention to CLS, SEP, and punctuation tokens

- CLS - summary
- SEP - no-op (look at SEP if this head relation is not applicable)
- punctuation - similar to SEP???





Attention identifiability

Output of one head from MHA: $\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})\boldsymbol{H} = \boldsymbol{A}\boldsymbol{E}\boldsymbol{W}^V\boldsymbol{H} = \boldsymbol{A}\boldsymbol{T}$

$$\begin{aligned}\text{rank}\,(\boldsymbol{T}) &\leq \min\left(\text{rank}(\boldsymbol{E}), \text{rank}(\boldsymbol{W}^V), \text{rank}(\boldsymbol{H})\right) \\ &\leq \min(d_s, d, d, d_v, d_v, d) \\ &= \min\left(d_s, d_v\right).\end{aligned}$$

BERT:
d_s = 512
d_v = 64

Null space of T:

$$\dim(\text{LN}(\boldsymbol{T})) = d_s - \text{rank}\,(\boldsymbol{T}) \geq d_s - \min\left(d_s, d_v\right) = \begin{cases} d_s - d_v, & \text{if } d_s > d_v \\ 0, & \text{otherwise} \end{cases}$$

+ they take into account probability constraints

Effective attention: $\boldsymbol{A}^{\perp} = \boldsymbol{A} - \text{Projection}_{\text{LN}(T)}\boldsymbol{A}.$

Attention identifiability

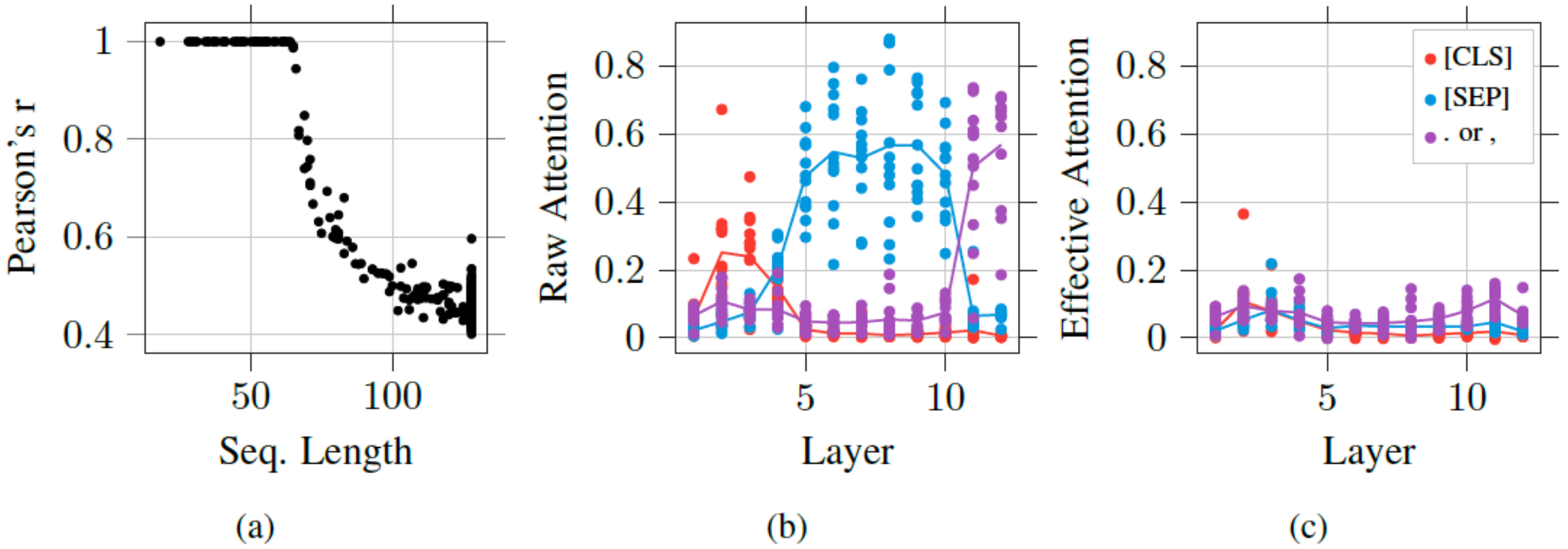
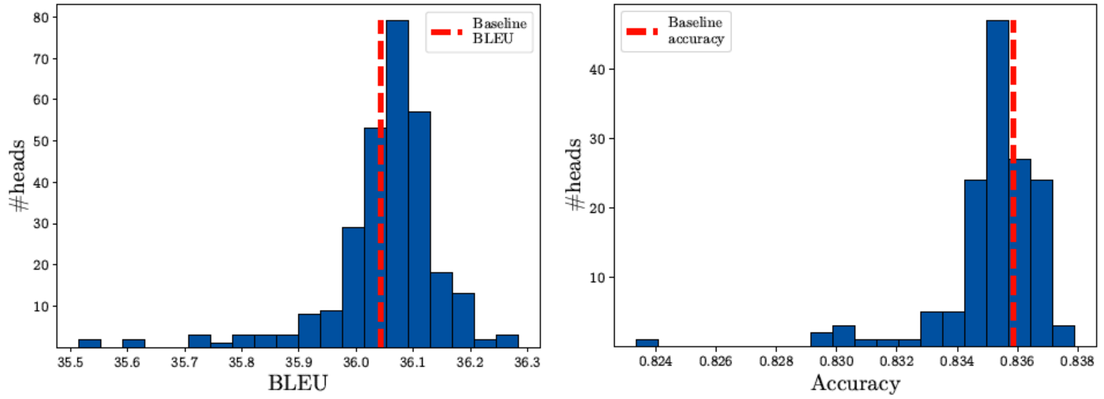


Figure 1: (a) Each point represents the Pearson correlation coefficient of effective attention and raw attention as a function of token length. (b) Raw attention vs. (c) effective attention, where each point represents the average (effective) attention of a given head to a token type.

Disabling self-attention heads

$$\text{MHAtt}(\mathbf{x}, q) = \sum_{h=1}^{N_h} \xi_h \text{Att}_{W_k^h, W_q^h, W_v^h, W_o^h}(\mathbf{x}, q)$$

Disable one head:



(a) WMT (b) BERT

Figure 1: Distribution of heads by model score after masking.

Disable all heads but one in a layer:

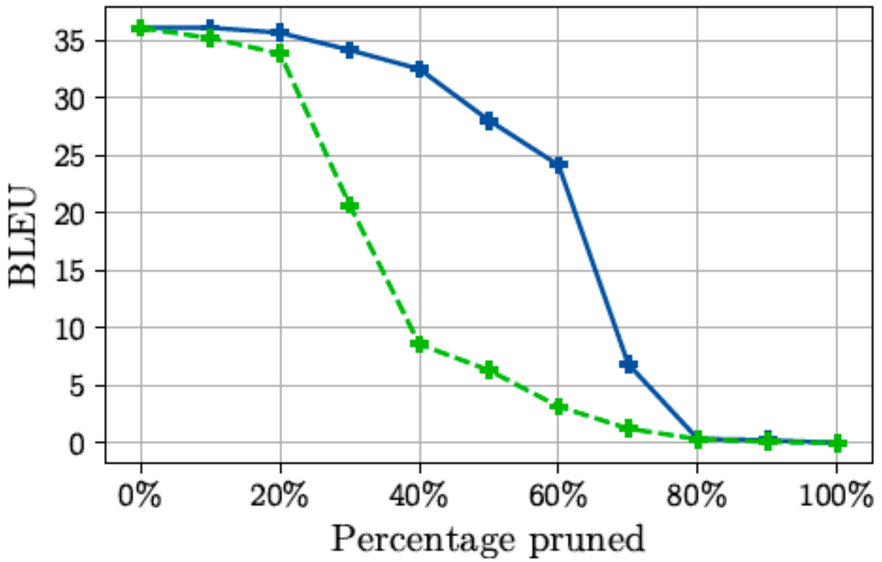
Layer	Enc-Enc	Enc-Dec	Dec-Dec
1	**-1.31**	**0.24**	-0.03
2	-0.16	0.06	0.12
3	0.12	0.05	0.18
4	-0.15	-0.24	0.17
5	0.02	**-1.55**	-0.04
6	**-0.36**	**-13.56**	0.24

Table 2: Best delta BLEU by layer when only one head is kept in the WMT model. Underlined numbers indicate that the change is statistically significant with $p < 0.01$.

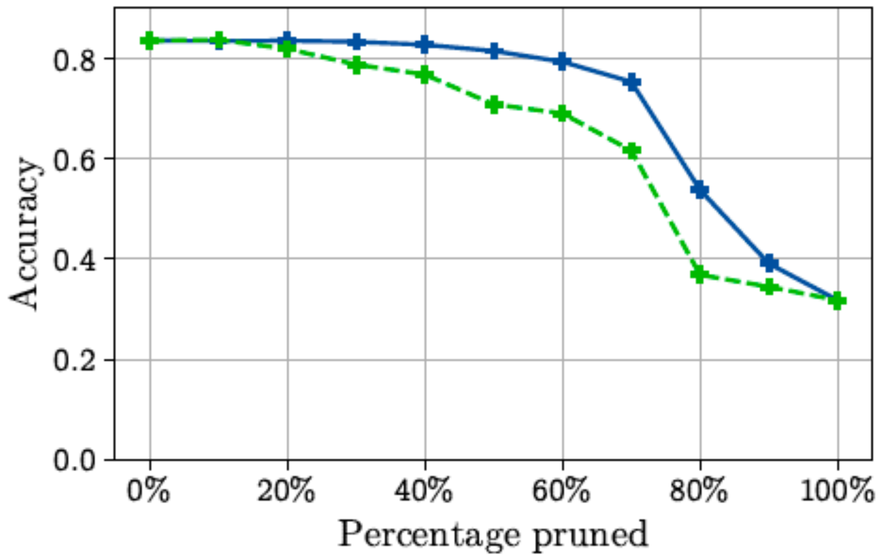
Layer		Layer	
1	-0.01%	7	0.05%
2	0.10%	8	-0.72%
3	-0.14%	9	-0.96%
4	-0.53%	10	0.07%
5	-0.29%	11	-0.19%
6	-0.52%	12	-0.12%

Table 3: Best delta accuracy by layer when only one head is kept in the BERT model. None of these results are statistically significant with $p < 0.01$.

Iterative Pruning of Attention Heads



(a) Evolution of BLEU score on `newstest2013` when heads are pruned from WMT.



(b) Evolution of accuracy on the MultiNLI-matched validation set when heads are pruned from BERT.

Figure 3: Evolution of accuracy by number of heads pruned according to I_h (solid blue) and individual oracle performance difference (dashed green).

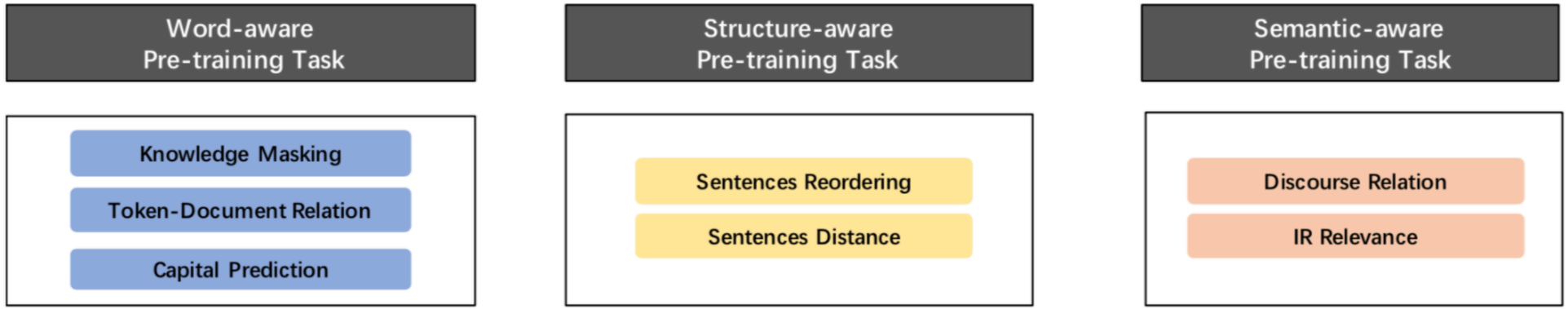
What else?

- Syntactic knowledge: BERT representations contain some information about syntactic tree structure and encode information about parts of speech, syntactic chunks and roles (this does not indicate that it actually relies on that knowledge)
- Semantic knowledge: BERT encodes information about entity types, relations, semantic roles, and proto-roles (e.g. BERT prefer “to tip a chef” to“to tip a robin” when the true sentence is “to tip a waiter”)
- …

BERTs friends

What to change?

- Longer training
- Better optimization
- More data:
  - just more data for MLM in XLNet, RoBERTa
  - sequential pre-training on a large number of tasks in ERNIE 2.0

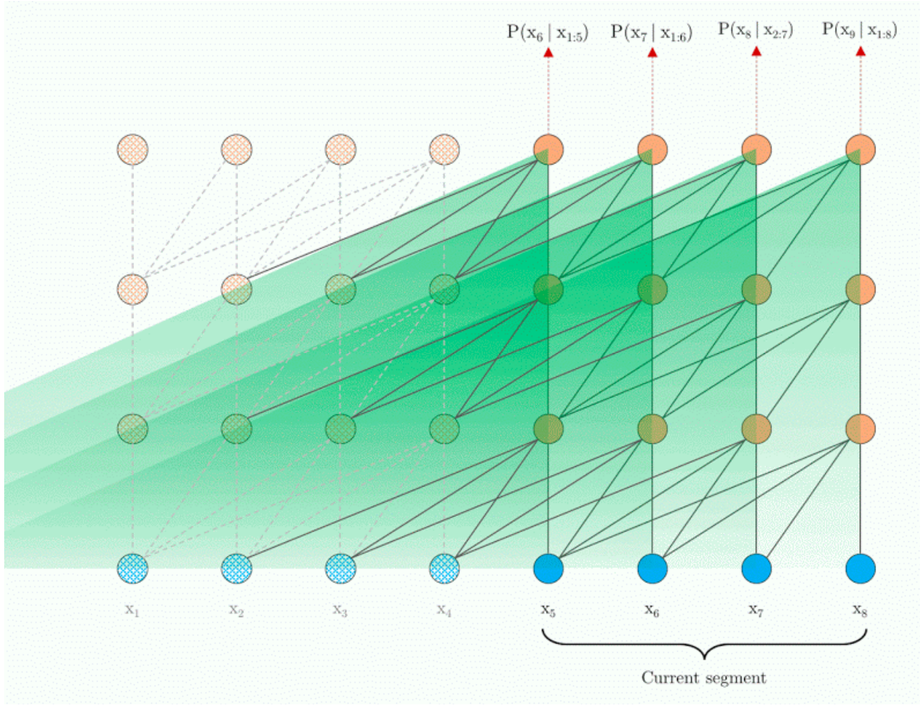


https://github.com/PaddlePaddle/ERNIE

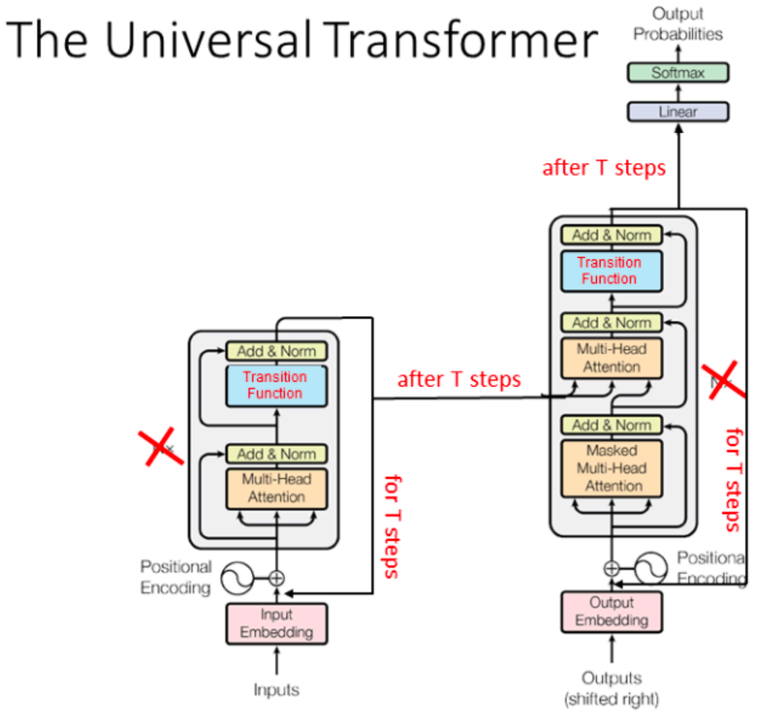
What to change?

Transformer type:

- Transformer-XL in XLNet



- Universal Transformer in ALBERT



https://ai.googleblog.com/2019/01/transformer-xl-unleashing-potential-of.html

https://glassboxmedicine.com/2019/09/07/universal-transformers/

What to change?

NSP objective:

- None in RoBERTa, SpanBERT
- Sentence-order prediction (SOP) in ALBERT
- Predict both previous and next sentences in StructBERT

What to change?

MLM objective:

- Dynamic masking in RoBERTa
- Sentence-order prediction (SOP) in ALBERT
- Permutation language modeling in XLNet
- Span boundary objective in SpanBERT
- Phrase masking and named entity masking in ERNIE
- + Discriminator for replaced token detection in ELECTRA

What to change?

Pre-training procedure:

- Recursive pertaining

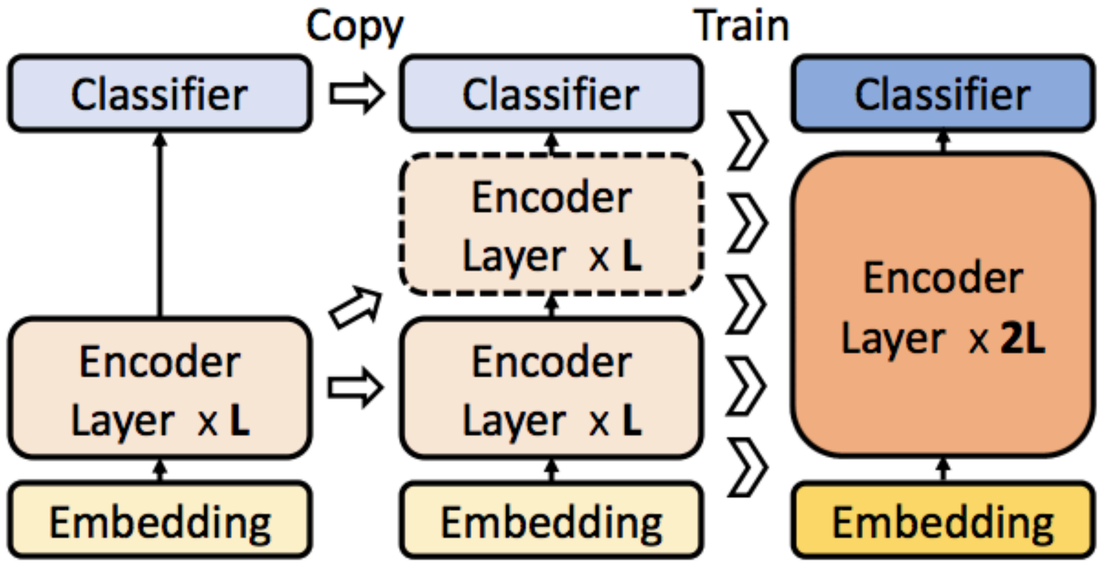


Figure 3. The diagram of the *stacking* algorithm.

BERT compression

		Compression	Performance	Speedup	Model	Evaluation
Distillation	DistilBERT (Sanh et al., 2019)	×2.5	90%	×1.6	$BERT_6$	All GLUE tasks
	$BERT_6$-PKD (Sun et al., 2019a)	×1.6	97%	×1.9	$BERT_6$	No WNLI, CoLA and STS-B
	$BERT_3$-PKD (Sun et al., 2019a)	×2.4	92%	×3.7	$BERT_3$	No WNLI, CoLA and STS-B
	(Aguilar et al., 2019)	×2	94%	-	$BERT_6$	CoLA, MRPC, QQP, RTE
	BERT-48 (Zhao et al., 2019)	×62	87%	×77	$BERT_{12}^{*\dagger}$	MNLI, MRPC, SST-2
	BERT-192 (Zhao et al., 2019)	×5.7	94%	×22	$BERT_{12}^{*\dagger}$	MNLI, MRPC, SST-2
	TinyBERT (Jiao et al., 2019)	×7.5	96%	×9.4	$BERT_4^{*\dagger}$	All GLUE tasks
	MobileBERT (Sun et al.)	×4.3	100%	×4	$BERT_{24}^{\dagger}$	No WNLI
	PD (Turc et al., 2019)	×1.6	98%	$\times 2.5^3$	$BERT_6^{\dagger}$	No WNLI, CoLA and STS-B
	MiniBERT(Tsai et al., 2019)	$\times 6^{\S}$	98%	$\times 27^{\S}$	$mBERT_3^{\dagger}$	CoNLL-2018 POS and morphology
	BiLSTM soft (Tang et al., 2019)	×110	91%	$\times 434^{\ddagger}$	$BiLSTM_1$	MNLI, QQP, SST-2
Quant.	Q-BERT (Shen et al., 2019)	×13	99%	-	$BERT_{12}$	MNLI, SST-2
	Q8BERT (Zafrir et al., 2019)	×4	99%	-	$BERT_{12}$	All GLUE tasks
Other	ALBERT-base (Lan et al., 2019)	×9	97%	×5.6	$BERT_{12}^{**}$	MNLI, SST-2
	ALBERT-xxlarge (Lan et al., 2019)	×0.47	107%	×0.3	$BERT_{12}^{**}$	MNLI, SST-2
	BERT-of-Theseus (Xu et al., 2020)	×1.6	98%	-	$BERT_6$	No WNLI

Table 1: Comparison of BERT compression studies. Compression, performance retention, and inference time speedup figures are given with respect to $BERT_{base}$, unless indicated otherwise. Performance retention is measured as a ratio of average scores achieved by a given model and by $BERT_{base}$. The subscript in the model description reflects the number of layers used. *Smaller vocabulary used. †The dimensionality of the hidden layers is reduced. **The dimensionality of the embedding layer is reduced. ‡Compared to $BERT_{large}$. §Compared to mBERT.

RoBERTa

A Robustly Optimized BERT Pretraining Approach

ICLR 2020

reject

RoBERTa: overview

- training the model longer: 31K(1M with small batches) -> 500K
- bigger batches: 256 sequences -> 8K
- more training data: 16GB -> 160GB
- larger BPE vocabulary: 30K -> 50K
- training on longer sequences: always 512

- removing the next sentence prediction objective
- dynamically changing the masking pattern applied to the training data

Result: longer training, the same size model, better performance

RoBERTa: dynamic masking

BERT:

- training data is duplicated 10 times
- static masking - generate one mask for each training instance before training

RoBERTa:

- dynamic masking - masks are generated at every epoch

Important for long training!

RoBERTa: no NSP objective

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss):				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss):				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
$\text{BERT}_{\text{BASE}}$	88.5/76.3	84.3	92.8	64.3
$\text{XLNet}_{\text{BASE}}$ (K = 7)	–/81.3	85.8	92.7	66.1
$\text{XLNet}_{\text{BASE}}$ (K = 6)	–/81.0	85.6	93.4	66.7

- removing the NSP loss matches or slightly improves performance
- BERT: removed the NSP while still retaining the SEGMENT-PAIR input format
- RoBERTa use FULL-SENTENCES to have consistent batch sizes

RoBERTa: GLUE

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$\text{BERT}_{\text{LARGE}}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$\text{XLNet}_{\text{LARGE}}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

Table 5: Results on GLUE. All results are based on a 24-layer architecture. $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ results are from Devlin et al. (2019) and Yang et al. (2019), respectively. RoBERTa results on the development set are a median over five runs. RoBERTa results on the test set are ensembles of *single-task* models. For RTE, STS and MRPC we finetune starting from the MNLI model instead of the baseline pretrained model. Averages are obtained from the GLUE leaderboard.

RoBERTa: SQuAD and RACE

Model	SQuAD 1.1 EM	SQuAD 1.1 F1	SQuAD 2.0 EM	SQuAD 2.0 F1
Single models on dev, w/o data augmentation				
$\text{BERT}_{\text{LARGE}}$	84.1	90.9	79.0	81.8
$\text{XLNet}_{\text{LARGE}}$	**89.0**	94.5	86.1	88.8
RoBERTa	88.9	**94.6**	**86.5**	**89.4**
Single models on test (as of July 25, 2019)				
$\text{XLNet}_{\text{LARGE}}$			86.3†	89.1†
RoBERTa			86.8	89.8
XLNet + SG-Net Verifier			**87.0**†	**89.9**†

Table 6: Results on SQuAD. † indicates results that depend on additional external training data. RoBERTa uses only the provided SQuAD data in both dev and test settings. $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ results are from Devlin et al. (2019) and Yang et al. (2019), respectively.

Model	Accuracy	Middle	High
Single models on test (as of July 25, 2019)			
$\text{BERT}_{\text{LARGE}}$	72.0	76.6	70.1
$\text{XLNet}_{\text{LARGE}}$	81.7	85.4	80.2
RoBERTa	**83.2**	**86.5**	**81.3**

Table 7: Results on the RACE test set. $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ results are from Yang et al. (2019).

ALBERT

A Lite BERT

ICLR 2020

ALBERT: overview

- factorized embedding parameterization:
  (V x H) to (V x E) and (E x H)
- cross-layer parameter sharing:
  share all parameters as in Universal Transformer or DEQ
- inter-sentence coherence loss:
  NSP is too easy as compared to MLM
  NSP -> sentence-order prediction (SOP) - IsNext or IsPrevious

Result: smaller/faser/worse performance or smaller/slower/better performance

ALBERT vs BERT

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

Table 2: Dev set results for models pretrained over BOOKCORPUS and Wikipedia for 125k steps. Here and everywhere else, the Avg column is computed by averaging the scores of the downstream tasks to its left (the two numbers of F1 and EM for each SQuAD are first averaged).

ALBERT: ablation study

- factorized embedding parameterization:

  hurts performance
- cross-layer parameter sharing:

  hurts performance
- inter-sentence coherence loss:

  SOP > NSP or None

ALBERT: SOTA (+ more data and dropout)

Models	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
BERT-large	86.6	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
XLNet-large	89.8	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa-large	90.2	94.7	**92.2**	86.6	96.4	**90.9**	68.0	92.4	-	-
ALBERT (1M)	90.4	95.2	92.0	88.1	96.8	90.2	68.7	92.7	-	-
ALBERT (1.5M)	**90.8**	**95.3**	**92.2**	**89.2**	**96.9**	**90.9**	**71.4**	**93.0**	-	-
Ensembles on test (from leaderboard as of Sept. 16, 2019)										
ALICE	88.2	95.7	**90.7**	83.5	95.2	92.6	**69.2**	91.1	80.8	87.0
MT-DNN	87.9	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2	98.6	90.3	86.3	96.8	93.0	67.8	91.6	90.4	88.4
RoBERTa	90.8	98.9	90.2	88.2	96.7	92.3	67.8	92.2	89.0	88.5
Adv-RoBERTa	91.1	98.8	90.3	88.7	96.8	93.1	68.0	92.4	89.0	88.8
ALBERT	**91.3**	**99.2**	90.5	**89.2**	**97.1**	**93.4**	69.1	**92.5**	**91.8**	**89.4**

Table 9: State-of-the-art results on the GLUE benchmark. For single-task single-model results, we report ALBERT at 1M steps (comparable to RoBERTa) and at 1.5M steps. The ALBERT ensemble uses models trained with 1M, 1.5M, and other numbers of steps.

ALBERT: SOTA (+ more data and dropout)

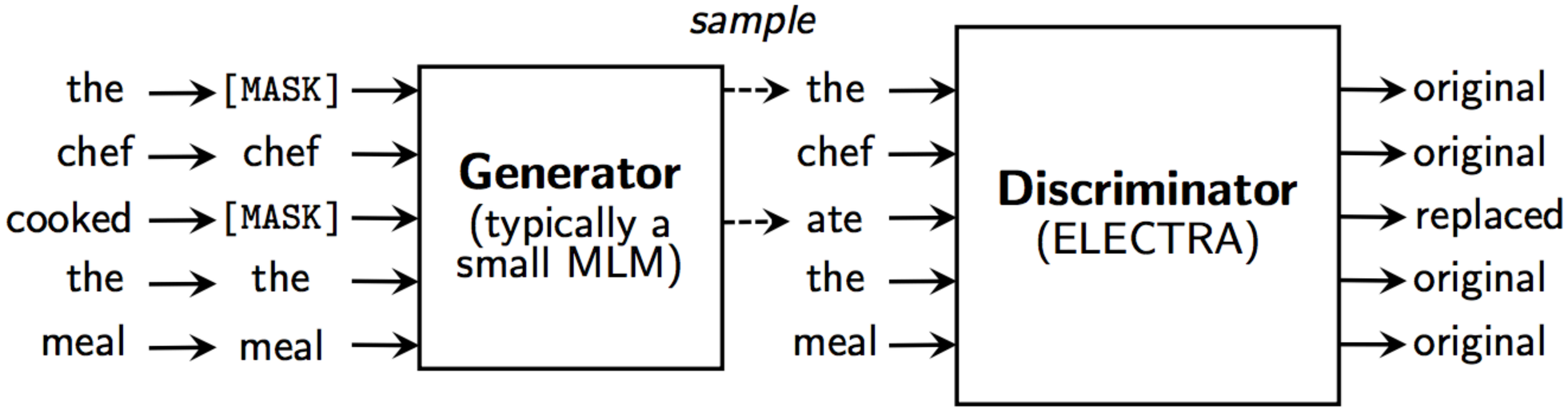
Models	SQuAD1.1 dev	SQuAD2.0 dev	SQuAD2.0 test	RACE test (Middle/High)
Single model (from leaderboard as of Sept. 23, 2019)				
BERT-large	90.9/84.1	81.8/79.0	89.1/86.3	72.0 (76.6/70.1)
XLNet	94.5/89.0	88.8/86.1	89.1/86.3	81.8 (85.5/80.2)
RoBERTa	94.6/88.9	89.4/86.5	89.8/86.8	83.2 (86.5/81.3)
UPM	-	-	89.9/87.2	-
XLNet + SG-Net Verifier++	-	-	90.1/87.2	-
ALBERT (1M)	94.8/89.2	89.9/87.2	-	86.0 (88.2/85.1)
ALBERT (1.5M)	**94.8/89.3**	**90.2/87.4**	**90.9/88.1**	**86.5 (89.0/85.5)**
Ensembles (from leaderboard as of Sept. 23, 2019)				
BERT-large	92.2/86.2	-	-	-
XLNet + SG-Net Verifier	-	-	90.7/88.2	-
UPM	-	-	90.7/88.2	
XLNet + DAAF + Verifier	-	-	90.9/88.6	-
DCMN+	-	-	-	84.1 (88.5/82.3)
ALBERT	**95.5/90.1**	**91.4/88.9**	**92.2/89.7**	**89.4 (91.2/88.6)**

ELECTRA

Efficiently Learning an Encoder that Classifies Token Replacements Accurately

ICLR 2020

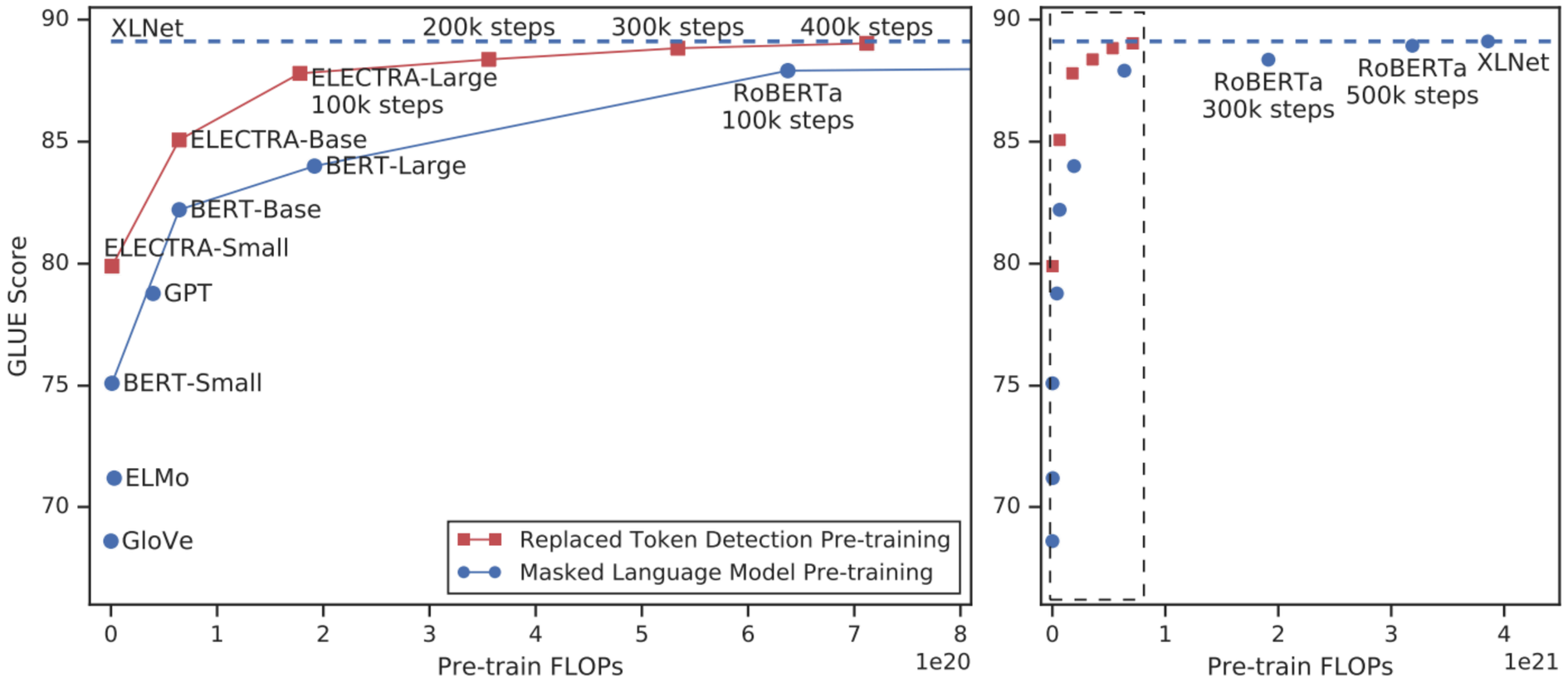
ELECTRA: idea



$$\mathcal{L}_{\text{MLM}}(\boldsymbol{x}, \theta_G) = \mathbb{E}\left(\sum_{i \in \boldsymbol{m}} -\log p_G(x_i | \boldsymbol{x}^{\text{masked}})\right)$$

$$\mathcal{L}_{\text{Disc}}(\boldsymbol{x}, \theta_D) = \mathbb{E}\left(\sum_{t=1}^{n} -\mathbb{1}(x_t^{\text{corrupt}} = x_t) \log D(\boldsymbol{x}^{\text{corrupt}}, t) - \mathbb{1}(x_t^{\text{corrupt}} \neq x_t) \log(1 - D(\boldsymbol{x}^{\text{corrupt}}, t))\right)$$

ELECTRA: results



- Less gap between training and testing
- Predictions on all tokens -> training is faster









Miles Brundage
@Miles_Brundage

2018: Language model papers have to introduce Sesame Street-related acronyms

2019: Language model papers need Sesame Street jokes in the title, all talks need at least one Sesame Street image.

2020: ACL/NAACL co-located with Sesame Street convention, Big Bird gives a keynote.

293 2:46 AM - Jun 12, 2019

57 people are talking about this

References

- Basics:
  - Transformer - https://arxiv.org/pdf/1706.03762.pdf
  - ELMO - https://arxiv.org/pdf/1802.05365.pdf
  - GPT - https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf
  - BERT - https://arxiv.org/pdf/1810.04805.pdf
- BERTology - https://arxiv.org/pdf/2002.12327.pdf
- Analysis of BERT (and Transformer):
  - https://arxiv.org/pdf/1909.00512.pdf
  - https://arxiv.org/pdf/1909.10430.pdf
  - https://arxiv.org/pdf/1908.08593v2.pdf
  - https://arxiv.org/pdf/1906.04341v1.pdf

References

- Analysis of BERT (and Transformer):
  - https://arxiv.org/pdf/1908.04211.pdf
  - https://arxiv.org/abs/1905.10650
  - https://arxiv.org/pdf/2002.10260.pdf
- RoBERTa - https://arxiv.org/abs/1907.11692
- ALBERT - https://arxiv.org/pdf/1909.11942.pdf
- Other BERT modifications:
  - https://proceedings.mlr.press/v97/gong19a/gong19a.pdf
  - https://arxiv.org/abs/1904.09223
  - https://arxiv.org/abs/1907.12412
  - https://arxiv.org/abs/1907.10529

References

- Other BERT modifications:
  - https://proceedings.mlr.press/v97/gong19a/gong19a.pdf
  - https://arxiv.org/abs/1904.09223
  - https://arxiv.org/abs/1907.12412
  - https://arxiv.org/abs/1907.10529
  - https://arxiv.org/abs/1908.04577