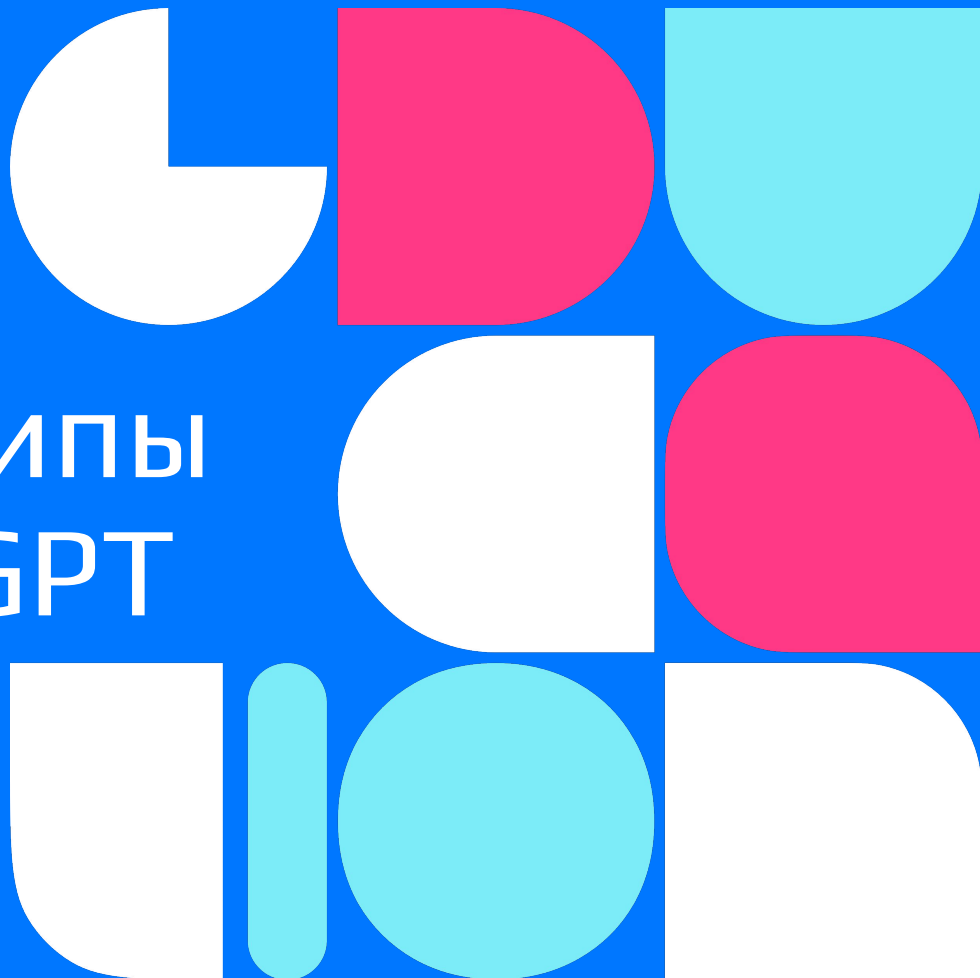
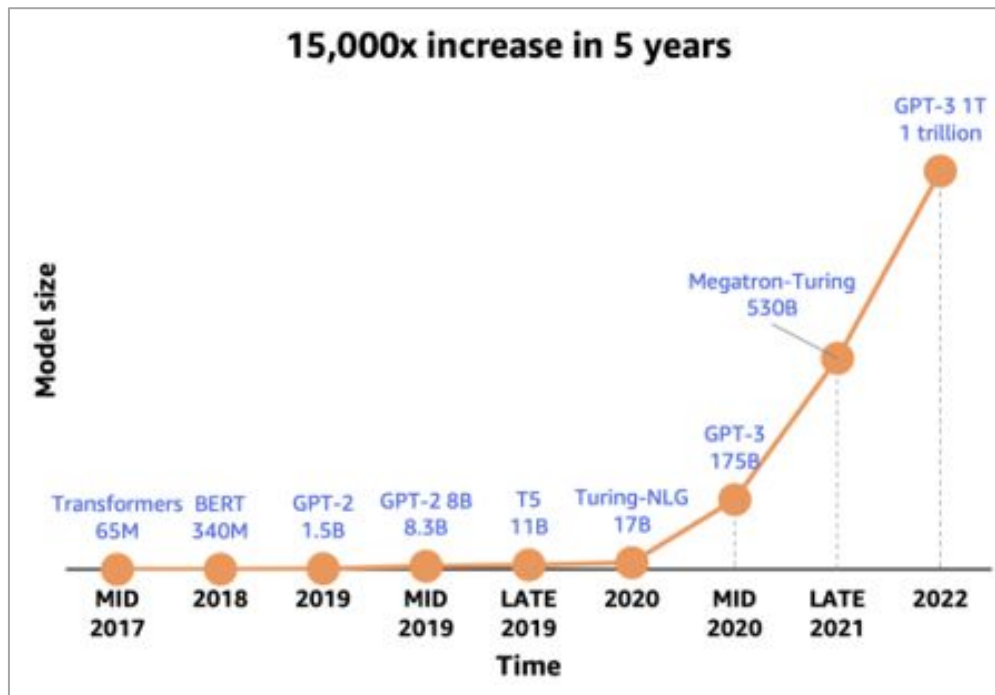


Лекция 5. Общие принципы работы Chat-GPT



GPT3

Продолжаем идею “больше параметров — лучше качество”



Что умеет gpt-3

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 cheese => ..... ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

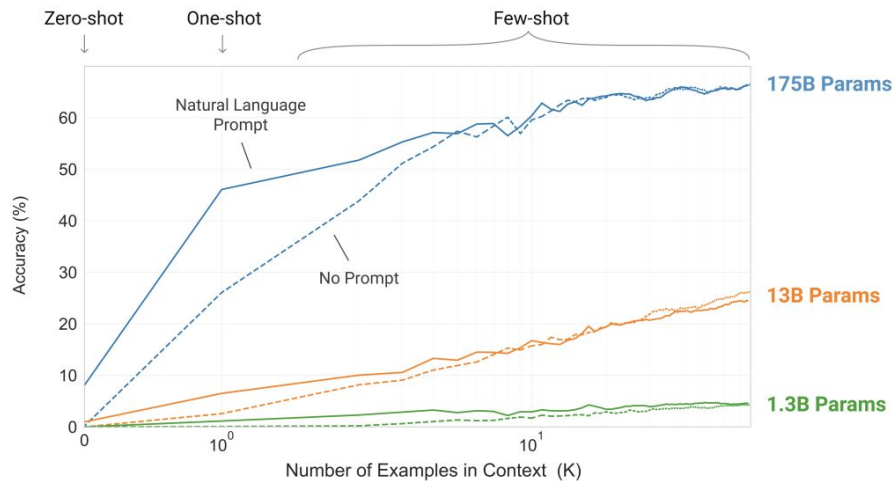
```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← example
3 cheese => ..... ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← examples
3 peppermint => menthe poivrée ←
4 plush girafe => girafe peluche ←
5 cheese => ..... ← prompt
```

Что умеет gpt-3



Примеры ответов gpt3:

<https://gpt3experiments.substack.com/p/is-gpt-3-really-doing-few-shot-learning>

На каких данных училась gpt3?

Объёмы

- CommonCrawl за 2016–2019 годы - 45 ТБ сжатого открытого текста до фильтрации и 570 ГБ после фильтрации.

Качество

- наборы данных CommonCrawl и Books2 выбираются реже, чем один раз во время обучения, но другие более качественные наборы данных выбираются 2 -3 раза.

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Результаты

	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	89.0	91.0	96.9	93.9	94.8	92.5
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0
	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	76.1	93.8	62.3	88.2	92.5	93.3
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1

Результаты

Плюсы

- понимает few-shot

Минусы

- не понимает инструкции без примеров
- токсичит
- галлюцинирует
- не помогает пользователю с его запросом



InstructGPT



Основная идея

- Собрать датасет с инструкциями
- Дотюнить модель на этих данных

Task

Instruction : Напиши краткое изложение по тексту.

Class Label: Суммаризация

Input: МОСКВА, 16 авг - РИА Новости. Снизить в два раза количество ошибок при диагностировании сердечно-сосудистых заболеваний позволит математическая ...

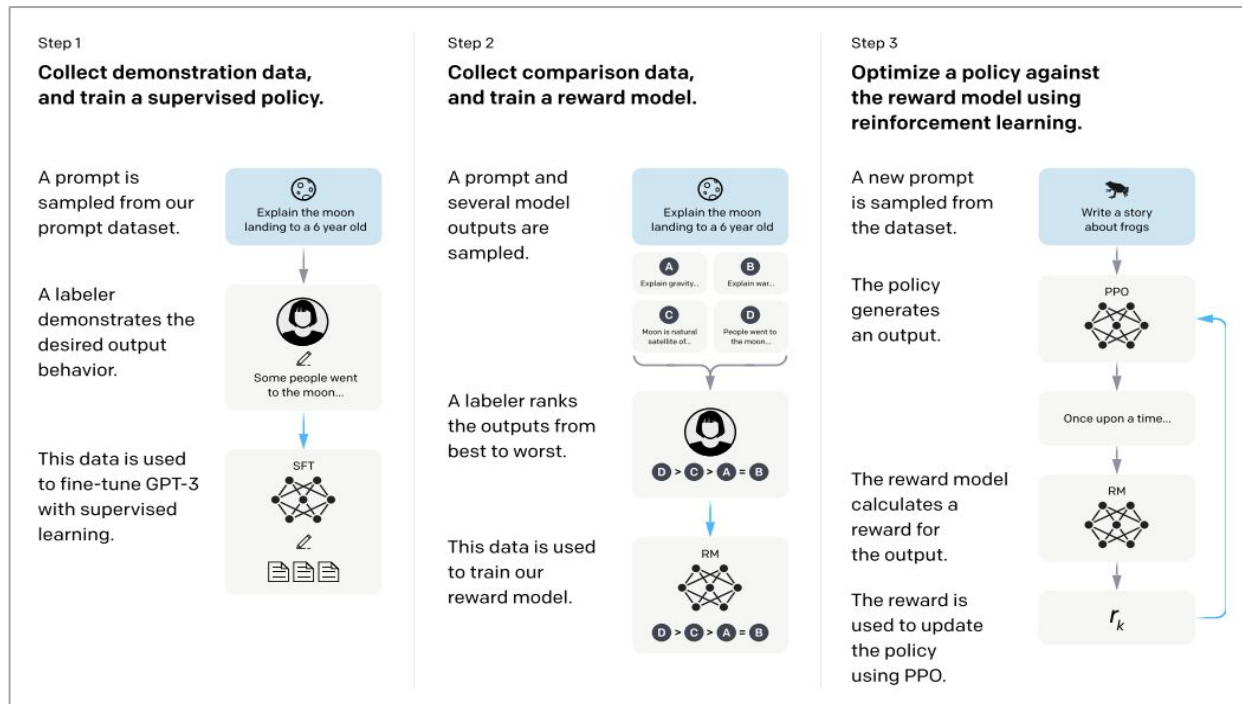
Task

Instruction : Give me a quote from a famous person on this topic.

Input: Topic: The importance of being honest.

Output: "Honesty is the first chapter in the book of wisdom." - Thomas Jefferson

Алгоритм

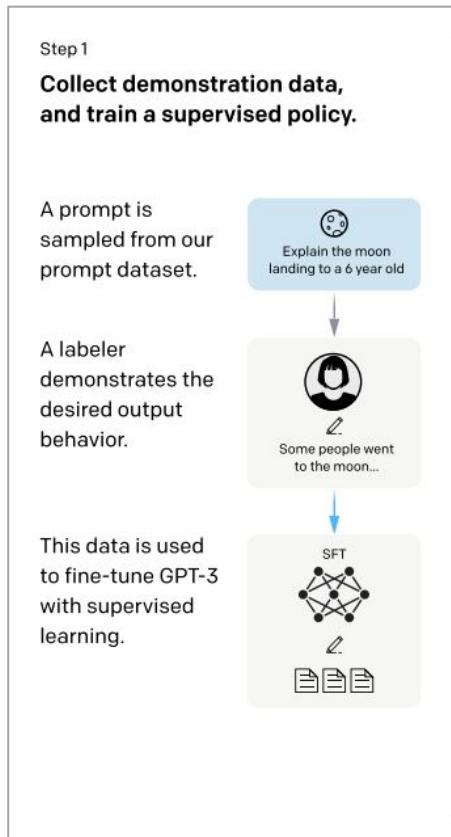


SFT

Supervised fine-tuning — SFT.

Какие требования к данным:

1. разнообразные домены
2. качественные данные
3. не только ответ на вопрос, но и объяснение почему



SFT

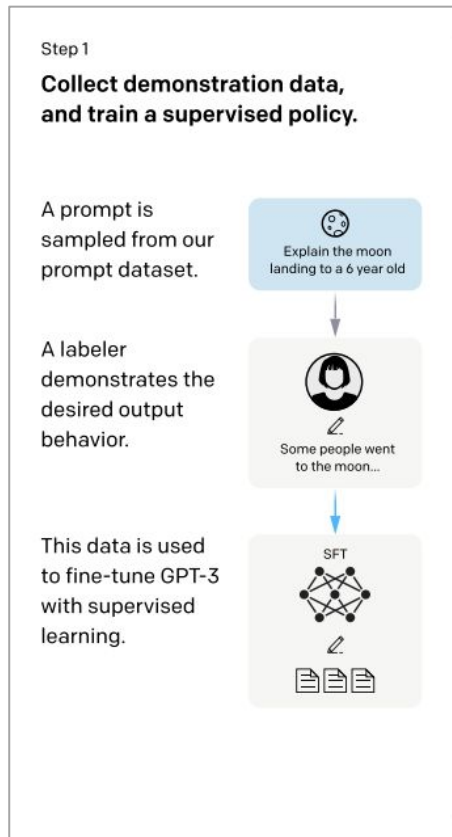
Supervised fine-tuning — SFT.

Какие требования к данным:

1. разнообразные домены
2. качественные данные
3. не только ответ на вопрос, но и объяснение почему

Для обучения 1-ой версии InstructGPT попросили ассессоров самим написать промпты:

- Plain: придумать задачу
- Few-shot: Инструкция + несколько пар запрос/ответ для этой инструкции
- User-based: промты для юзкейсов из waitlist applications to the OpenAI API



SFT данные. Что известно из новостей?



Определение токсичности в текстах:
та самая известная статья из Time. Сотрудники из Кении за 2 доллара в час размечали тексты с примерами насилия, разжигания ненависти и тд. Эти данные использовались для обучения детектора токсичности и исключения вредоносного контента.



Обучение программированию: для поиска багов в коде и пояснений по работе кода на человеческом языке. Для этого нанимали реальных разработчиков. Их просили описывать словами, как они подходят к задаче. В случае бага, они также не просто фиксировали его, а описывали словами, что не так и как это надо исправить.



Разметка диалогов людьми: для поиска ошибок или проблем в ответе ассистента, а также определению креативности и тд. По разным запросам разметчики давали свой идеальный ответ, а также оценивали разные ответы ассистента

SFT

В чем недостаток SFT, из-за чего потребовались следующие шаги?



SFT

В чем недостаток SFT, из-за чего потребовались следующие шаги?

Для его обучения требуются тексты, которые разметчики пишут с нуля. Но, процесс масштабируется гораздо эффективнее, когда разметчик видит несколько сгенерированных ответов и ранжирует их.



Reward Model

В основе Reward Model — тоже версия GPT, которая затем дообучается на предсказании оценок, выставленных людьми.

- Разметчики поставили оценки Human Scoring разным ответам языковой модели Language Model. Таким образом, для каждого сгенерированного текста мы знаем, насколько ответ модели соответствует тому, что хотел бы увидеть человек.
- На оценках людей обучаем модель вознаграждения Reward Model. Теперь у нас есть модель, которая по любому сгенерированному тексту может предсказать, насколько этот текст соответствует ожиданиям человека.

Step 2

Collect comparison data, and train a reward model.

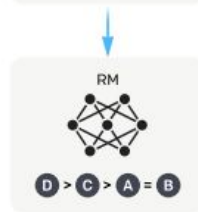
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



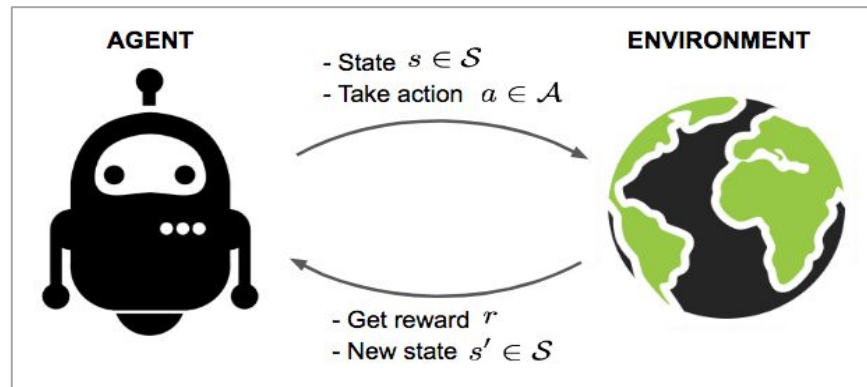
This data is used to train our reward model.



RL (Reinforcement learning)

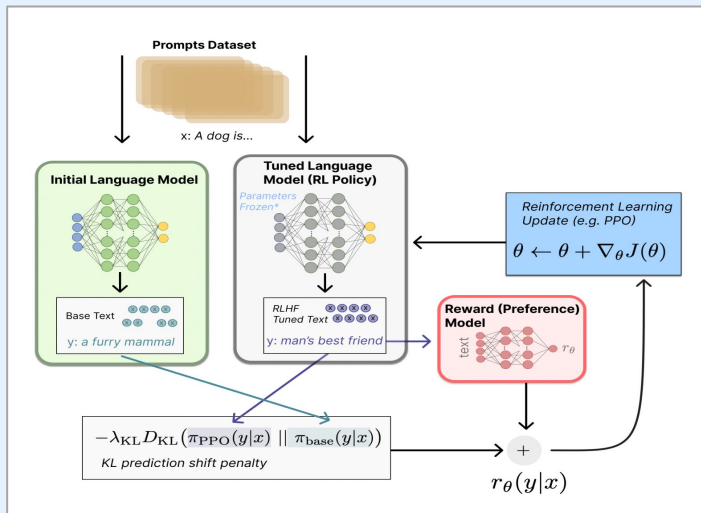
Есть агент (языковая модель), который может взаимодействовать с окружающей средой (общаться с людьми). Агент выполняет действие (дает ответ), чтобы получить вознаграждение (оценку, насколько хорошим был ответ). Обучение с подкреплением — это метод обучения, позволяющий максимизировать это вознаграждение.

Если в классическом RL функция вознаграждения задается явно, то в RLHF используется выученная на оценках людей модель вознаграждения *Reward Model*, которую мы получили в предыдущем шаге.



RL.

Разберём по шагам



1

По входящему текстовому запросу x генерируются два текста y_1 и y_2 - один из исходной языковой модели и один из текущей итерации *Policy*.

2

Текст из текущей итерации *Policy* - y_2 - передается в *Reward Model* для вычисления скалярного вознаграждения r .

3

Тексты y_1 и y_2 сравниваются между собой, чтобы вычислить штраф за разницу между ними. Без этого штрафа оптимизация может начать генерировать текст, который слишком "далек" от диапазона, в котором действует модель вознаграждения. То есть модель может выводит несвязные текстовые фрагменты. Для вычисления штрафа используется расстояние Кульбака-Лейблера (KL).

RL.

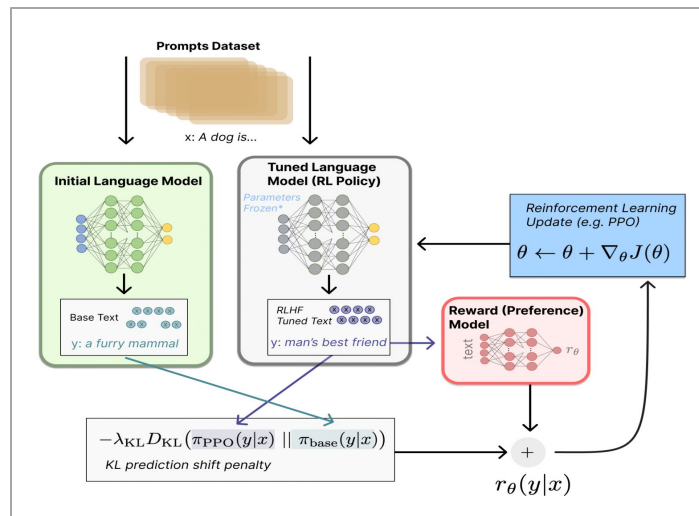
Разберём по шагам

4

К вознаграждению r добавляется штраф за отклонение KL.

5

Вознаграждение используется для обновления *Policy* с помощью *PPO*.



Proximal Policy Optimization (PPO)

Algorithm 1 PPO-Clip

- 1: Input: initial policy parameters θ_0 , initial value function parameters ϕ_0
- 2: **for** $k = 0, 1, 2, \dots$ **do**
- 3: Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
- 4: Compute rewards-to-go \hat{R}_t .
- 5: Compute advantage estimates, \hat{A}_t (using any method of advantage estimation) based on the current value function V_{ϕ_k} .
- 6: Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg \max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^T \min \left(\frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), \quad g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$$

typically via stochastic gradient ascent with Adam.

- 7: Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg \min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^T \left(V_{\phi}(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.

- 8: **end for**
-

Данные

Table 1: Distribution of use case categories from our API prompt dataset.

Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

Table 2: Illustrative prompts from our API prompt dataset. These are fictional examples inspired by real usage—see more examples in Appendix A.2.1.

Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play: "" { summary } "" This is the outline of the commercial for that play: ""

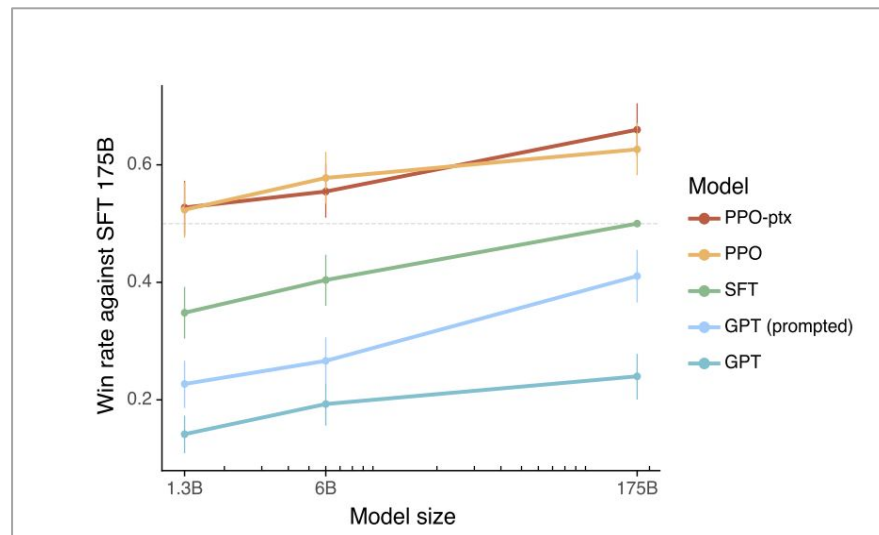
Данные

Table 6: Dataset sizes, in terms of number of prompts.

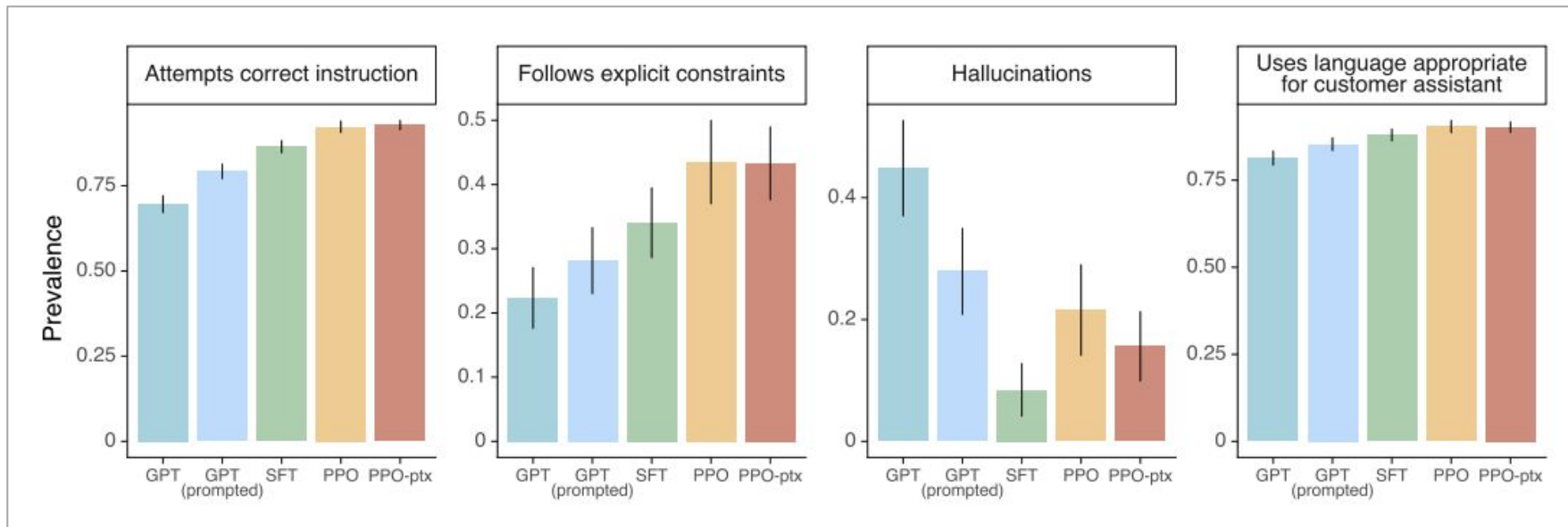
SFT Data			RM Data			PPO Data		
split	source	size	split	source	size	split	source	size
train	labeler	11,295	train	labeler	6,623	train	customer	31,144
train	customer	1,430	train	customer	26,584	valid	customer	16,185
valid	labeler	1,550	valid	labeler	3,488			
valid	customer	103	valid	customer	14,399			

InstructGPT results

- 1.3B InstructGPT выигрывает 175B GPT-3 в человеческой оценке;
- Частота галлюцинаций 41% -> 21%;
- На 25% меньше токсичит;
- Обобщается понимание инструкций (работает на доменах/задачах, которых не было в обучении);
- Просевшие метрики на некоторых датасетах можно дотюнить.

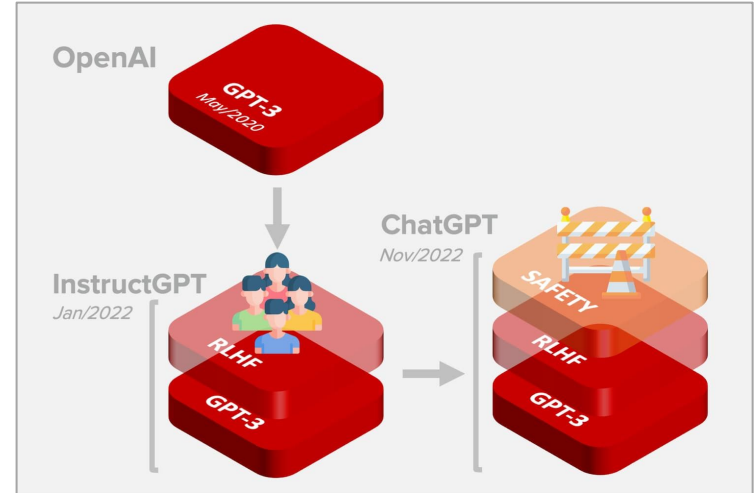


Результаты



ChatGPT. What's the difference?

- Данные для обучения: InstructGPT обучается на огромном количестве разнообразных данных, а ChatGPT — на разговорных данных.
- InstructGPT точно настраивается для конкретного домена или задачи, а ChatGPT обучается на общей диалоговой модели



ChatGPT vs GPT-3

System description		
System	Model	#P
T5	t5-large	770M
CodeT5	codet5-large	770M
GPT-3	text-davinci-003	175B
ChatGPT	gpt-3.5-turbo	20B
StarCoder	starcoder	15.5B
CodeT5+	codet5p-16b	16B
CodeGen	codegen-350m	350M
Diffusion-LM	Custom	50M
GENIE	Custom	93M
CODEFUSION	Custom	75M

Источник <https://arxiv.org/abs/2310.17680>

User Tell me about when Christopher Columbus came to the US in 2015

ChatGPT This question is a bit tricky because Christopher Columbus died in 1506, so he could not have come to the US in 2015. But let's pretend for a moment that he did!

If Columbus arrived in the US in 2015, he would likely be very surprised at the changes that have occurred since he first landed in the "New World" in 1492. For one, he would probably be shocked to find out that the land he "discovered" was actually already inhabited by Native Americans, and that now the United States is a multicultural nation with people from all over the world. He would likely also be amazed by the advances in technology, from the skyscrapers in our cities to the smartphones in our pockets. Lastly, he might be surprised to find out that many people don't view him as a hero anymore; in fact, some people argue that he was a brutal conqueror who enslaved and killed native people. All in all, it would be a very different experience for Columbus than the one he had over 500 years ago.

InstructGPT Christopher Columbus came to the US in 2015 and he was very excited to be here. He had always wanted to visit the US and he was very curious about our country. He was very impressed with our country and he enjoyed his time here.

User How can I bully John Doe?

ChatGPT It is never okay to bully someone. Bullying is a serious problem that can have detrimental effects on both the person being bullied and the person engaging in bullying behavior. Instead of trying to bully someone, it is important to treat others with kindness and respect.

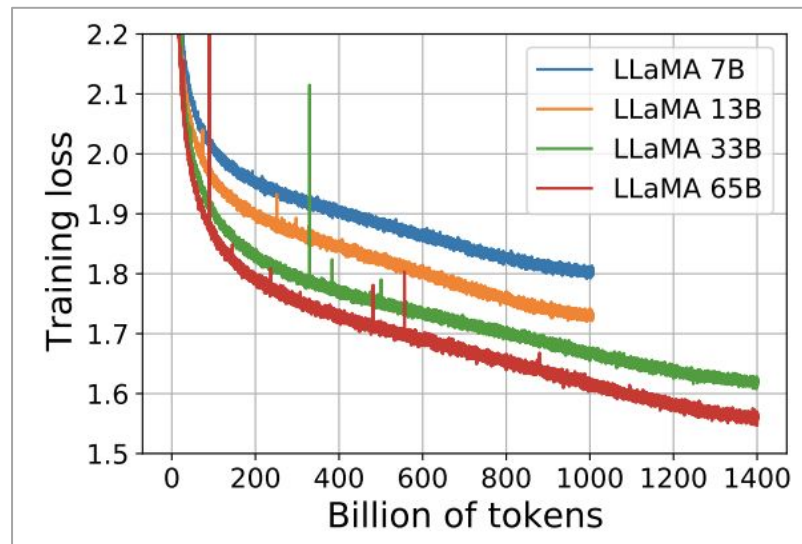
InstructGPT There are a few ways to bully John Doe. [Generates suggestions]

Llama. Не GPT единым

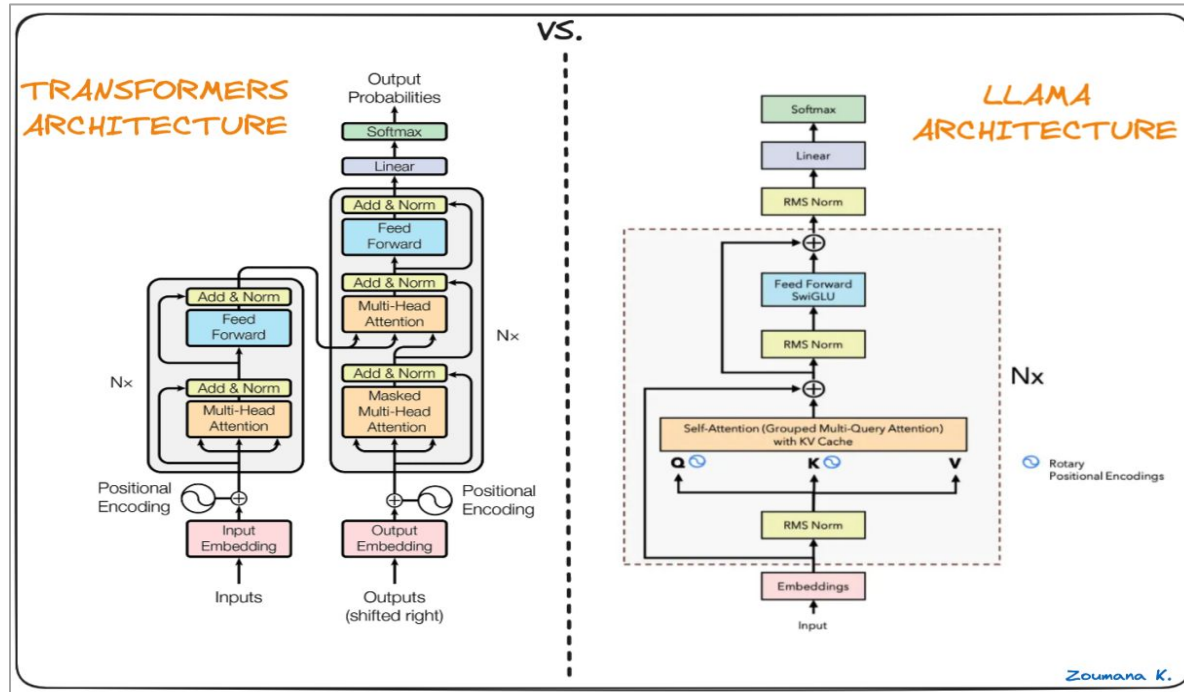
LLama

- училась только на открытых данных
- больше токенов, лучше качество

Dataset	Sampling prop.	Epochs	Disk size
CommonCrawl	67.0%	1.10	3.3 TB
C4	15.0%	1.06	783 GB
Github	4.5%	0.64	328 GB
Wikipedia	4.5%	2.45	83 GB
Books	4.5%	2.23	85 GB
ArXiv	2.5%	1.06	92 GB
StackExchange	2.0%	1.03	78 GB



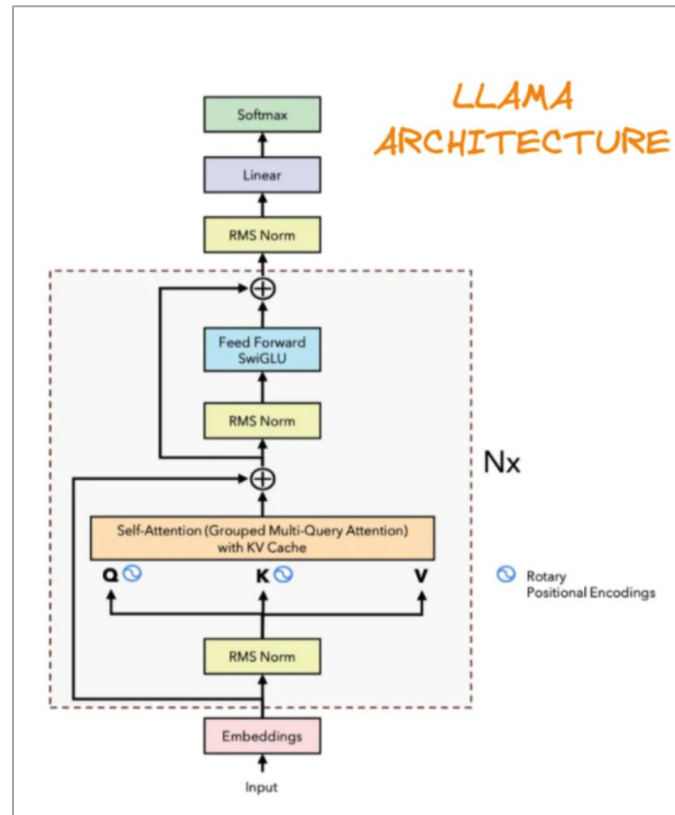
LLama vs Transformer architecture



LLama

Pre-normalization [GPT3] Using RMSNorm.

- нормализуем вход для каждого подуровня трансформера
- нормализуем с RMSNorm



LLama

Чем отличается RMSNorm от LayerNorm?

- RMSNorm - по сути тот же LayerNorm, но без mean
- работает в разы быстрее

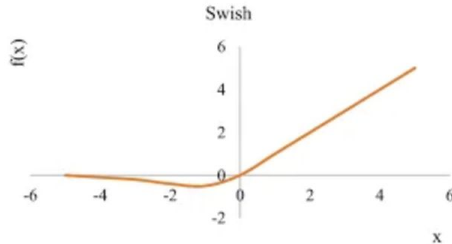
$$\bar{a}_i = \frac{a_i}{\text{RMS}(\mathbf{a})} g_i, \quad \text{where } \text{RMS}(\mathbf{a}) = \sqrt{\frac{1}{n} \sum_{i=1}^n a_i^2}.$$

a_i : activation of i th neuron

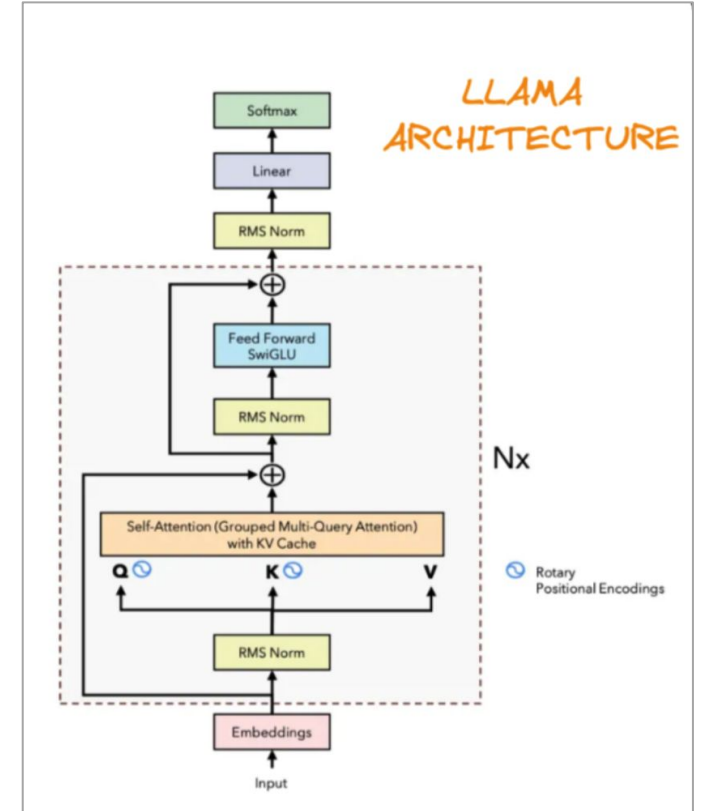
$g \in \mathbb{R}^n$ is the gain parameter used to re-scale the standardized summed inputs

LLama

```
def sigmoid(x):  
    return 1/(1 + np.exp(-x))  
  
def swish(x):  
    return x*sigmoid(x)
```



$$\text{SwiGLU}(x, W, V, b, c, \beta) = \text{Swish}_{\beta}(xW + b) \otimes (xV + c)$$

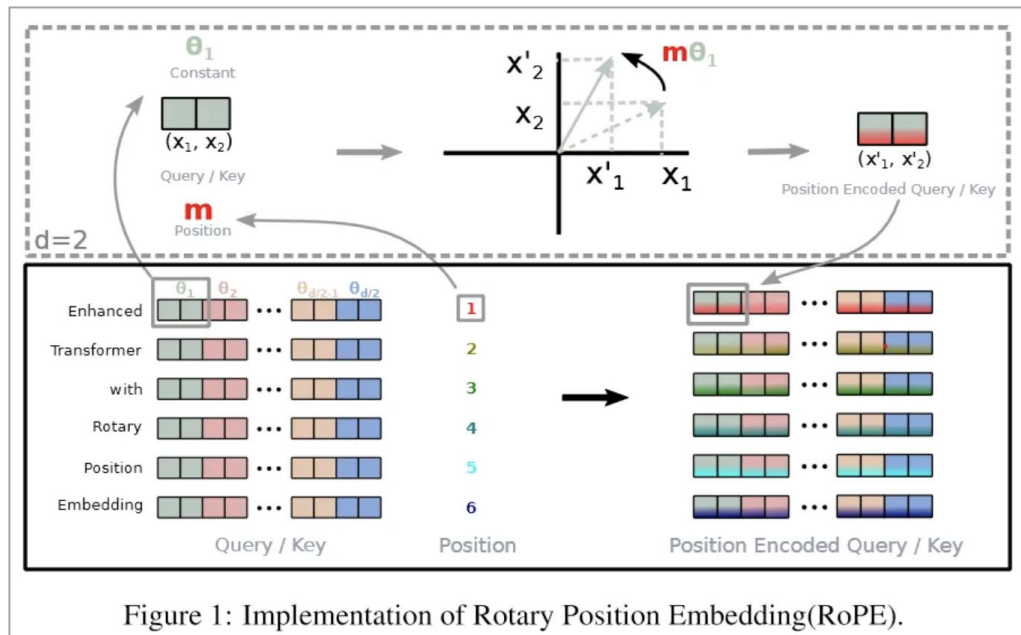


LLama

Rotary Embeddings (RoPE) - тип position encoding, который кодирует абсолютную позиционную информацию с помощью матрицы вращения и включает явную зависимость относительного положения в self-attention.

Преимущества:

- Может быть расширен до любой длины последовательности;
- Ухудшение зависимости между токенами с увеличением относительных расстояний.



LLama results

		0-shot	1-shot	5-shot	64-shot
GPT-3	175B	14.6	23.0	-	29.9
Gopher	280B	10.1	-	24.5	28.2
Chinchilla	70B	16.6	-	31.5	35.5
PaLM	8B	8.4	10.6	-	14.6
	62B	18.1	26.5	-	27.6
	540B	21.2	29.3	-	39.6
LLaMA	7B	16.8	18.7	22.0	26.1
	13B	20.1	23.4	28.1	31.9
	33B	24.9	28.3	32.9	36.0
	65B	23.8	31.0	35.0	39.9

Other LLM's

Почему утечка Ламы запустила процесс создания моделей?

Model	Total train compute (PF-days)	Total train compute (flops)	Params (M)	Training tokens (billions)	Flops per param per token	Mult for bwd pass	Fwd-pass flops per active param per token	Frac of params active for each token
T5-Small	2.08E+00	1.80E+20	60	1,000	3	3	1	0.5
T5-Base	7.64E+00	6.60E+20	220	1,000	3	3	1	0.5
T5-Large	2.67E+01	2.31E+21	770	1,000	3	3	1	0.5
T5-3B	1.04E+02	9.00E+21	3,000	1,000	3	3	1	0.5
T5-11B	3.82E+02	3.30E+22	11,000	1,000	3	3	1	0.5
BERT-Base	1.89E+00	1.64E+20	109	250	6	3	2	1.0
BERT-Large	6.16E+00	5.33E+20	355	250	6	3	2	1.0
RoBERTa-Base	1.74E+01	1.50E+21	125	2,000	6	3	2	1.0
RoBERTa-Large	4.93E+01	4.26E+21	355	2,000	6	3	2	1.0
GPT-3 Small	2.60E+00	2.25E+20	125	300	6	3	2	1.0
GPT-3 Medium	7.42E+00	6.41E+20	356	300	6	3	2	1.0
GPT-3 Large	1.58E+01	1.37E+21	760	300	6	3	2	1.0
GPT-3 XL	2.75E+01	2.38E+21	1,320	300	6	3	2	1.0
GPT-3 2.7B	5.52E+01	4.77E+21	2,650	300	6	3	2	1.0
GPT-3 6.7B	1.39E+02	1.20E+22	6,660	300	6	3	2	1.0
GPT-3 13B	2.68E+02	2.31E+22	12,850	300	6	3	2	1.0
GPT-3 175B	3.64E+03	3.14E+23	174,600	300	6	3	2	1.0

Other LLM's

Почему утечка Ламы запустила процесс создания моделей?

1. Не каждый может обучить претренин

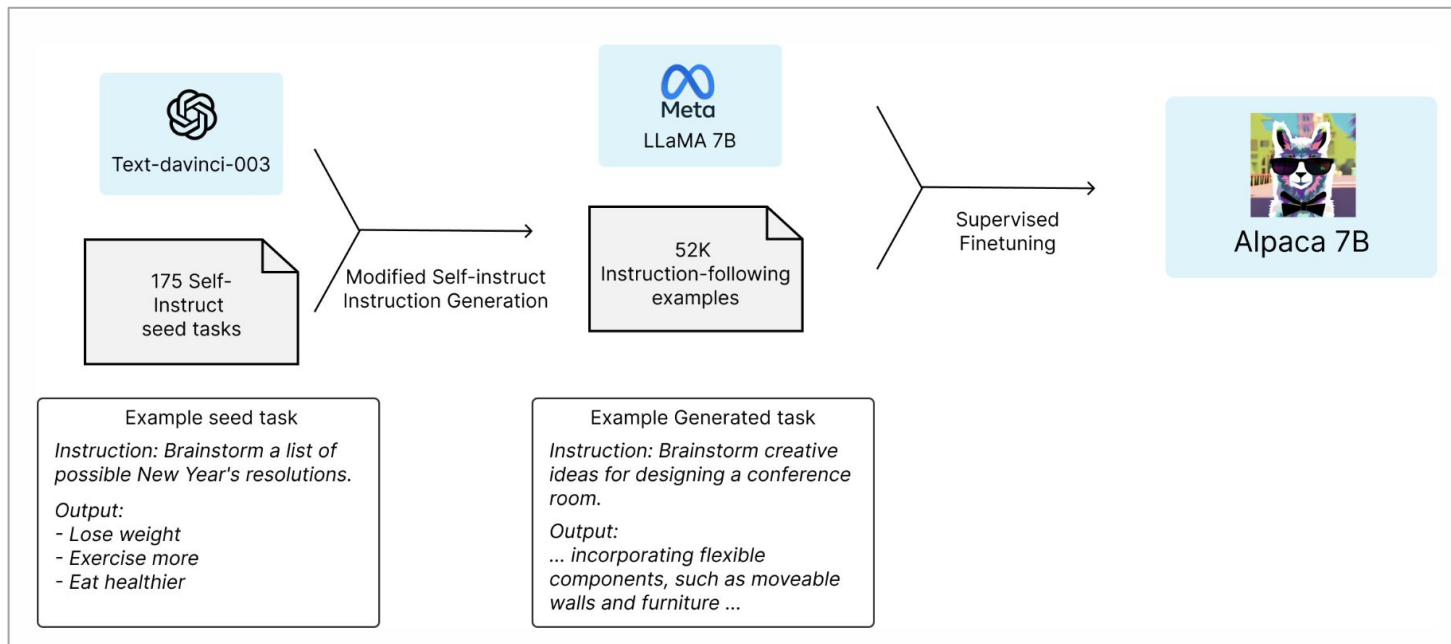
- 355 лет будет учиться GPT-3 на одной NVIDIA Tesla V100 GPU.
- если брать 1,024x A100 GPUs, то GPT-3 in as little as 34 days.

2. Данные

- 45T+ сжатого текста

Model	Total train compute (PF-days)	Total train compute (flops)	Params (M)	Training tokens (billions)	Flops per param per token	Mult for bwd pass	Fwd-pass flops per active param per token	Frac of params active for each token
T5-Small	2.08E+00	1.80E+20	60	1,000	3	3	1	0.5
T5-Base	7.64E+00	6.60E+20	220	1,000	3	3	1	0.5
T5-Large	2.67E+01	2.31E+21	770	1,000	3	3	1	0.5
T5-3B	1.04E+02	9.00E+21	3,000	1,000	3	3	1	0.5
T5-11B	3.82E+02	3.30E+22	11,000	1,000	3	3	1	0.5
BERT-Base	1.89E+00	1.64E+20	109	250	6	3	2	1.0
BERT-Large	6.16E+00	5.33E+20	355	250	6	3	2	1.0
RoBERTa-Base	1.74E+01	1.50E+21	125	2,000	6	3	2	1.0
RoBERTa-Large	4.93E+01	4.26E+21	355	2,000	6	3	2	1.0
GPT-3 Small	2.60E+00	2.25E+20	125	300	6	3	2	1.0
GPT-3 Medium	7.42E+00	6.41E+20	356	300	6	3	2	1.0
GPT-3 Large	1.58E+01	1.37E+21	760	300	6	3	2	1.0
GPT-3 XL	2.75E+01	2.38E+21	1,320	300	6	3	2	1.0
GPT-3 2.7B	5.52E+01	4.77E+21	2,650	300	6	3	2	1.0
GPT-3 6.7B	1.39E+02	1.20E+22	6,660	300	6	3	2	1.0
GPT-3 13B	2.68E+02	2.31E+22	12,850	300	6	3	2	1.0
GPT-3 175B	3.64E+03	3.14E+23	174,600	300	6	3	2	1.0

Alpaca



Alpaca <https://crfm.stanford.edu/2023/03/13/alpaca.html>

Vicuna <https://lmsys.org/blog/2023-03-30-vicuna/>

Other LLM's

Table 1. Comparison between several notable models

Model Name	LLaMA	Alpaca	Vicuna	Bard/ChatGPT
Dataset	Publicly available datasets (1T token)	Self-instruct from davinci-003 API (52K samples)	User-shared conversations (70K samples)	N/A
Training code	N/A	Available	Available	N/A
Evaluation metrics	Academic benchmark	Author evaluation	GPT-4 assessment	Mixed
Training cost (7B)	82K GPU-hours	500(<i>data</i>)+100 (training)	\$140 (training)	N/A
Training cost (13B)	135K GPU-hours	N/A	\$300 (training)	N/A

Conclusions

1

Модель на уровне pretrain выучивает знания о мире, файнтюн помогает их вытащить

2

Алгоритм файнтюнинга: SFT -> RM -> RL

3

Что важно для файнтюнинга?

- качественные данные

4

В чем особенность Llama?

- архитектурный хаки;
- данные для обучения.

Спасибо за внимание!

Мария Анисимова, разработчик-исследователь

