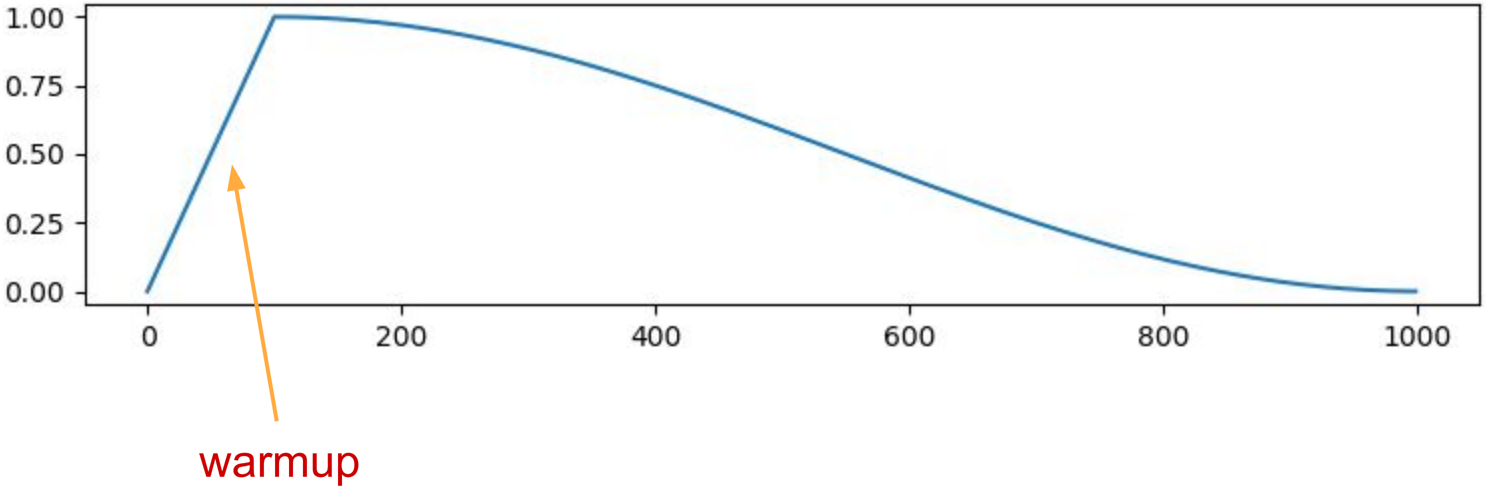
Трюки для трансформеров

ВШЭ ФКН, NLP

Шабалин Александр

Warmup

Постепенное увеличение скорости обучения на ранних шагах оптимизации.



Warmup

Трансформеры не учатся без warmup!

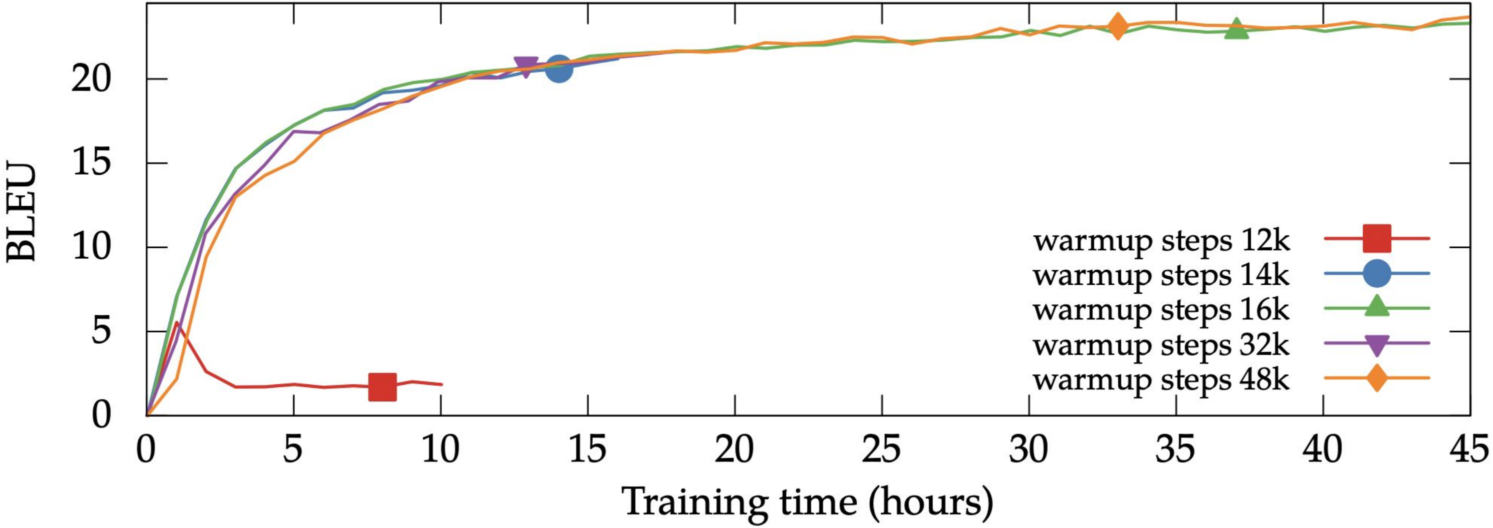


Figure 8: Effect of the warmup steps on a single GPU. All trained on CzEng 1.0 with the default batch size (1500) and learning rate (0.20).

https://arxiv.org/pdf/1804.00247.pdf

Batch size

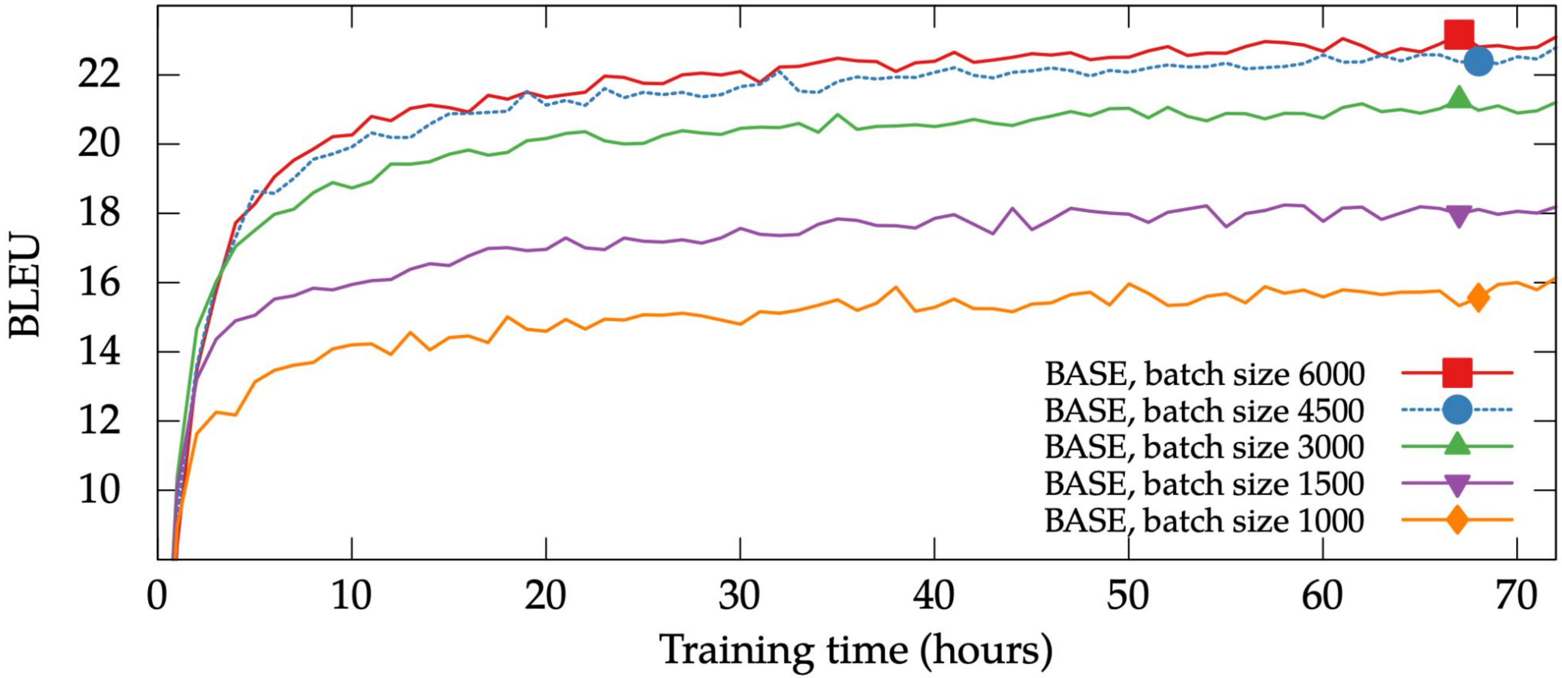


Figure 5: Effect of the batch size with the BASE model. All trained on a single GPU.

https://arxiv.org/pdf/1804.00247.pdf

Batch size

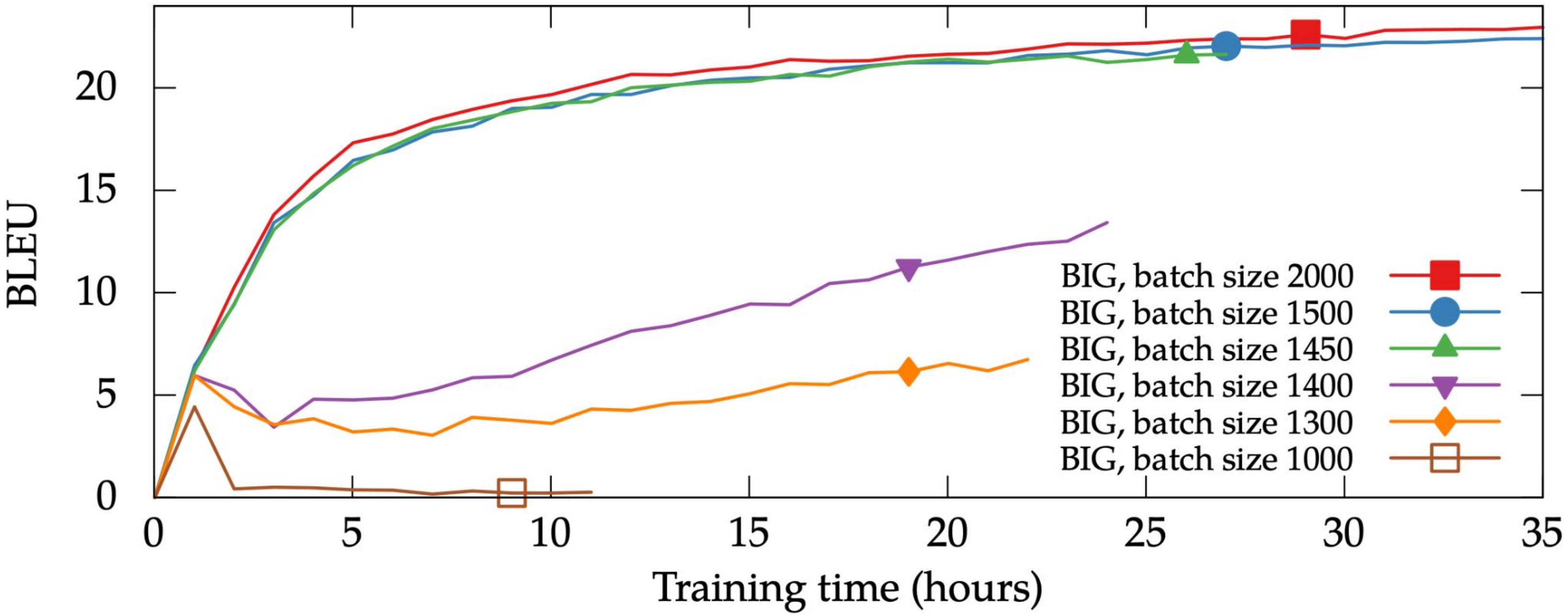


Figure 6: Effect of the batch size with the BIG model. All trained on a single GPU.

Еще трюки

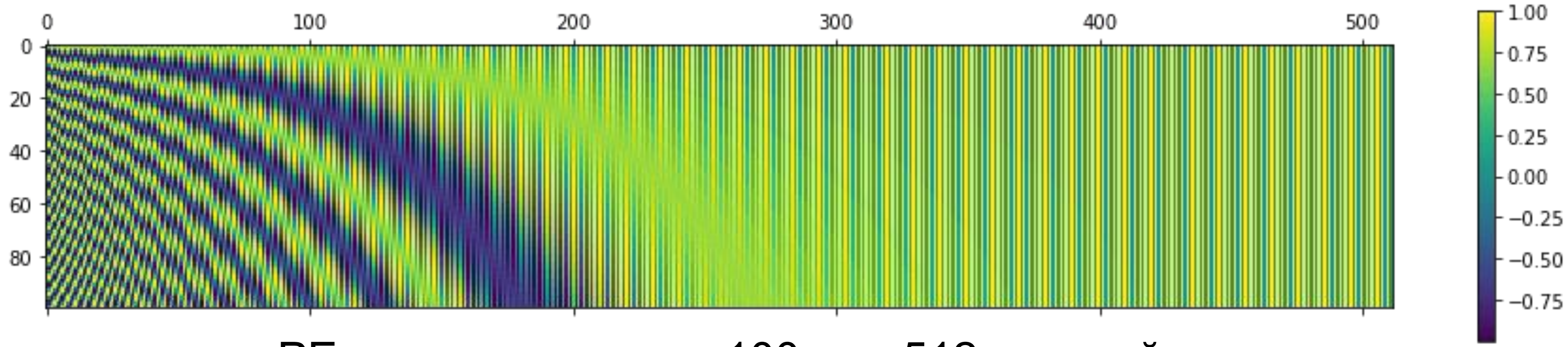
- SGD не работает, используйте Adam (AdamW, LAMB, AdaFactor, …)
- Аккумулирование градиентов
- Группировка текстов по длине
- Gradient clipping
- Mixed-precision
- Лучше мало чистых текстов, чем много грязных
- Для warmup можно взять меньше батчи и короче тексты

Positional encodings, Google

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

- Уникально кодируют все позиции
- Все значения лежат в [-1; 1]
- Можно восстановить относительное расположение токенов:

  $PE(p+k) = R \cdot PE(p)$, где R – матрица поворота



PE векторы размера 100 для 512 позиций

https://arxiv.org/pdf/1706.03762.pdf

Relative Position Encodings (RPE), Google

Явно передаем в модель информацию об относительном расположении токенов

$$\text{RelativeAttention} = \text{Softmax}\left(\frac{QK^\top + S^{rel}}{\sqrt{D_h}}\right)V.$$

$S^{rel} = QR$, где $R \in \mathbb{R}^{[L \times L \times d]}$ – это матрица с эмбеддингами разностей позиций [query pos – key pos]

https://arxiv.org/pdf/1809.04281.pdf
https://arxiv.org/pdf/1803.02155.pdf

Rotary Position Embeddings (RoPE)

- Выходы Q и K матриц домножаются на блочную матрицу поворота
- m и n – позиции токенов

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = (\boldsymbol{R}^d_{\Theta,m} \boldsymbol{W}_q \boldsymbol{x}_m)^\intercal (\boldsymbol{R}^d_{\Theta,n} \boldsymbol{W}_k \boldsymbol{x}_n) = \boldsymbol{x}^\intercal \boldsymbol{W}_q R^d_{\Theta,n-m} \boldsymbol{W}_k \boldsymbol{x}_n$$

$$\boldsymbol{R}^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

$$\Theta = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, ..., d/2]\}$$

https://arxiv.org/pdf/2104.09864.pdf

Rotary Position Embeddings (RoPE)

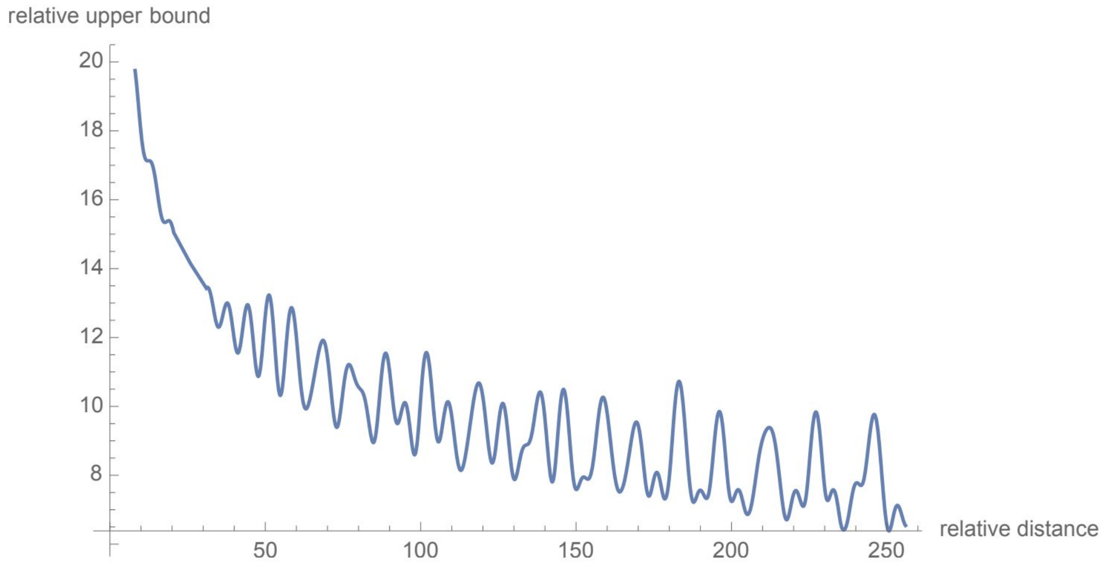
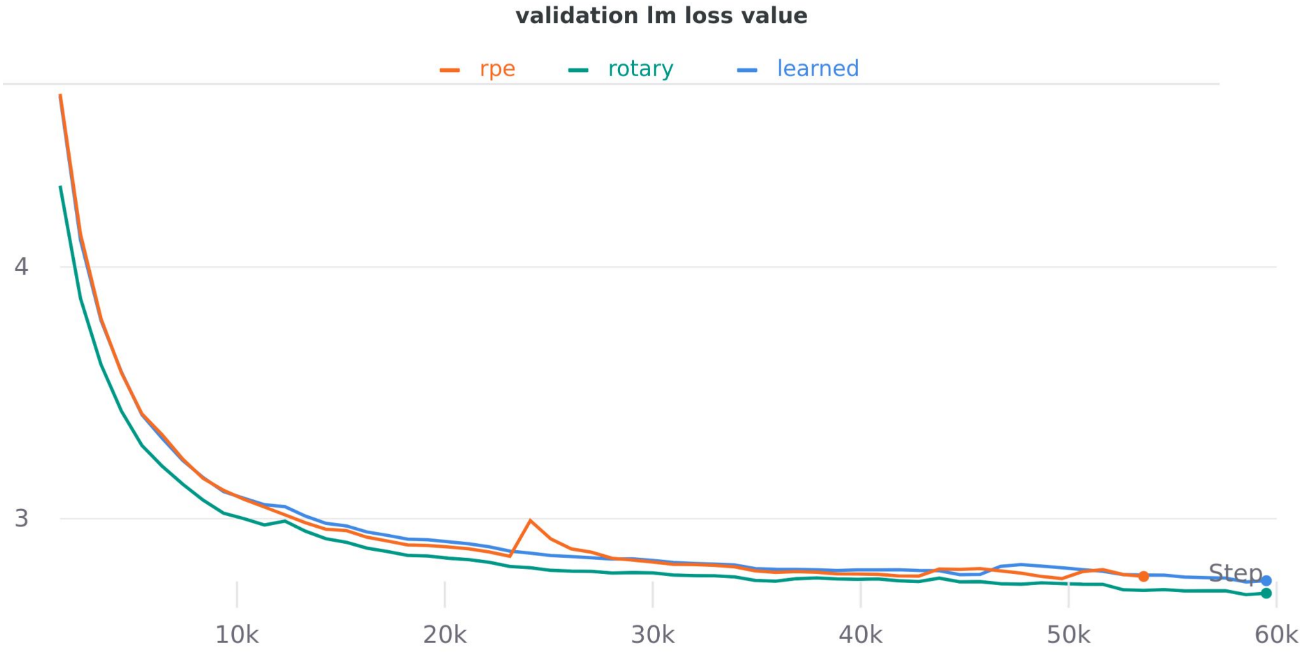


Figure 2: Long-term decay of RoPE.

https://arxiv.org/pdf/2104.09864.pdf

Rotary Position Embeddings (RoPE)

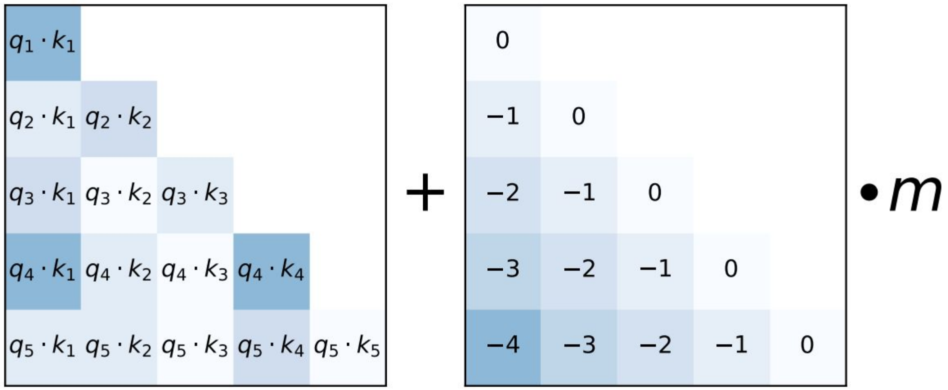


https://arxiv.org/pdf/2104.09864.pdf

Attention with Linear Biases (ALiBi), Meta

Добавляем линейный сдвиг к score в attention.

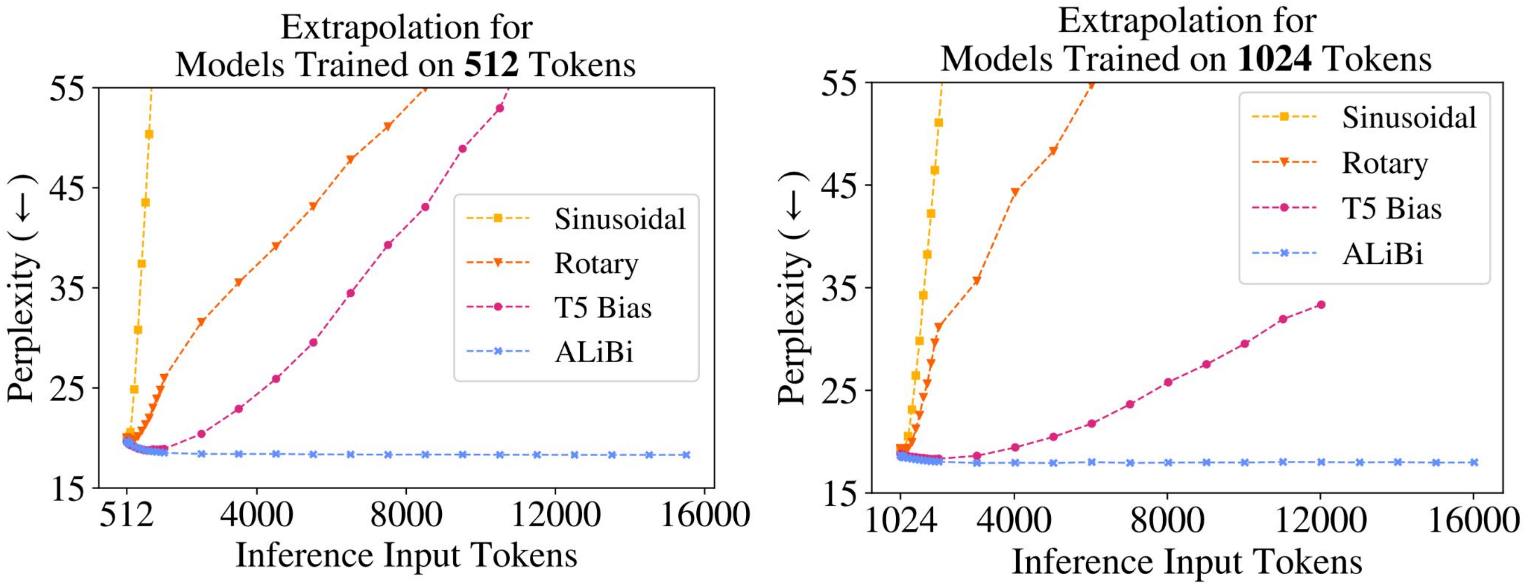
$$\text{softmax}(\mathbf{q}_i\mathbf{K}^\top + m \cdot [-(i-1), ..., -2, -1, 0])$$



Для каждой головы свой $m_i = 2^{-\frac{i}{2}}$

https://arxiv.org/pdf/2108.12409.pdf

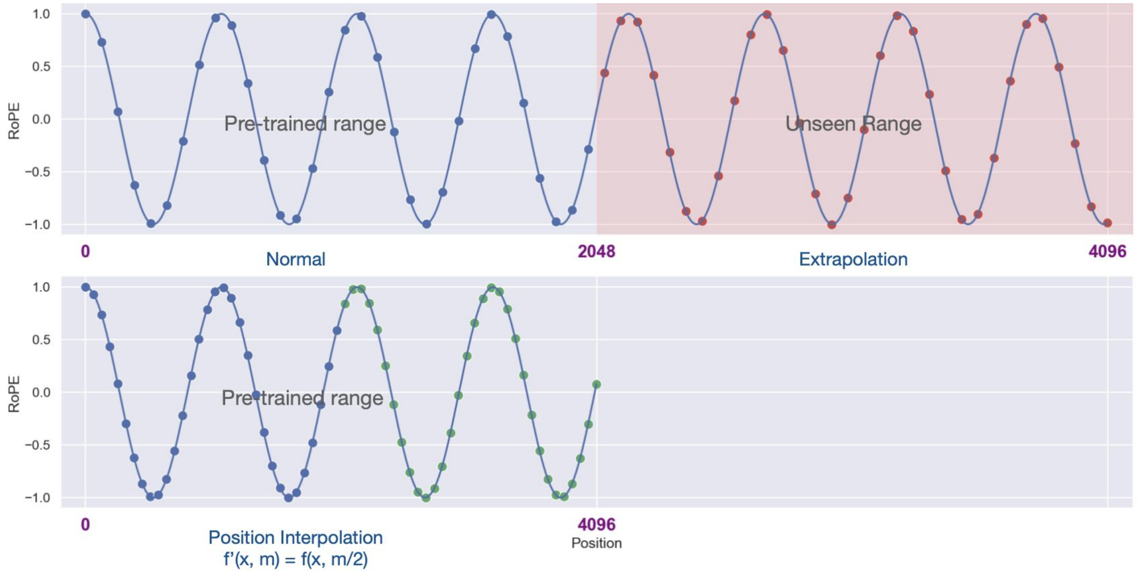
Attention with Linear Biases (ALiBi), Meta



https://arxiv.org/pdf/2108.12409.pdf

Position Interpolation, Meta

Если модель училась на длине текста L, то на длине текста L’ > L она не будет работать.



Идея: заменим номер позиции с S на LS / L’

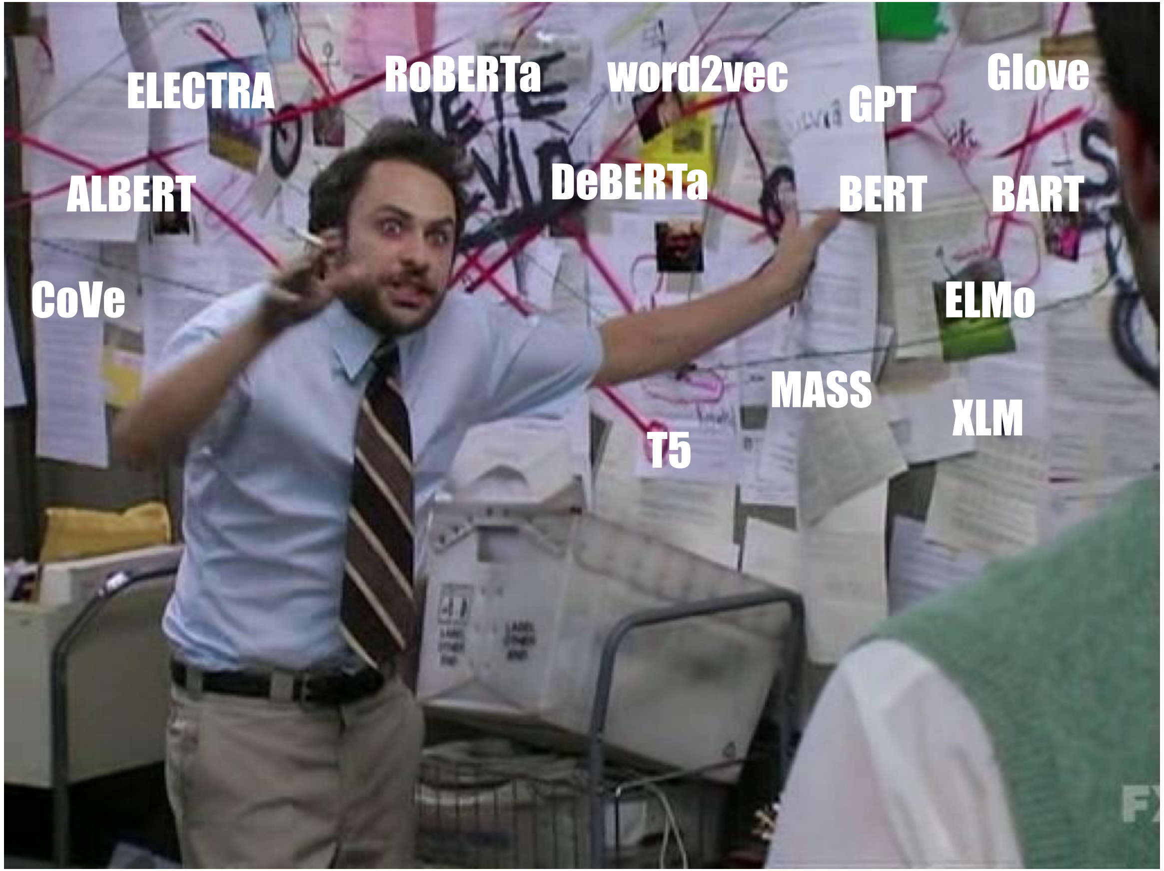
https://arxiv.org/pdf/2306.15595.pdf

Position Interpolation, Meta

RoPE is used here

	Model		Evaluation Context Window Size				
Size	Context Window	Method	2048	4096	8192	16384	32768
7B	2048	None	7.20	$> 10^3$	$> 10^3$	$> 10^3$	$> 10^3$
7B	8192	FT	7.21	7.34	7.69	-	-
7B	8192	PI	7.13	6.96	6.95	-	-
7B	16384	PI	7.11	6.93	6.82	6.83	-
7B	32768	PI	7.23	7.04	6.91	6.80	6.77
13B	2048	None	6.59	-	-	-	-
13B	8192	FT	6.56	6.57	6.69	-	-
13B	8192	PI	6.55	6.42	6.42	-	-
13B	16384	PI	6.56	6.42	6.31	6.32	-
13B	32768	PI	6.54	6.40	6.28	6.18	6.09
33B	2048	None	5.82	-	-	-	-
33B	8192	FT	5.88	5.99	6.21	-	-
33B	8192	PI	5.82	5.69	5.71	-	-
33B	16384	PI	5.87	5.74	5.67	5.68	-
65B	2048	None	5.49	-	-	-	-
65B	8192	PI	5.42	5.32	5.37	-	-

https://arxiv.org/pdf/2306.15595.pdf


ELECTRA
RoBERTa
word2vec
GPT
Glove
ALBERT
DeBERTa
BERT
BART
CoVe
ELMo
MASS
XLM
T5
FX

RoBERTa, 2019 (Facebook)

Robustly Optimized BERT Approach

- Train the model longer
- Use bigger batches
- Collect more data
- Dynamic masking
- Without NSP

Masking	**SQuAD 2.0**	**MNLI-m**	**SST-2**
reference	76.3	84.3	92.8
Our reimplementation:			
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

https://arxiv.org/pdf/1907.11692.pdf

RoBERTa, 2019 (Facebook)

Robustly Optimized BERT Approach

- Train the model longer
- Use bigger batches
- Collect more data
- Dynamic masking
- Without NSP

Masking	SQuAD 2.0	MNLI-m	SST-2
reference	76.3	84.3	92.8
Our reimplementation:			
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

bsz	steps	lr	ppl	MNLI-m	SST-2
256	1M	1e-4	3.99	84.7	92.7
2K	125K	7e-4	**3.68**	**85.2**	**92.9**
8K	31K	1e-3	3.77	84.6	92.8

https://arxiv.org/pdf/1907.11692.pdf

RoBERTa, 2019 (Facebook)

Robustly Optimized BERT Approach

- Train the model longer
- Use bigger batches
- Collect more data
- Dynamic masking
- Without NSP

Masking	SQuAD 2.0	MNLI-m	SST-2
reference	76.3	84.3	92.8
Our reimplementation:			
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

bsz	steps	lr	ppl	MNLI-m	SST-2
256	1M	1e-4	3.99	84.7	92.7
2K	125K	7e-4	**3.68**	**85.2**	**92.9**
8K	31K	1e-3	3.77	84.6	92.8

https://arxiv.org/pdf/1907.11692.pdf

RoBERTa, 2019 (Facebook)

Robustly Optimized BERT Approach

- BookCorpus + Wikipedia (16GB)
- CC-News (76GB)
- OpenWebText (38GB)
- Stories (31GB)

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
$\text{XLNet}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

https://arxiv.org/pdf/1907.11692.pdf

ALBERT, 2019 (Google)

A Lite BERT

- We can improve quality by scaling a model, but it requires higher computational costs.
- ALBERT propose embedding matrix factorization and parameter sharing to make model lighter.
- Replace NSP loss with SOP loss.

https://arxiv.org/pdf/1909.11942.pdf

ALBERT, 2019 (Google)

A Lite BERT

- We can improve quality by scaling a model, but it requires higher computational costs.
- ALBERT propose embedding matrix factorization and parameter sharing to make model lighter.
- Replace NSP loss with SOP loss.

$$x \in OHE(V)$$

$$emb(x) = x \underbrace{A}_{[V,H]}$$

$$emb_{fact}(x) = x \underbrace{A}_{[V,E]} \underbrace{B}_{[E,H]}$$

https://arxiv.org/pdf/1909.11942.pdf

ALBERT, 2019 (Google)

A Lite BERT

- We can improve quality by scaling a model, but it requires higher computational costs.
- ALBERT propose embedding matrix factorization and parameter sharing to make model lighter.
- Replace NSP loss with SOP loss.

$$x \in OHE(V)$$

$$emb(x) = x \underbrace{A}_{[V,H]}$$

$$emb_{fact}(x) = x \underbrace{A}_{[V,E]} \underbrace{B}_{[E,H]}$$

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

https://arxiv.org/pdf/1909.11942.pdf

ALBERT, 2019 (Google)

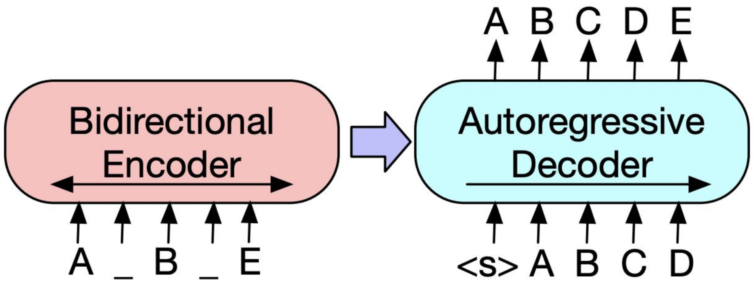
A Lite BERT

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**

BART, 2019 (Facebook)

Bidirectional and Auto-Regressive Transformers

- A denoising autoencoder for pretraining seq2seq models.
- Attempts to connect BERT (due to the bidirectional encoder) with GPT (with the left-to-right decoder).

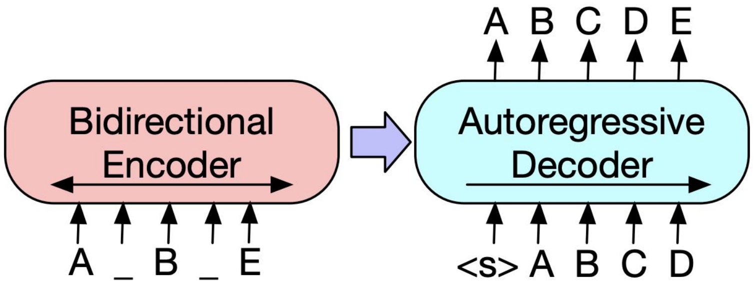


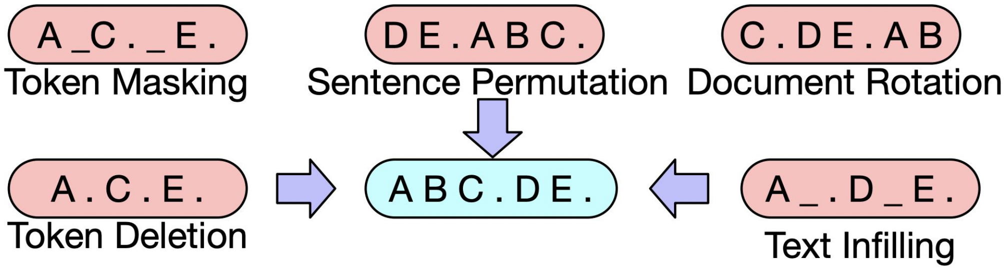
https://arxiv.org/pdf/1910.13461.pdf

BART, 2019 (Facebook)

Bidirectional and Auto-Regressive Transformers

- A denoising autoencoder for pretraining seq2seq models.
- Attempts to connect BERT (due to the bidirectional encoder) with GPT (with the left-to-right decoder).



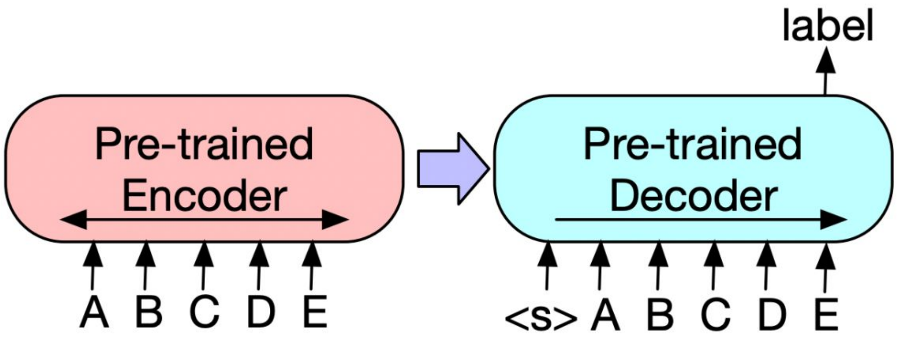


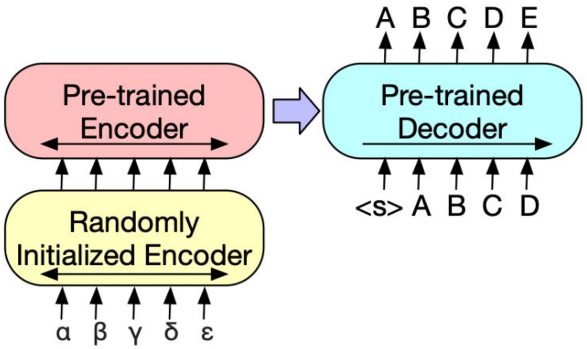
https://arxiv.org/pdf/1910.13461.pdf

BART, 2019 (Facebook)

Bidirectional and Auto-Regressive Transformers

- **Token Classification:** Top hidden state of the decoder is used as a representation for each word. This representation is used to classify the token.
- **Sequence Generation Tasks:** Standard autoregressive scheme.
- **Machine Translation:** BART's encoder embedding layer is replaced with a randomly initialized encoder. The new encoder can use a separate vocabulary from the original BART model.





https://arxiv.org/pdf/1910.13461.pdf

BART, 2019 (Facebook)

Bidirectional and Auto-Regressive Transformers

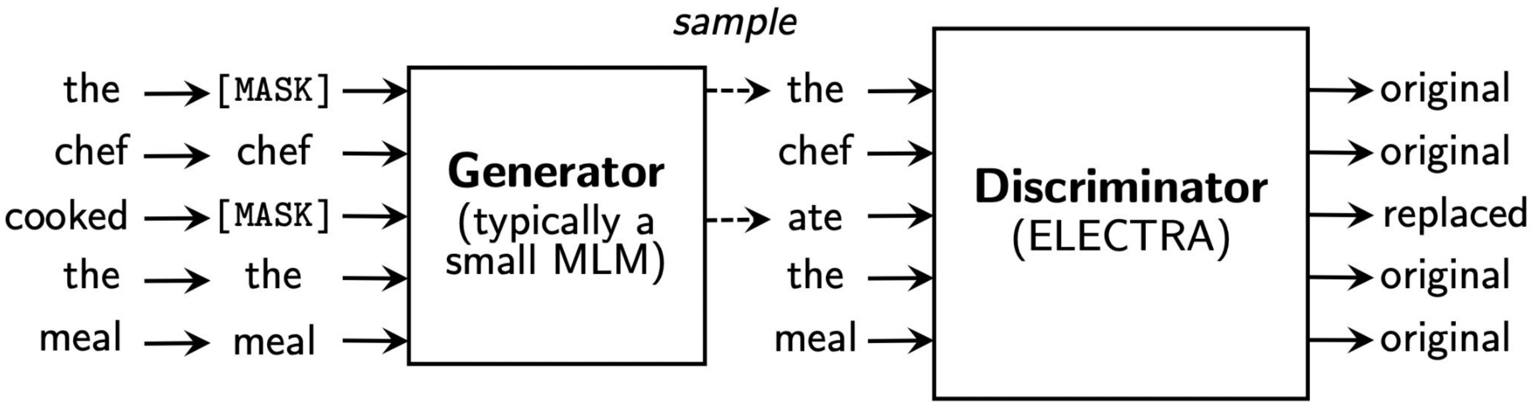
Model	SQuAD 1.1 F1	MNLI Acc	ELI5 PPL	XSum PPL	ConvAI2 PPL	CNN/DM PPL
BERT Base (Devlin et al., 2019)	88.5	**84.3**	-	-	-	-
Masked Language Model $(BERT)$	90.0	83.5	24.77	7.87	12.59	7.06
Masked Seq2seq $(MASS)$	87.0	82.1	23.40	6.80	11.43	6.19
Language Model (GPT)	76.7	80.1	**21.40**	7.00	11.51	6.56
Permuted Language Model $(XLNet)$	89.1	83.7	24.03	7.69	12.23	6.96
Multitask Masked Language Model	89.2	82.4	23.73	7.50	12.39	6.74
BART Base						
w/ Token Masking	90.4	84.1	25.05	7.08	11.73	6.10
w/ Token Deletion	90.4	84.1	24.61	6.90	11.46	5.87
w/ Text Infilling	**90.8**	84.0	24.26	**6.61**	**11.05**	5.83
w/ Document Rotation	77.2	75.3	53.69	17.14	19.87	10.59
w/ Sentence Shuffling	85.4	81.5	41.87	10.93	16.67	7.89
w/ Text Infilling + Sentence Shuffling	**90.8**	83.8	24.17	6.62	11.12	**5.41**

https://arxiv.org/pdf/1910.13461.pdf

ELECTRA, 2020 (Google)

Efficiently Learning an Encoder that Classifies Token Replacements Accurately

- Has two encoder networks: generator **G** (small) and discriminator **D** (ELECTRA)
- Generator learns on MLM objective and predicts masked tokens
- Discriminator learns to distinguish generated tokens from original



https://arxiv.org/pdf/2003.10555.pdf

ELECTRA, 2020 (Google)

Efficiently Learning an Encoder that Classifies Token Replacements Accurately

$$m_i \sim \text{unif}\{1, n\} \text{ for } i = 1 \text{ to } k \qquad \boldsymbol{x}^{\text{masked}} = \text{REPLACE}(\boldsymbol{x}, \boldsymbol{m}, \texttt{[MASK]})$$

$$\hat{x}_i \sim p_G(x_i|\boldsymbol{x}^{\text{masked}}) \text{ for } i \in \boldsymbol{m} \qquad \boldsymbol{x}^{\text{corrupt}} = \text{REPLACE}(\boldsymbol{x}, \boldsymbol{m}, \hat{\boldsymbol{x}})$$

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

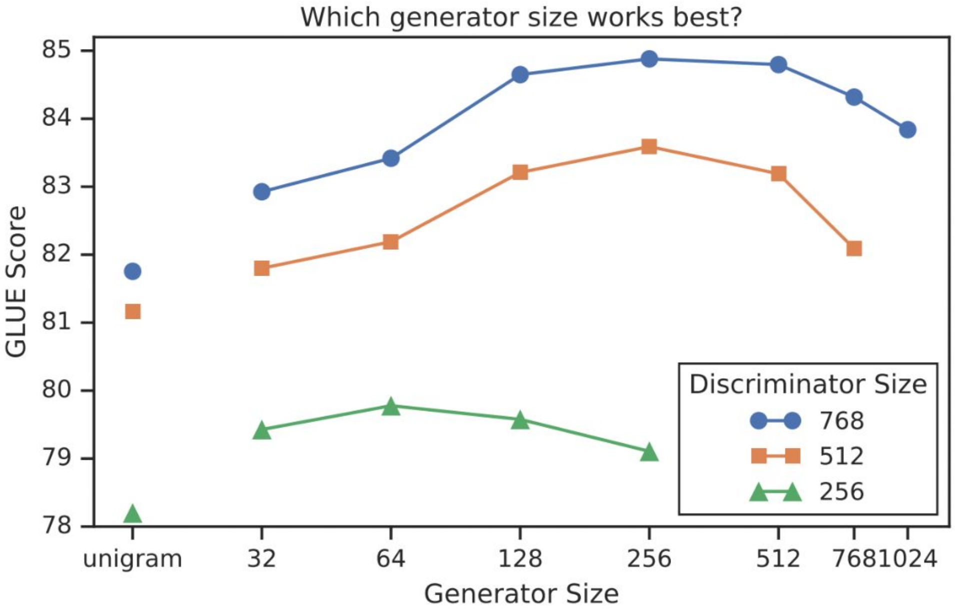
$$\mathcal{L}_{\text{Disc}}(\boldsymbol{x}, \theta_D) = \mathbb{E}\left(\sum_{t=1}^{n} -\mathbb{1}(x_t^{\text{corrupt}} = x_t)\log D(\boldsymbol{x}^{\text{corrupt}}, t) - \mathbb{1}(x_t^{\text{corrupt}} \neq x_t)\log(1 - D(\boldsymbol{x}^{\text{corrupt}}, t))\right)$$

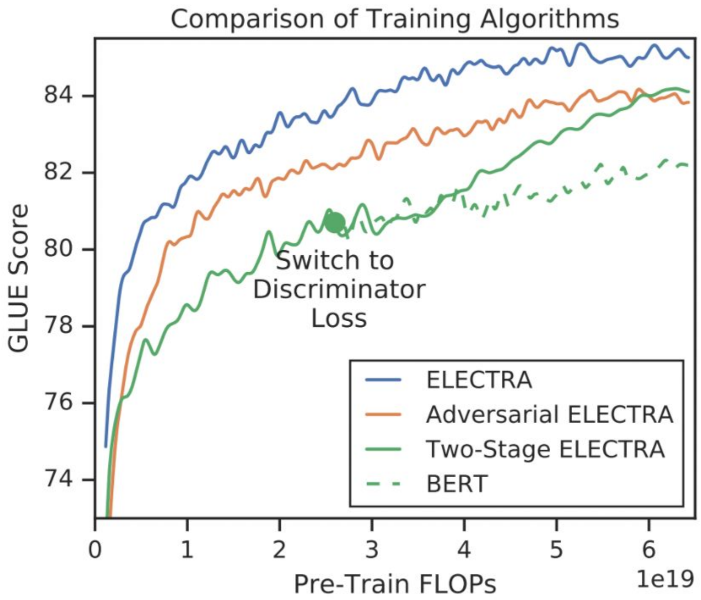
$$\min_{\theta_G, \theta_D} \sum_{\boldsymbol{x} \in \mathcal{X}} \mathcal{L}_{\text{MLM}}(\boldsymbol{x}, \theta_G) + \lambda \mathcal{L}_{\text{Disc}}(\boldsymbol{x}, \theta_D)$$

https://arxiv.org/pdf/2003.10555.pdf

ELECTRA, 2020 (Google)

Efficiently Learning an Encoder that Classifies Token Replacements Accurately





https://arxiv.org/pdf/2003.10555.pdf

ELECTRA, 2020 (Google)

Efficiently Learning an Encoder that Classifies Token Replacements Accurately

Model	Train FLOPs	Params	SQuAD 1.1 dev		SQuAD 2.0 dev		SQuAD 2.0 test	
			EM	F1	EM	F1	EM	F1
BERT-Base	6.4e19 (0.09x)	110M	80.8	88.5	–	–	–	–
BERT	1.9e20 (0.27x)	335M	84.1	90.9	79.0	81.8	80.0	83.0
SpanBERT	7.1e20 (1x)	335M	88.8	94.6	85.7	88.7	85.7	88.7
XLNet-Base	6.6e19 (0.09x)	117M	81.3	–	78.5	–	–	–
XLNet	3.9e21 (5.4x)	360M	**89.7**	**95.1**	87.9	**90.6**	87.9	90.7
RoBERTa-100K	6.4e20 (0.90x)	356M	–	94.0	–	87.7	–	–
RoBERTa-500K	3.2e21 (4.5x)	356M	88.9	94.6	86.5	89.4	86.8	89.8
ALBERT	3.1e22 (44x)	235M	89.3	94.8	87.4	90.2	88.1	90.9
BERT (ours)	7.1e20 (1x)	335M	88.0	93.7	84.7	87.5	–	–
ELECTRA-Base	6.4e19 (0.09x)	110M	84.5	90.8	80.5	83.3	–	–
ELECTRA-400K	7.1e20 (1x)	335M	88.7	94.2	86.9	89.6	–	–
ELECTRA-1.75M	3.1e21 (4.4x)	335M	**89.7**	94.9	**88.0**	**90.6**	**88.7**	**91.4**

https://arxiv.org/pdf/2003.10555.pdf

Conclusions

- More data is better
- Bigger model is better
- The choice of pre-train task hugely depends on the downstream task