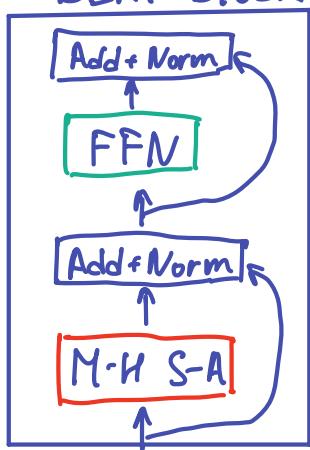
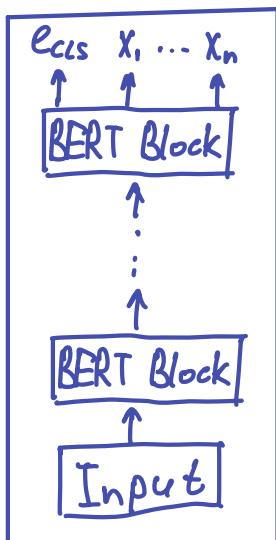


BERTology

BERT Block



BERT



Что мы можем узкать
о том, как работает BERT?

- 1) Можем посмотреть на слой Multi-head self-attention. Оказывается, что:
 - Каждая голова "смотрит" на определенные зависимости между токенами.
 - Ранько не все головы нужны. После обучения можно удалить большинство из них и не потерять в качестве.

2) Linear Probing

Узнав многую голову
проверяя выходов любого слоя
модели. Так мы можем узкать,
каким ли выходам нужна
загадка информация.

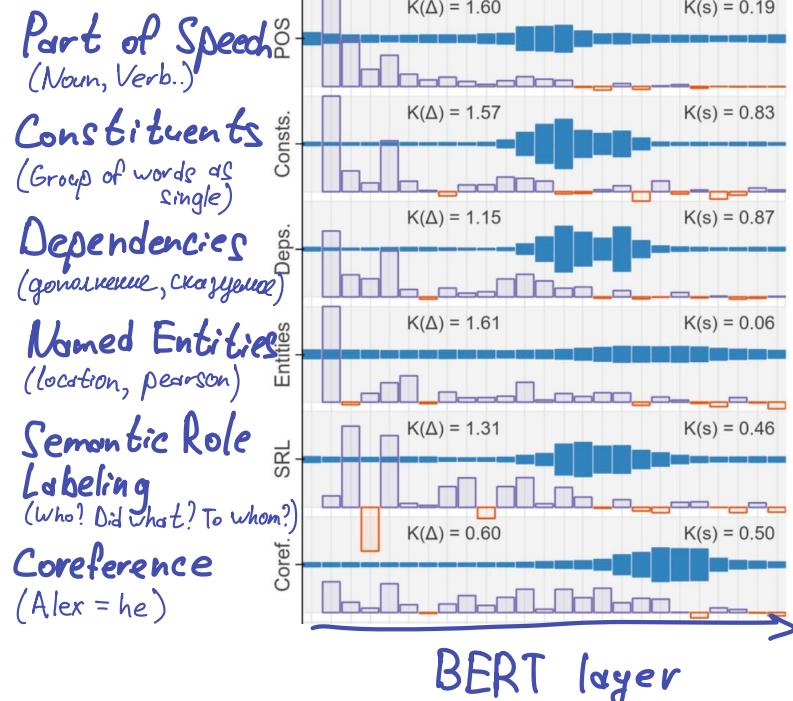
Вариантные задачи:

- Длина текста
- Позиция слова в тексте
- NER

На графике показано в
каких слоях BERT хранится
информация для решения
каждой задачи.

Оказалось, что также загадка,
также зависит от входа леммы
информации для нее.

Слово
смыс-
ла
загадки



Как добувают модель?

1) Linear Probing.

Обучаем линейную голову на выходах последнего слоя.

Веса модели заново определяются.

- + Эффективно по нагрузке
- + Очень быстро, если сохранить выходы модели на диск.
- + Хорошо работает, если применяется на OOD тестовых данных.
- Недостаточно хорошо на ID тестовых данных

2) Fine-tuning

Обучаем всю модель целиком с новой головой

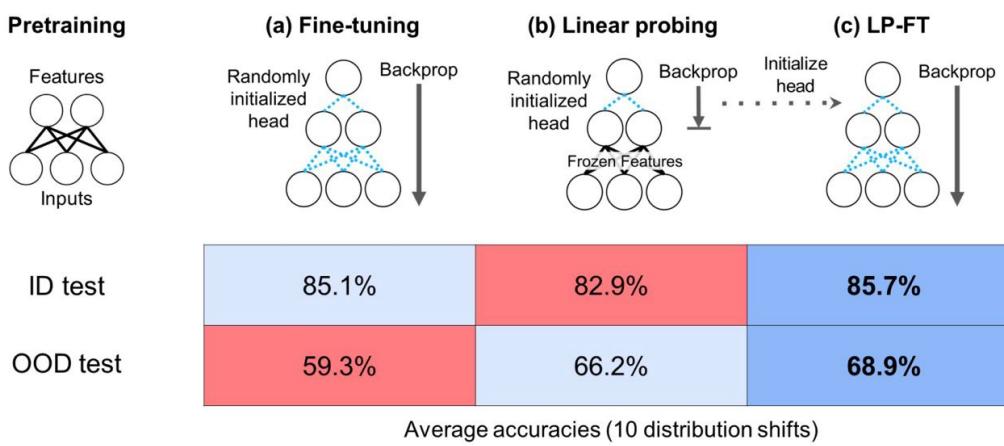
- + Очень хорошо работает на ID тестовых данных
- Требует много вычислений и памяти.

3) Linear Probing - Fine-tuning (LP-FT)

1. Учим голову (LP)

2. Учим всю модель с этой головой (FT)

- + Работает лучше и на ID, и на OOD тестовых данных.
- Требует все выше вычислений и памяти

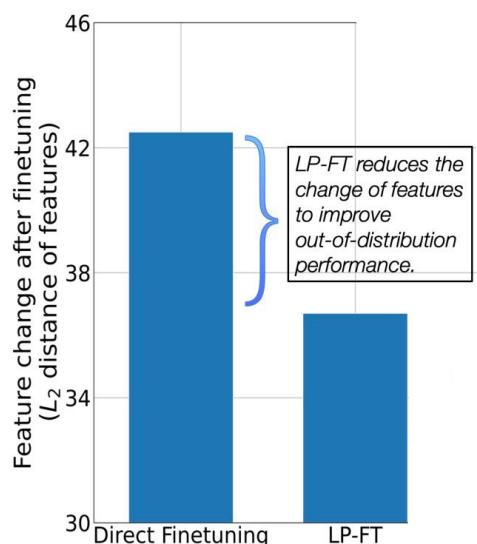


Поговорим?

Из-за инициализации случайной головой на первых итерациях Fine-tuning'а веса градиенты очень велики \Rightarrow веса модели меняются.

Поэтому модель переобучается под ID и теряет свою генерализацию.

Если инициализировать не случайной головой, а предобученной, веса будут меняться меньше, но при этом модель обучится под задачу и генерализацию сохранила



Расстояние между весами до и после обучения

4) Parameter-Efficient Fine-tuning (PEFT)

	#params	Data
GPT-1	117M	5 GB
GPT-2	1.5B	40 GB
GPT-3	175B	45 TB

Современные модели невозможны добывать целиком из-за ограниченного веса градиентов.

\Rightarrow надо обучать кебельное подмножество параметров.

4.1) Bias-terms Fine-tuning (BitFit)

Attention:

$$Q(x) = W_Q x + b_Q$$

$$K(x) = W_K x + b_K$$

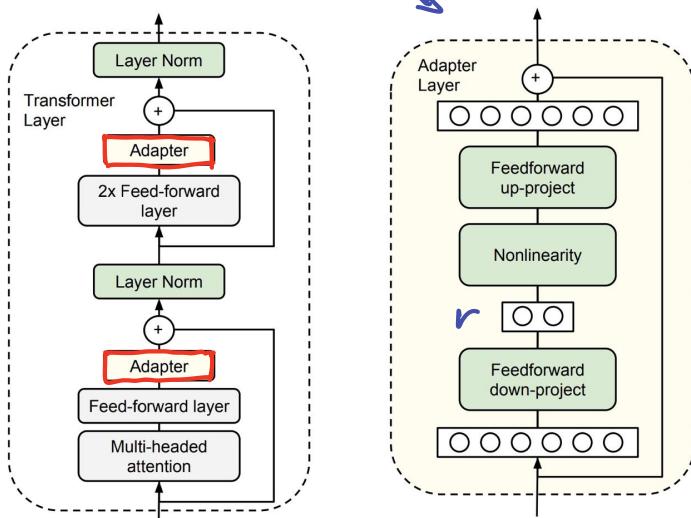
$$V(x) = W_V x + b_V$$

Чему малко сдвиги линейных слоев Self-attention (Q, K, V) и головы.

4.2) Adapters

Вставляем 2-слойное MLP между Self-Attention и FFN и уменьшаем их и голову.

Число обучаемых параметров контролируется размером доминанты r .



4.3) Low Rank Adaptation (LoRa)

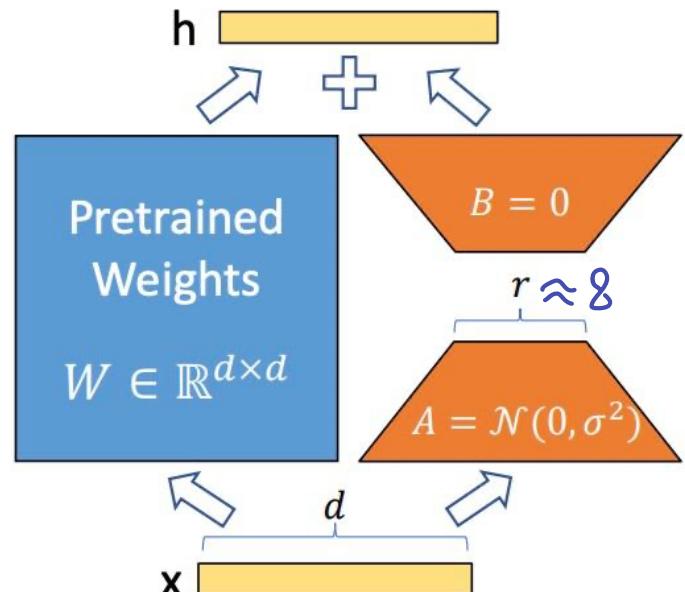
Идея: Градиентный шаг

$$w' \leftarrow w - \eta \nabla_w$$

$\underbrace{_{A \cdot B}}$

Явно будем использовать градиенты в виде $A \cdot B$
 $A \in \mathbb{R}^{[d \times r]}, B \in \mathbb{R}^{[r \times d]}$

Уменьшили массу w_k и w_v
 в столбцах выделенных.



$$w'_k \leftarrow w_k + A_k \cdot B_k$$

$$w'_v \leftarrow w_v + A_v \cdot B_v$$

Число обучаемых параметров регулируется r .

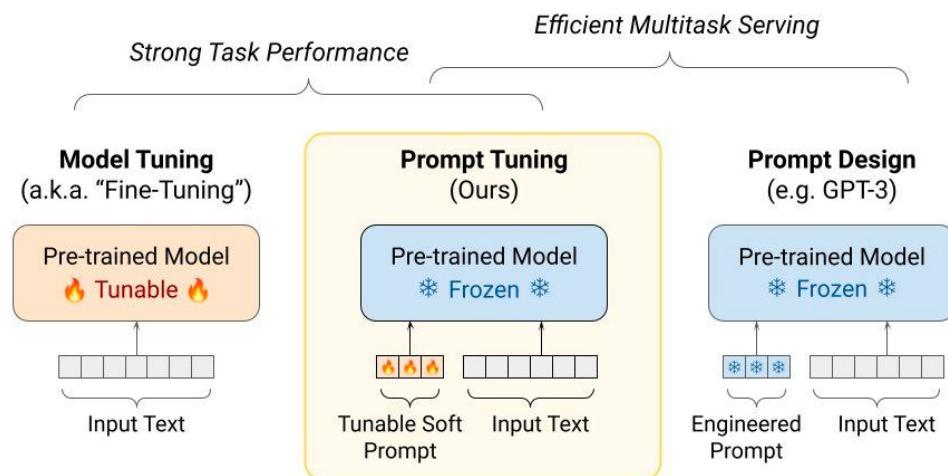
Инициализация: $A \sim \mathcal{N}(0, \sigma^2)$; $B = 0$, чтобы в начале добавка ничего не меняла.

Model & Method	# Trainable Parameters	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
RoB _{base} (FT)*	125.0M	87.6	94.8	90.2	63.6	92.8	91.9	78.7	91.2	86.4
RoB _{base} (BitFit)*	0.1M	84.7	93.7	92.7	62.0	91.8	84.0	81.5	90.8	85.2
RoB _{base} (Adpt ^D)*	0.3M	87.1 _{±.0}	94.2 _{±.1}	88.5 _{±1.1}	60.8 _{±.4}	93.1 _{±.1}	90.2 _{±.0}	71.5 _{±2.7}	89.7 _{±.3}	84.4
RoB _{base} (Adpt ^D)*	0.9M	87.3 _{±.1}	94.7 _{±.3}	88.4 _{±.1}	62.6 _{±.9}	93.0 _{±.2}	90.6 _{±.0}	75.9 _{±2.2}	90.3 _{±.1}	85.4
RoB _{base} (LoRA)	0.3M	87.5 _{±.3}	95.1 _{±.2}	89.7 _{±.7}	63.4 _{±1.2}	93.3 _{±.3}	90.8 _{±.1}	86.6 _{±.7}	91.5 _{±.2}	87.2
RoB _{large} (FT)*	355.0M	90.2	96.4	90.9	68.0	94.7	92.2	86.6	92.4	88.9
RoB _{large} (LoRA)	0.8M	90.6 _{±.2}	96.2 _{±.5}	90.9 _{±1.2}	68.2 _{±1.9}	94.9 _{±.3}	91.6 _{±.1}	87.4 _{±2.5}	92.6 _{±.2}	89.0
RoB _{large} (Adpt ^P)†	3.0M	90.2 _{±.3}	96.1 _{±.3}	90.2 _{±.7}	68.3 _{±1.0}	94.8 _{±.2}	91.9 _{±.1}	83.8 _{±2.9}	92.1 _{±.7}	88.4
RoB _{large} (Adpt ^P)†	0.8M	90.5 _{±.3}	96.6 _{±.2}	89.7 _{±1.2}	67.8 _{±2.5}	94.8 _{±.3}	91.7 _{±.2}	80.1 _{±2.9}	91.9 _{±.4}	87.9
RoB _{large} (Adpt ^H)†	6.0M	89.9 _{±.5}	96.2 _{±.3}	88.7 _{±2.9}	66.5 _{±4.4}	94.7 _{±.2}	92.1 _{±.1}	83.4 _{±1.1}	91.0 _{±1.7}	87.8
RoB _{large} (Adpt ^H)†	0.8M	90.3 _{±.3}	96.3 _{±.5}	87.7 _{±1.7}	66.3 _{±2.0}	94.7 _{±.2}	91.5 _{±.1}	72.9 _{±2.9}	91.5 _{±.5}	86.4
RoB _{large} (LoRA)†	0.8M	90.6 _{±.2}	96.2 _{±.5}	90.2 _{±1.0}	68.2 _{±1.9}	94.8 _{±.3}	91.6 _{±.2}	85.2 _{±1.1}	92.3 _{±.5}	88.6
DeB _{XXL} (FT)*	1500.0M	91.8	97.2	92.0	72.0	96.0	92.7	93.9	92.9	91.1
DeB _{XXL} (LoRA)	4.7M	91.9 _{±.2}	96.9 _{±.2}	92.6 _{±.6}	72.4 _{±1.1}	96.0 _{±.1}	92.9 _{±.1}	94.9 _{±.4}	93.0 _{±.2}	91.3

Сравнение BitFit, Adapters, LoRA и Fine-tuning (FT)

4.4) Prompt Tuning

Идея: Большие языковые модели (LLM) могут генерировать текст по примеру. При этом от изменения контекста меняется качество решения задачи. Будет учтено контекст, а не подбирают его руками!



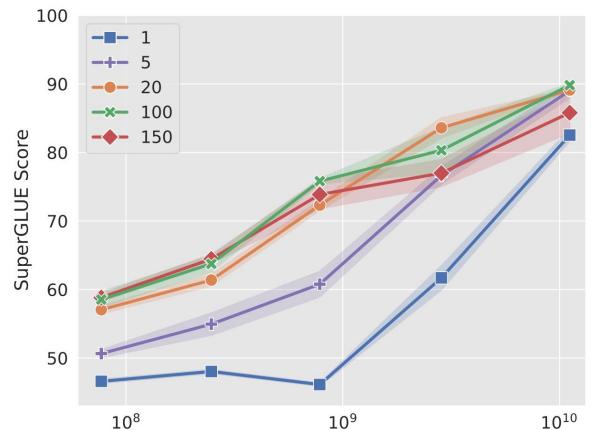
Таким образом будем учить Р задеддиков, которые будем добавлять в начало.

=> Обучаемые параметры:

- количество задеддиков $M \in \mathbb{R}^{[prod]}$
- голова модели.

Dataset	Domain	Model	Prompt	Δ
SQuAD	Wiki	94.9 ± 0.2	94.8 ± 0.1	-0.1
TextbookQA	Book	54.3 ± 3.7	66.8 ± 2.9	+12.5
BioASQ	Bio	77.9 ± 0.4	79.1 ± 0.3	+1.2
RACE	Exam	59.8 ± 0.6	60.7 ± 0.5	+0.9
RE	Wiki	88.4 ± 0.1	88.8 ± 0.2	+0.4
DuoRC	Movie	68.9 ± 0.7	67.7 ± 1.1	-1.2
DROP	Wiki	68.9 ± 1.7	67.1 ± 1.9	-1.8

↑
ID
OOD
↑
Memog работаем с новыми б/cero
на OOD данных

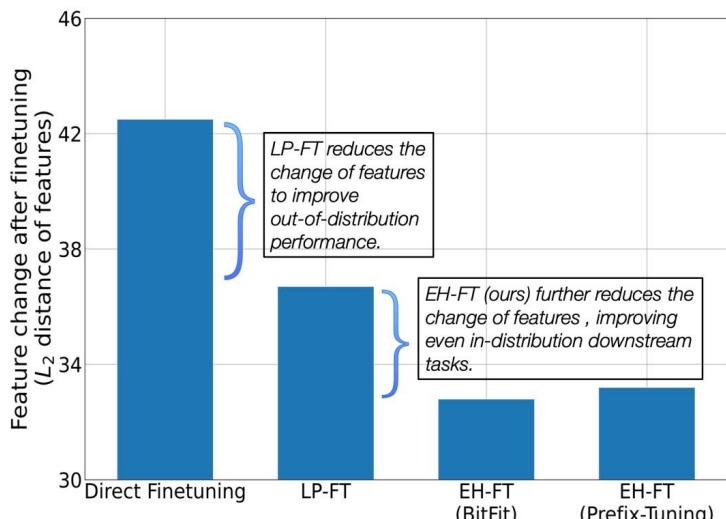


(a) Prompt length

4.5) PEFT + LP-FT (EH-FT)

1. Учим PEFT и сортируем от керо голову.
2. Инициализируем исходную модель этой головы и запускаем Fine-tuning.

- + Обработка всех по каскаду, потому что наша модель еще не обучена.
- Куда много памяти из-за FT.



Methods	RTE	BoolQ	COPA	CB	WIC	MRPC	QNLI	COLA	STS-B	Avg
Finetuning (Liu et al., 2019b)	86.60‡	86.90	94.00	98.20‡	75.60	90.90‡	94.70	68.00	92.40‡	87.48
Finetuning (reproduce)	87.52	86.32	93.75	94.64	73.90	90.81	94.75	68.72	92.27	86.96
Linear Probing	61.10	64.31	80.25	79.47	67.97	75.45	69.95	33.17	60.25	65.77
Prefix-Tuning	77.00	83.90	87.75	100	65.05	89.56	94.55	57.89	89.92	82.85
LoRA	88.50	86.29	94.75	100	73.04	90.43	94.37	67.86	92.17	87.49
BitFit	87.05	86.13	95.50	99.11	72.01	90.32	94.68	57.89	92.05	87.24
Top-K Tuning	86.55	85.34	93.00	98.66	73.71	90.94	94.12	65.78	91.47	86.62
Mixout	85.56	86.06	95.00	98.66	74.45	90.87	94.18	65.45	91.62	86.87
Child-Tuning _D	88.18	86.65	94.5	92.86	74.07	91.36	94.44	68.52	92.51	87.00
LP-FT	86.14	86.38	94.00	93.50	74.73	91.47	94.78	67.45	92.20	86.74
EH-FT _{LoRA}	88.68	86.69	94.5	99.12	74.65	91.00	94.73	69.00	92.24	87.85
EH-FT _{PT}	87.22	86.9	95.00	100	73.63	90.56	94.89	69.10	92.31	87.73
EH-FT _{BitFit}	88.10	86.97	94.75	99.12	75.20	91.00	94.61	68.78	92.25	87.86