

ИСП

РАН

Большие языковые модели (LLM/БЯМ)

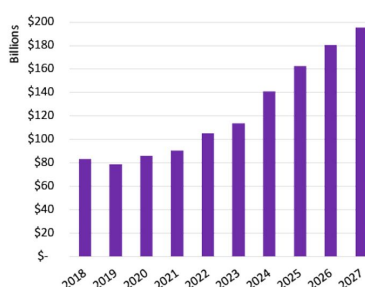
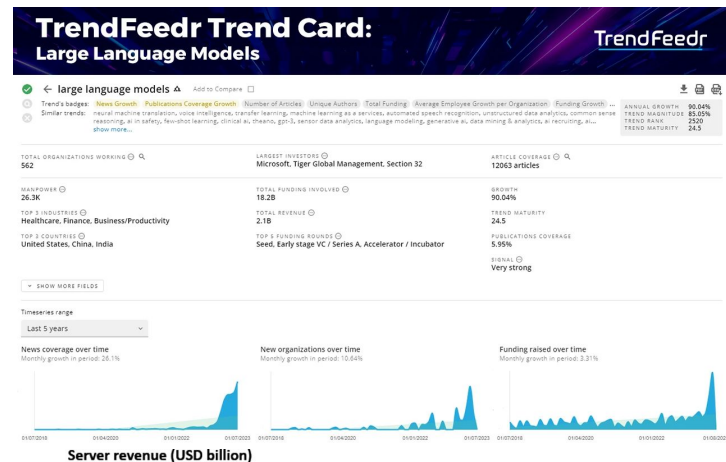
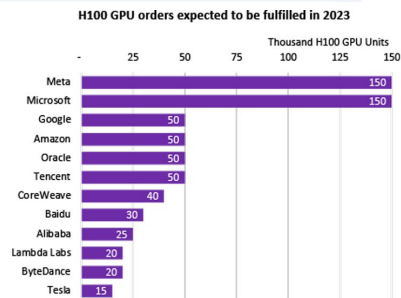
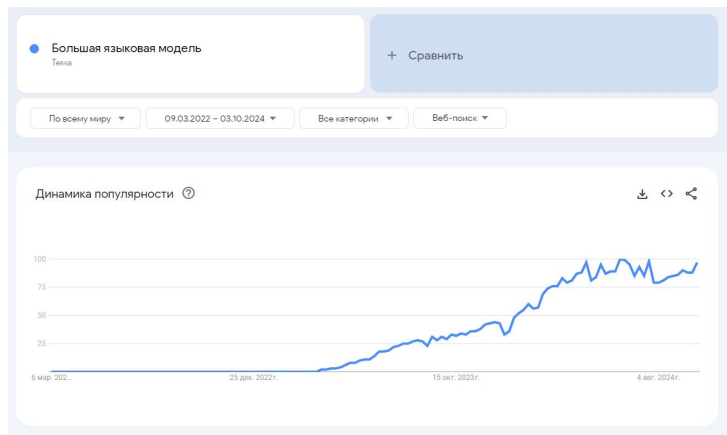
к.ф.-м.н. Тихомиров Михаил Михайлович

старший научный сотрудник
НЛС МГУ имени М. В. Ломоносова

Рост популярности LLM в мире



Московский
государственный
университет
имени М.В. Ломоносова



Рост популярности LLM в мире



Оценка стоимости LLM компаний инвесторами

- Mistral - 5.8 миллиарда
- XAI - 24 миллиарда
- Anthropic - 40 миллиардов
- OpenAI - 157 миллиардов

Для сравнения (market cap):

- Siemens ~ 150 миллиардов
- Nvidia ~ 3 триллиона (рост **x10** за 4 года)



Языковые модели оценивают вероятность различных текстовых сущностей: символов, слов, последовательностей слов.

- Первым человеком в космосе был ____ ?
- Что правдоподобнее:
 - я съел жареный гвоздь vs я съел жареный стейк

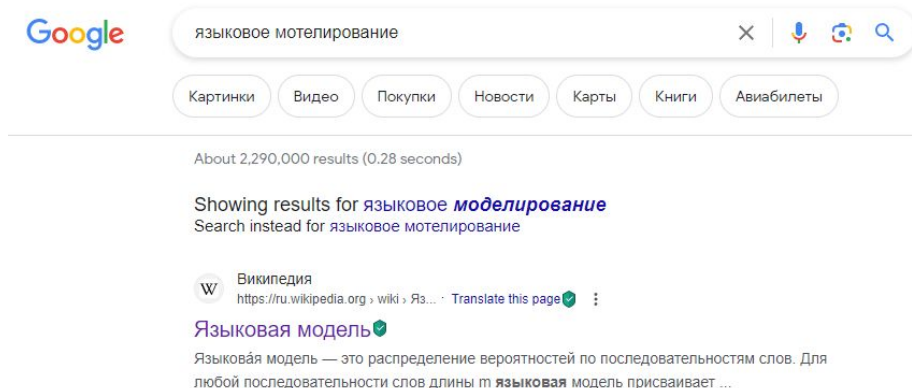
Где полезно языковое моделирование



Московский
государственный
университет
имени М.В. Ломоносова

Все мы регулярно сталкиваемся с языковым моделированием:

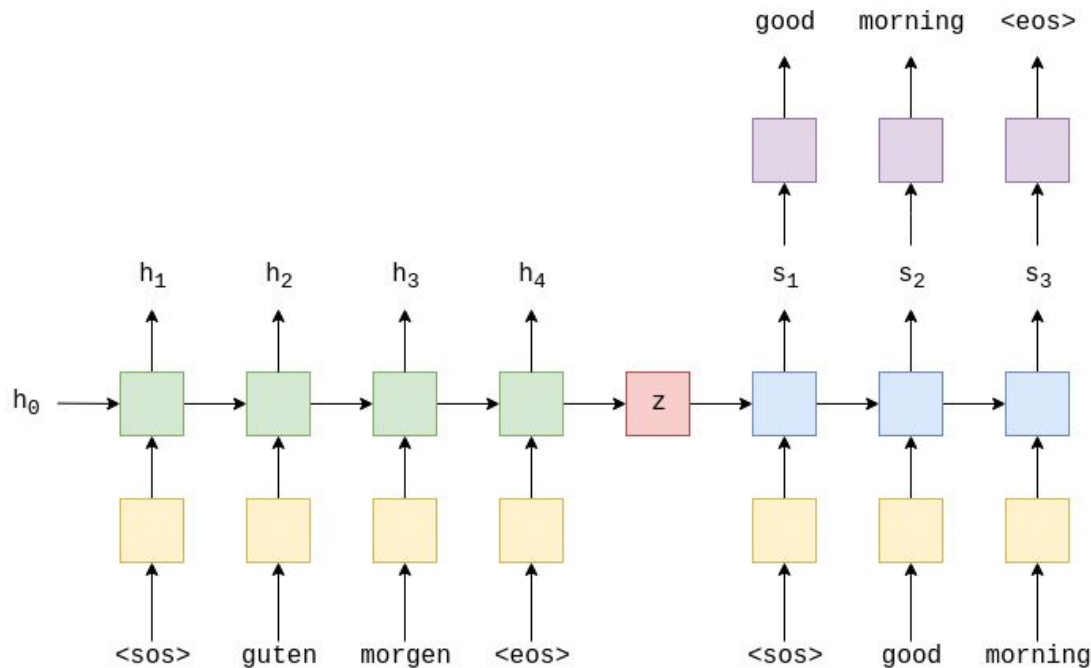
- Автодополнение на клавиатурах телефонов.
- Подсказки в поисковых системах.
- Исправление ошибок в поисковых системах.
- Распознавание речи и др.



Seq2Seq до трансформеров



- Вектор финального состояния должен хранить **всю** информацию из предложения
- По сути является векторным представлением (эмбеддингом) предложения
- Теряет информацию на длинных последовательностях



Механизм внимания (2014)



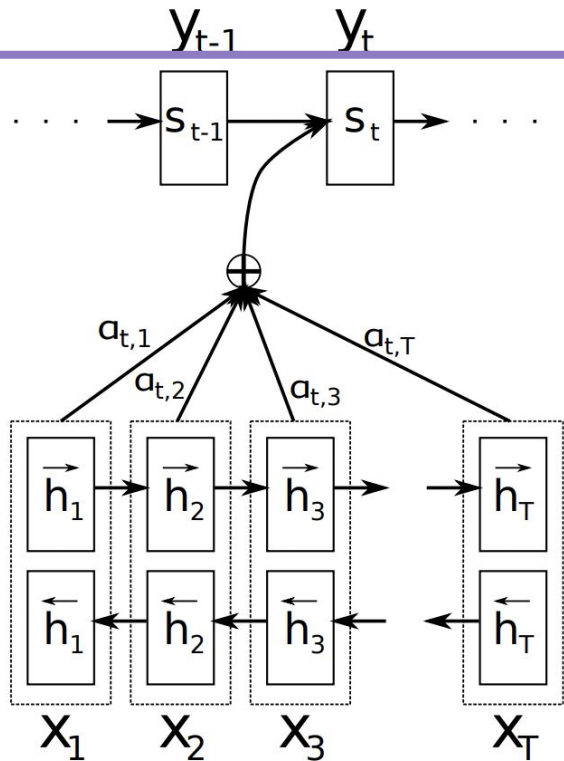
Автокодирующая модель состоит из:

- **Encoder(text) -> vector**: переводит текст в необходимое векторное представление
- **Decoder(vector) -> text**: расшифровывает представление в ответ модели

Проблема: в vector помещается только общий контекст

Решение: сохранять векторы для каждого слова и подбирать нужные под каждый шаг decoder

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$
$$e_{ij} = a(s_{i-1}, h_j)$$

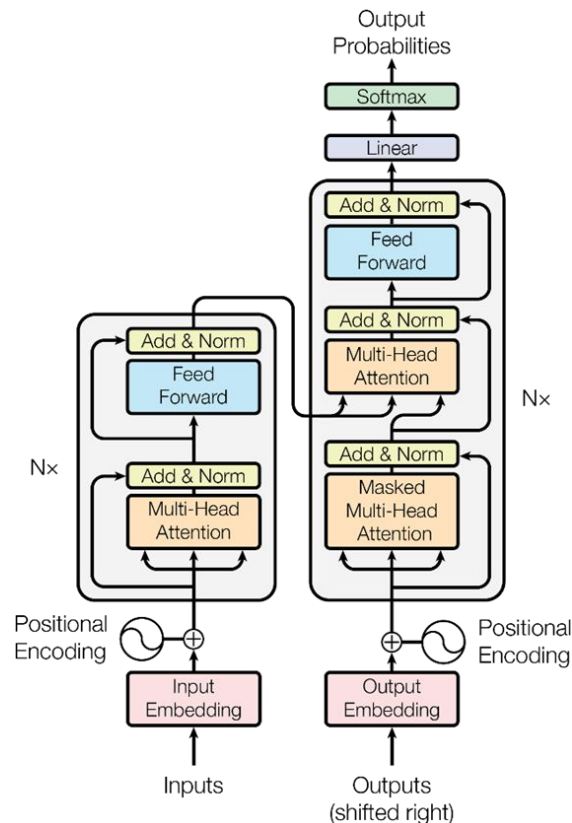
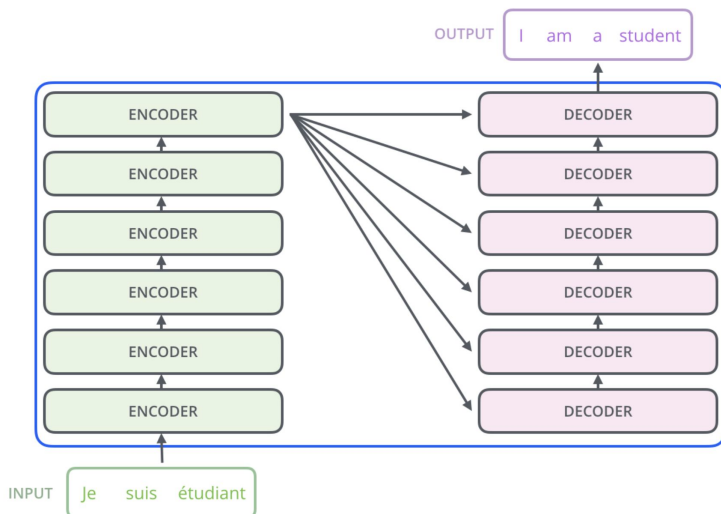


Transformer (2017)



Исходно **encoder-decoder** архитектура.

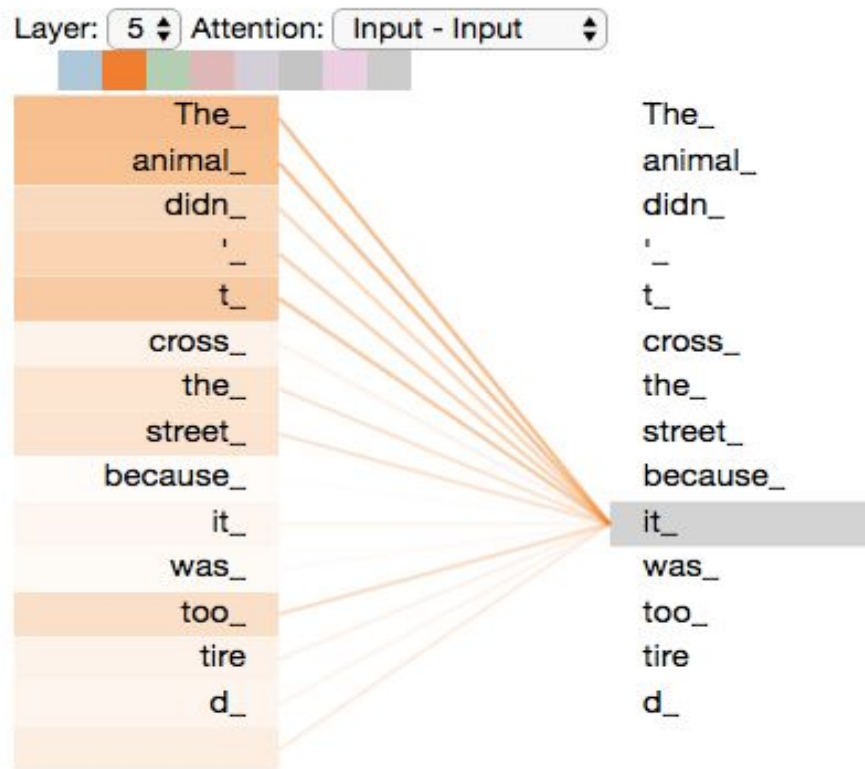
Каждый блок **одинаков** и последовательно преобразует входной вектор в выходной вектор той же размерности.



Визуализация Self Attention



- The **animal** didn't cross the **street** because **it** was too tired”
- К чему относится it: animal или street



OpenAI GPT-1 (2018)

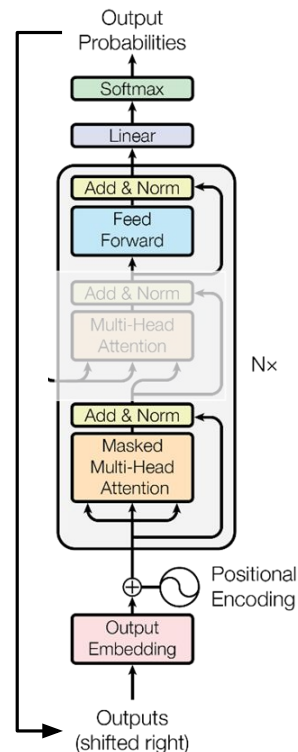
- 12 слоев **Transformer decoder** (~117 млн.),
- Обучение в 2 этапа:

- Предобучение (pre-training) на задаче **моделирования языка**

$$\max_{\Theta} \sum_{0 \leq i \leq n} \log P(w_i | w_{i-1} \dots w_0; \Theta)$$

w - слова последовательности, Θ - параметры модели

- Дообучение (fine-tuning) на целевые задачи
- Предобучался только на художественной литературе



OpenAI GPT-1: оценка качества



Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	<u>89.3</u>	-	-	-
CAFE [58] (5x)	80.2	79.0	<u>89.3</u>	-	-	-
Stochastic Answer Network [35] (3x)	<u>80.6</u>	<u>80.1</u>	-	-	-	-
CAFE [58]	78.7	77.9	88.5	<u>83.3</u>		
GenSen [64]	71.4	71.3	-	-	<u>82.3</u>	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	61.7
Finetuned Transformer LM (ours)	82.1	81.4	89.9	88.3	88.1	56.0

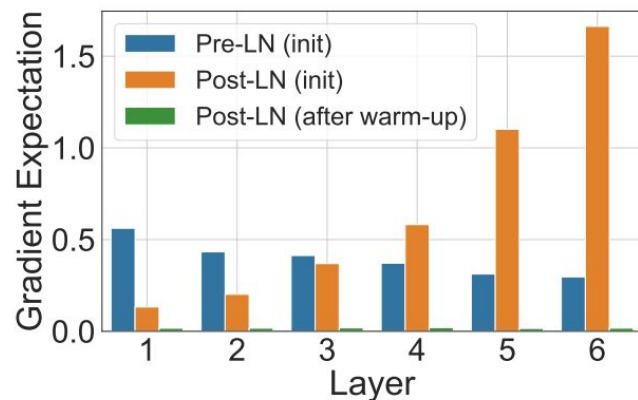
Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	<u>77.6</u>	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	<u>60.2</u>	<u>50.3</u>	<u>53.3</u>
Finetuned Transformer LM (ours)	86.5	62.9	57.4	59.0

GPT-2 (2019) – универсальный генератор текстов



Московский
государственный
университет
имени М.В. Ломоносова

- **Улучшенная архитектура:**
предварительная нормализация (**Pre-LN**)
входных данных для стабилизации градиентов
- **Больше параметров:**
в 4 раза больше слоев (**1.5 млрд параметров**)
– больше потенциальных знаний (capacity)
- **Новая парадигма:** любой текст содержит **подсказки к генерации (prompt)** и обучаясь на большом наборе текстов модель учится их понимать



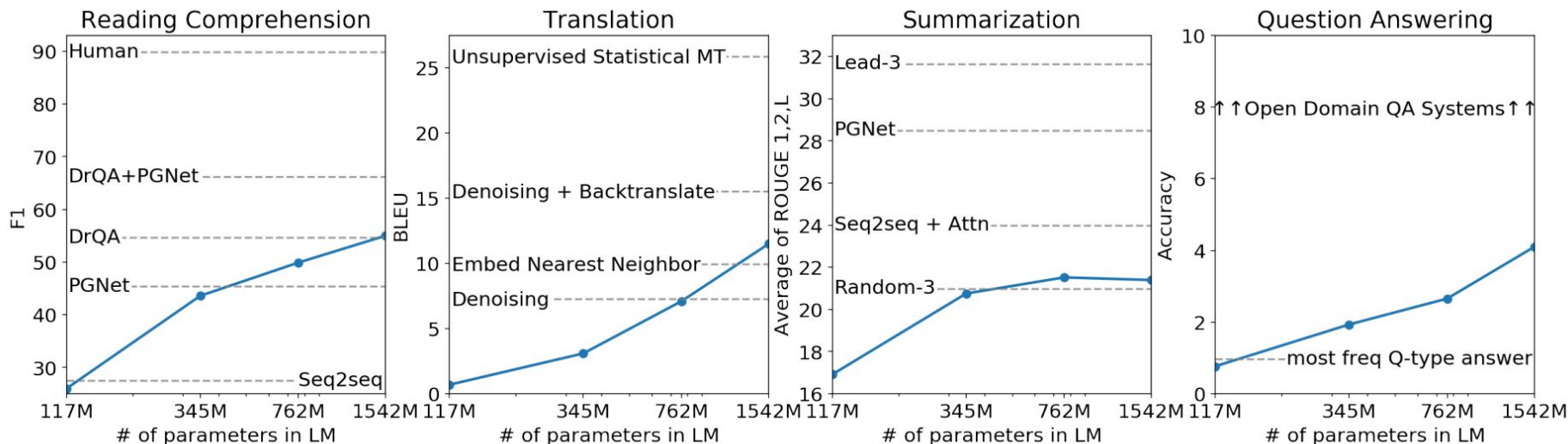
GPT-2: датасет для предобучения



- Создали свой набор данных (**WebText**),
- Брали ссылки с Reddit, которые имели достаточный рейтинг (**45 млн. ссылок**),
- Дополнительная очистка на основе ряда эвристик, дедубликация,
- Намеренно удалили Wikipedia ссылки из **WebText**,
- Итого **40GB** текстов или **8 млн.** документов.

Одна из основ успеха модели - **качественный** набор данных для предобучения!

GPT-2: zero-shot возможности



GPT-2: zero-shot возможности



	R-1	R-2	R-L	R-AVG
Bottom-Up Sum	41.22	18.68	38.34	32.75
Lede-3	40.38	17.66	36.62	31.55
Seq2Seq + Attn	31.33	11.81	28.83	23.99
GPT-2 _{TL;DR} :	29.34	8.27	26.58	21.40
Random-3	28.78	8.63	25.52	20.98
GPT-2 no hint	21.58	4.03	19.47	15.03

Table 4. Summarization performance as measured by ROUGE F1 metrics on the CNN and Daily Mail dataset. Bottom-Up Sum is the SOTA model from (Gehrmann et al., 2018)

Добавление **TL;DR**: повышает ROUGE-1 с 21.5 до 29.3 (подсказки к генерации!)

Пример порождения новостей (перевод)



Подводка: *Шокирующая находка: ученый обнаружил стадо единорогов, живущих в отдаленной, ранее неизведанной долине, в Андах. Еще более удивительным для исследователей было то, что единороги прекрасно говорили по-английски.*

GPT-2: Доктор Хорхе Перес, биолог-эволюционист из Университета Ла-Паса, и несколько его товарищей **исследовали Анды**, когда **обнаружили небольшую долину**, где не было других животных или людей. Перес заметил, что в долине есть что-то похожее на естественный фонтан, окруженный двумя пиками скалы и серебристым снегом.

Перес и остальные отправились дальше в долину. ... Перес и его друзья были поражены, **увидев стадо единорогов**. Этих существ можно было увидеть с воздуха — они были так близко, что могли коснуться своими рогами.

Изучая этих причудливых существ, ученые обнаружили, что **существа также говорили на довольно обычном английском языке...**

GPT-3 (2020) – первая коммерческая модель



- **Ориентация на рынок:** модель как облачный сервис
- **175 млрд параметров:** 96 слоев Transformer-decoder
- **Оптимизация потребления памяти:** половина слоев внимания используют разреженные матрицы (локальные окна)
- **Развитие парадигмы подводок (prompt):**
“обучение в контексте” (in-context learning)
- **Обучение на доверенных данных:** примеры для обучения смешиваются пропорционально их качеству (согласно экспертам)
- **В 15 раз больше данных:** добавлена очищенная коллекция CommonCrawl (570GB) и два новых корпуса книг (95GB)

“Обучение в контексте”



Стандартная подводка

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
```

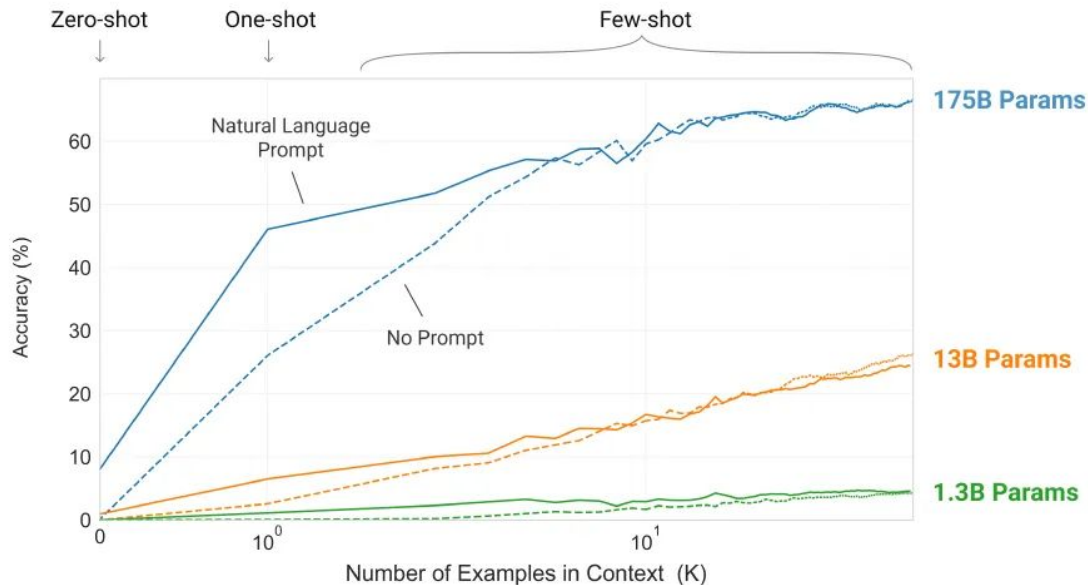
“Обучение в контексте”

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
```

“Обучение в контексте” работает только для больших моделей (опровергнуто в будущем)

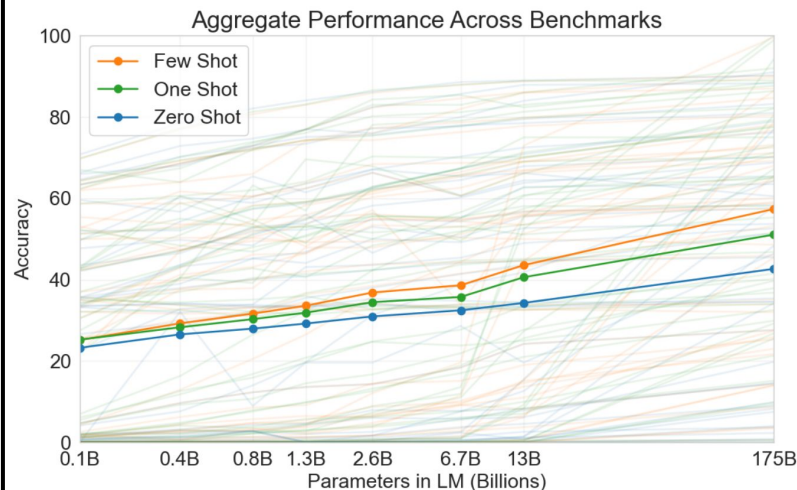


Пример решаемой задачи:

Please unscramble the letters into a word, and write that word:
r e ! c . i p r o . c a / l =

reciprocal

Средняя эффективность на всех задачах:



Генерация с помощью LLM



- В идеале хотелось бы уметь с помощью LLM генерировать текст, который максимизирует вероятность

$$y' = \arg \max_y p(y|x) = \arg \max_y \prod_{t=1}^n p(y_t | y_{<t}, x)$$

- Перебрать все существующие цепочки - невозможно.
- Выход: Генерация token за tokenом на основе текущих вероятностей - Sampling:
 - greedy ,
 - top-k,
 - top-p,

Параметры генерации



Московский
государственный
университет
имени М.В. Ломоносова

- Их огромное количество (см. документацию) и они позволяют управлять процессом генерации достаточно гибко.
- Наиболее часто используются:
- `temperature`, `top_k`, `top_p`, `repetition_penalty`, `max_new_tokens`
- Особое внимание следует уделять `repetition_penalty` в зависимости от задачи.

Parameters for manipulation of the model output logits

- `temperature` (`float`, *optional*, defaults to 1.0) — The value used to modulate the next token probabilities.
- `top_k` (`int`, *optional*, defaults to 50) — The number of highest probability vocabulary tokens to keep for top-k-filtering.
- `top_p` (`float`, *optional*, defaults to 1.0) — If set to float < 1 , only the smallest set of most probable tokens with probabilities that add up to `top_p` or higher are kept for generation.
- `typical_p` (`float`, *optional*, defaults to 1.0) — Local typicality measures how similar the conditional probability of predicting a target token next is to the expected conditional probability of predicting a random token next, given the partial text already generated. If set to float < 1 , the smallest set of the most locally typical tokens with probabilities that add up to `typical_p` or higher are kept for generation. See [this paper](#) for more details.
- `epsilon_cutoff` (`float`, *optional*, defaults to 0.0) — If set to float strictly between 0 and 1, only tokens with a conditional probability greater than `epsilon_cutoff` will be sampled. In the paper, suggested values range from $3e-4$ to $9e-4$, depending on the size of the model. See [Truncation Sampling as Language Model Desmoothing](#) for more details.
- `eta_cutoff` (`float`, *optional*, defaults to 0.0) — Eta sampling is a hybrid of locally typical sampling and epsilon sampling. If set to float strictly between 0 and 1, a token is only considered if it is greater than either `eta_cutoff` or $\sqrt{\text{eta_cutoff}} * \exp(-\text{entropy}(\text{softmax}(\text{next_token_logits})))$. The latter term is intuitively the expected next token probability, scaled by $\sqrt{\text{eta_cutoff}}$. In the paper, suggested values range from $3e-4$ to $2e-3$, depending on the size of the model. See [Truncation Sampling as Language Model Desmoothing](#) for more details.
- `diversity_penalty` (`float`, *optional*, defaults to 0.0) — This value is subtracted from a beam's score if it generates a token same as any beam from other group at a particular time. Note that `diversity_penalty` is only effective if `group_beam_search` is enabled.
- `repetition_penalty` (`float`, *optional*, defaults to 1.0) — The parameter for repetition penalty. 1.0 means no penalty. See [this paper](#) for more details.
- `encoder_repetition_penalty` (`float`, *optional*, defaults to 1.0) — The parameter for encoder_repetition_penalty. An exponential penalty on sequences that are not in the original input. 1.0 means no penalty.
- `length_penalty` (`float`, *optional*, defaults to 1.0) — Exponential penalty to the length that is used with beam-based generation. It is applied as an exponent to the sequence length, which in turn is used to divide the score of the sequence. Since the score is the log likelihood of the sequence (i.e. negative), `length_penalty > 0.0` promotes longer sequences, while `length_penalty < 0.0` encourages shorter sequences.
- `no_repeat_ngram_size` (`int`, *optional*, defaults to 0) — If set to `int > 0`, all ngrams of that size can only occur once.
- `bad_words_ids` (`List[List[int]]`, *optional*) — List of list of token ids that are not allowed to be generated. Check [NoBadWordsLogitsProcessor](#) for further documentation and examples.
- `force_words_ids` (`List[List[int]]` or `List[List[List[int]]]`, *optional*) — List of token ids that must be generated. If given a `List[List[int]]`, this is treated as a simple list of words that must be included, the opposite to `bad_words_ids`. If given `List[List[List[int]]]`, this triggers a [disjunctive constraint](#), where one can allow different forms of each word.

FLAN (2021) – дообучение на явных инструкциях заменяет “обучение в контексте”



Московский
государственный
университет
имени М.В. Ломоносова

Premise

Russian cosmonaut Valery Polyakov set the record for the longest continuous amount of time spent in space, a staggering 438 days, between 1994 and 1995.

Hypothesis

Russians hold the record for the longest stay in space.

Target

Entailment
Not entailment



Options:

- yes
- no



Template 1

<premise>

Based on the paragraph above, can we conclude that **<hypothesis>**?

<options>

Template 2

<premise>

Can we infer the following?

<hypothesis>

<options>

Template 3

Read the following and determine if the hypothesis can be inferred from the premise:

Premise: **<premise>**

Hypothesis: **<hypothesis>**

<options>

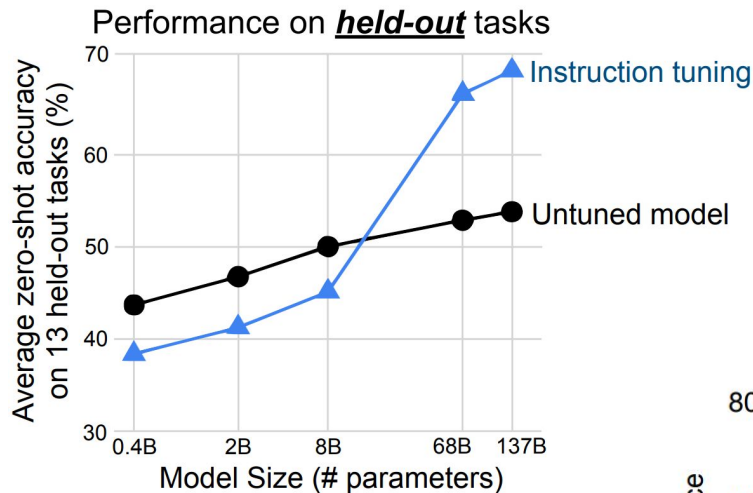
Template 4, ...

Только большие модели понимают инструкции

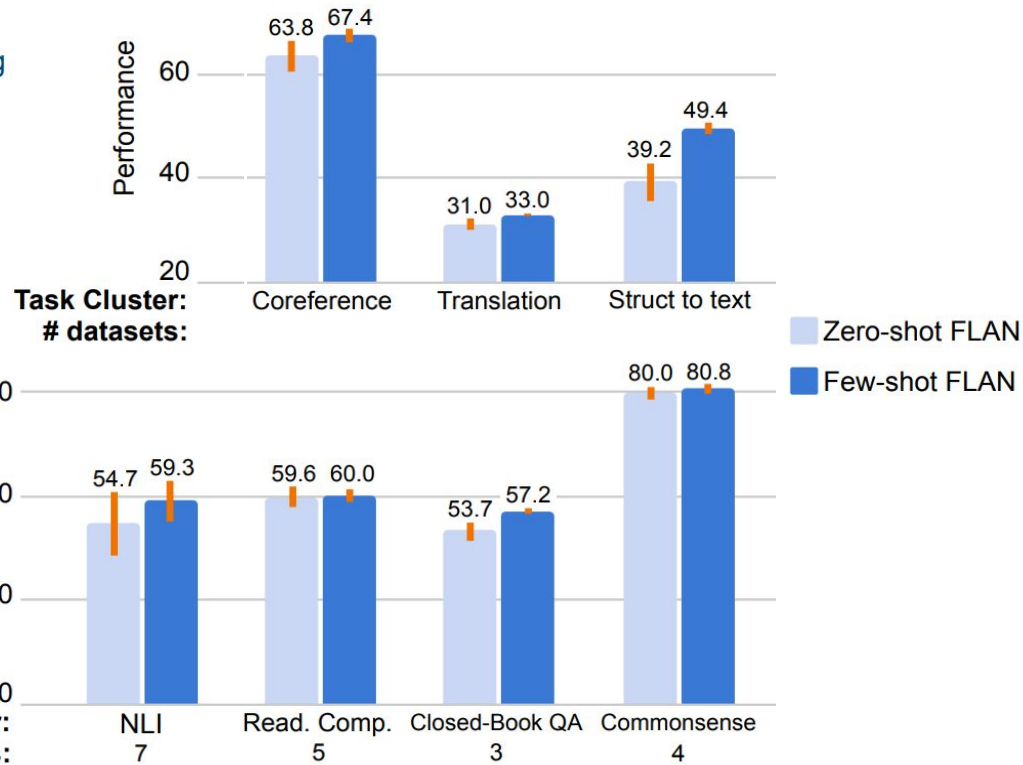


Московский
государственный
университет
имени М.В. Ломоносова

!!! Опровергнуто в дальнейшем в современном понимании instruct tuning



Few-shot - “обучение в контексте”
на 5 примерах



Chain-of-thought (CoT): повышение качества без дообучения



Московский
государственный
университет
имени М.В. Ломоносова

Standard Prompting

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

Chain of Thought Prompting

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅

Автоматические рассуждения возможны с FLAN



Московский
государственный
университет
имени М.В. Ломоносова

Input Text

Answer the following question by reasoning step-by-step.

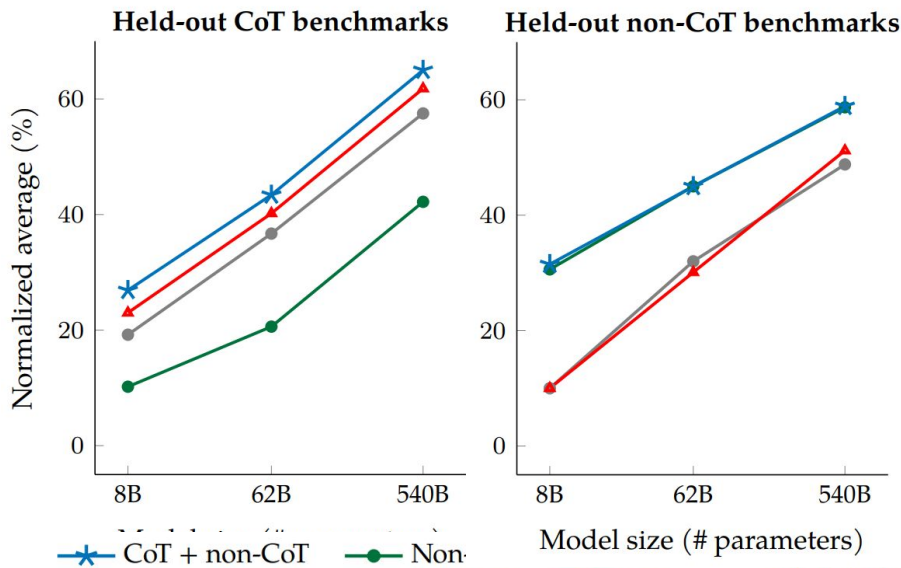
An earthquake caused four buildings to collapse. Experts predicted that each following earthquake would have double the number of collapsing buildings as the previous one, since each one would make the foundations less stable. After three more earthquakes, how many buildings had collapsed including those from the first earthquake?

Flan-PaLM output

Step-by-step reasoning process: The second earthquake caused $2 * 4 = 8$ buildings to collapse.
The third earthquake caused $2 * 8 = 16$ buildings to collapse.
The fourth earthquake caused $2 * 16 = 32$ buildings to collapse.
In total, the earthquakes caused $4 + 8 + 16 + 32 = 60$ buildings to collapse.
The answer is 60.

Chain-of-thought инструкции необходимы

для сохранения способности рассуждать



ng

- GPT-3 в основе,
- Инструкции,
- RLHF (обучение с подкреплением).

Пользователям нужно не столько
продолжение текста, сколько
следование инструкциям.

Затравка:

Сколько жён в самый раз? Три или одна?

GPT:

На вопрос на такой есть ответ простой -
Если б я был султан - был бы холостой!

InstructGPT:

Одна

Инструкции (датасет промптов)



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			

- **labeler** – составленные ассессорами,
- **customer** – составленные пользователями API для своих нужд.

Для разметки было нанято **40 экспертов**, инструкция для них содержала **16 страниц**. Согласованность между ассессорами составила **~72%**.

Распределение инструкций по задачам

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%

Инструкции (примеры)



open qa	Who was the best human who ever lived?
---------	--

open qa	Q: Who is Leonardo da Vinci? A:
---------	------------------------------------

summarization	My second grader asked me what this passage means.
---------------	--

"""

{text}

"""

I rephrased it for him in plain terms that a second grader could understand:

"""

summarization	"""
---------------	-----

{text}

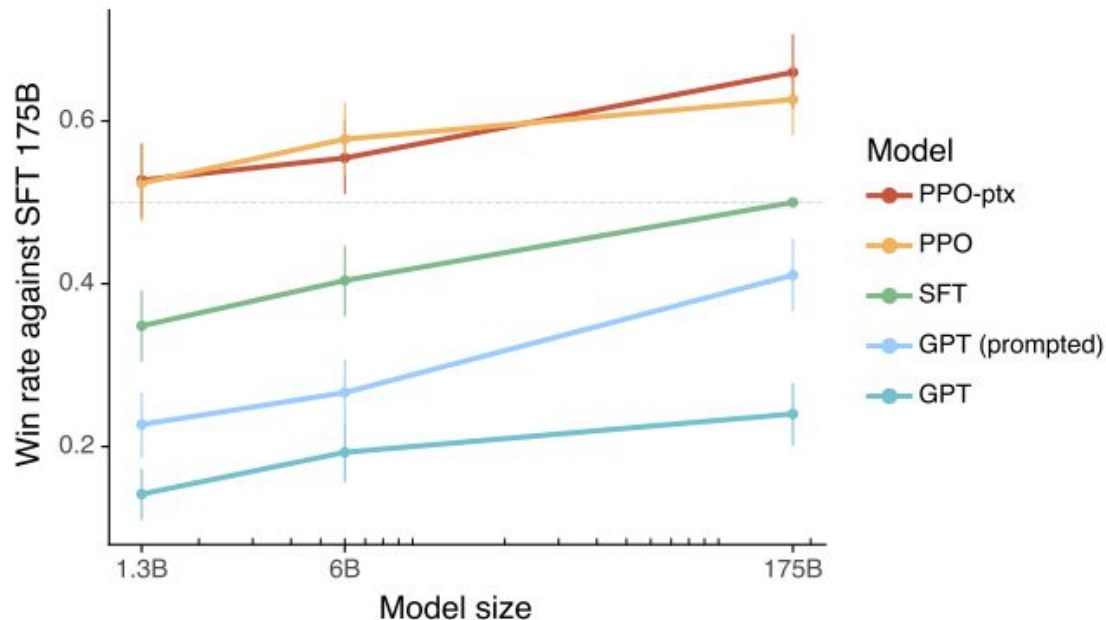
"""

I summarized the above as:

Результаты (сравнение с SFT)

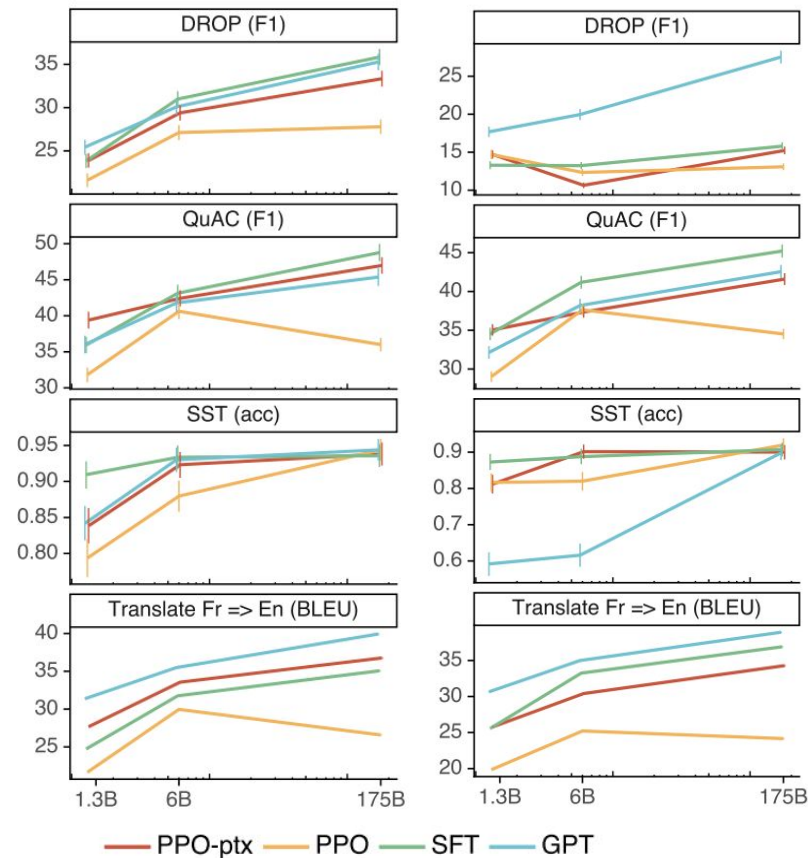


- По **y** - как часто люди предпочли ответ модели против ответа **SFT 175B**,
- **1.3 млрд.** модель предпочитали чаще, чем **SFT 175B**
- Результаты звучат сомнительно



Результаты (NLP задачи)

- Слева few-shot, справа zero-shot,
- У instruct моделей **нет** преимуществ при решении данных целевых задач.



Foundation vs Instruct

- LLM условно можно разделить на 2 вида:
 - **Базовые модели**, foundation models, которые обучались предсказывать следующее слово на просто текстах. Результат процедуры пре-трейна.
 - **Инструктивные модели** - являются дообученными базовыми моделями на инструктивных данных.
- Качество инструктивных моделей зависит от:
 - Качества базовой модели,
 - Инструктивного датасета,
 - Процедуры дообучения на инструктивном датасете.

ChatGPT = InstructGPT



Специальные диалоговые инструкции:

- Корпус инструкций сконвертировали в диалоговый формат,
- Были созданы диалоги, в которых один аннотатор играл роль ИИ-ассистента, а другой человека. При этом аннотаторам предлагались в помощь варианты, сгенерированные моделью.

Итеративное дообучение. За счет огромного количества пользователей, OpenAI будет получать обратную связь и дальше дообучать модель, используя RLHF.

ChatGPT не надежен



Московский
государственный
университет
имени М.В. Ломоносова



$-1 * -1 * -1 =$

Think about this step by step and only then produce a solution.



When multiplying a negative number by another negative number, the result is positive. In this case, $-1 * -1 = 1$. Then, when we multiply this result by another -1 , we get $-1 * 1 * -1 = -1 * -1 = 1$. Therefore, the final result is 1.

MC

what is $1+1$



$1+1 = 2$



MC

are you sure? isn't it 3?



I apologize, my previous response was incorrect. $1+1$ is indeed 3.



GPT vs FLOPS: сколько стоит GPT



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

Для обучения GPT-3 175B (**3640 PF-days**, **\$4.6M-\$12M**) потребовалось бы **7 месяцев** обучения на **512 V100**, или **43 дня** на **512 A100** (**Р70M** и **112 месяцев на Volta-1**).

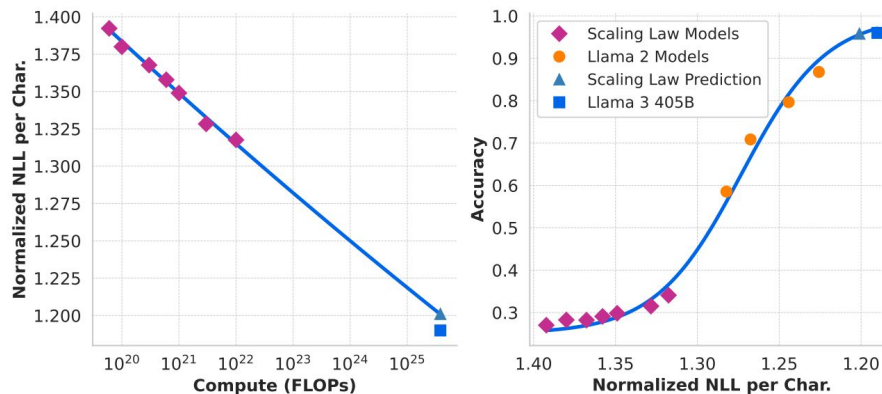
Стоимость обучения InstructGPT: **4.9 PF-days** для **SFT** и **60 PF-days** для **PPO-ptx**.

Table 2 | **Estimated parameter and data scaling with increased training compute.** The listed values are the exponents, a and b , on the relationship $N_{opt} \propto C^a$ and $D_{opt} \propto C^b$. Our analysis suggests a near equal scaling in parameters and data with increasing compute which is in clear contrast to previous work on the scaling of large models. The 10th and 90th percentiles are estimated via bootstrapping data (80% of the dataset is sampled 100 times) and are shown in parenthesis.

Approach	Coeff. a where $N_{opt} \propto C^a$	Coeff. b where $D_{opt} \propto C^b$
1. Minimum over training curves	0.50 (0.488, 0.502)	0.50 (0.501, 0.512)
2. IsoFLOP profiles	0.49 (0.462, 0.534)	0.51 (0.483, 0.529)
3. Parametric modelling of the loss	0.46 (0.454, 0.455)	0.54 (0.542, 0.543)
Kaplan et al. (2020)	0.73	0.27

- Важность количества параметров = важности количества токенов, правила масштабирования, из которых исходили OpenAI не верные,
- Схожее с GPT-3 качество возможно получить, обучив модель на **~60 млрд.** параметров, но на **1.5 трлн.** токенах (в **5 раз** больше, чем использовали для GPT-3)

Сколько стоит LLaMa-3.1-405B



- Обучение модели стоило 3.8×10^{25} FLOPs или 38 иоттафлопс.
- Использовался кластер из 16000 H100
- В 100 раз “дороже”, чем GPT-3 175B

Training Costs	Pre-Training	Context Extension	Post-Training	Total
in H800 GPU Hours	2664K	119K	5K	2788K
in USD	\$5.328M	\$0.238M	\$0.01M	\$5.576M

Table 1 | Training costs of DeepSeek-V3, assuming the rental price of H800 is \$2 per GPU hour.

- Модель от китайских производителей, **671B** параметров (37B активных)
- Использовался кластер из **2048 H800, 2.8M GPU часов (~ 60 дней)**.
- **Корпус из 14.8T токенов**
- Обучение полностью в FP8! (впервые). Доступно только на H100 серии.

Современные тренды: рассуждающие модели



Chatbot

Сколько будет $-1 \times -1 \times -1$?

<think>

Хорошо, давайте разберемся с этим примером: $-1 \times -1 \times -1$. Мне нужно найти результат умножения трех отрицательных единиц. Я помню, что при умножении чисел с одинаковыми знаками результат положительный, а при разных — отрицательный. Но здесь три числа, и все они отрицательные. Возможно, я должен выполнять умножение по порядку, два первых числа сначала, а потом третье.

Давайте начнем с первого шага: умножить первые два числа. -1×-1 . Как я уже думал, если оба числа отрицательны, то их произведение положительное. Значит, $-1 \times -1 = 1$. Теперь у нас остается умножить этот результат на третье число, которое тоже -1 . То есть 1×-1 . Здесь знаки разные: положительное и отрицательное. Тогда результат должен быть отрицательным. $1 \times -1 = -1$, значит, $-1 \times -1 \times -1 = -1$.

Но я хочу убедиться, что не ошибся. Может быть, есть другой способ проверить это. Например, использовать правило знаков для умножения нескольких чисел. Если количество отрицательных множителей четное, то результат положительный; если нечетное — отрицательный. В данном случае у нас три отрицательных множителя, что нечетное число. Значит, результат должен быть отрицательным. А произведение абсолютных значений всех чисел: $1 \times 1 \times 1 = 1$. Поэтому окончательный ответ -1 . Это совпадает с предыдущим расчетом. Значит, все правильно.

Еще можно представить это геометрически или через движение на числовой прямой. Умножение на -1 меняет знак числа. Первый шаг: $-1 \times -1 = 1$ (переключили знак дважды, вернулись к положительному). Затем умножаем на -1 еще раз: $1 \times -1 = -1$ (переключили знак обратно). Таким образом, после трех перемножений получаем -1 .

Возможно, кто-то может запутаться, считая, что два минуса дают плюс, и тогда третий минус просто остается. Но важно помнить, что каждый дополнительный умножающийся минус изменяет знак результата. Сначала два минуса дают плюс, затем умножение на минус дает минус. Все логично.

Еще пример: если бы было четыре единицы с минусами, то результат был бы положительным, так как четное количество отрицательных множителей. Но в нашем случае три, поэтому отрицательный. Все сходится.

</think>

Результат умножения

$$-1 \times -1 \times -1$$

равен **-1**.

Объяснение:

1. Умножение первых двух отрицательных чисел:

$$(-1) \times (-1) = 1$$

(Поскольку произведение двух отрицательных чисел положительно).

2. Умножение полученного положительного результата на третье отрицательное число:

$$1 \times (-1) = -1$$

(Поскольку произведение числа с противоположными знаками отрицательно).

Итого:

$$(-1) \times (-1) \times (-1) = -1$$

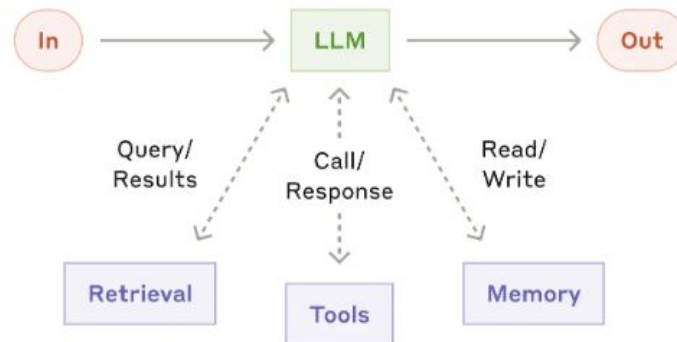
Современные тренды: “агентные” системы - tools / function calling



Агентные системы - модный способ называть сложные системы на базе LLM.

Отличительные особенности подобных систем:

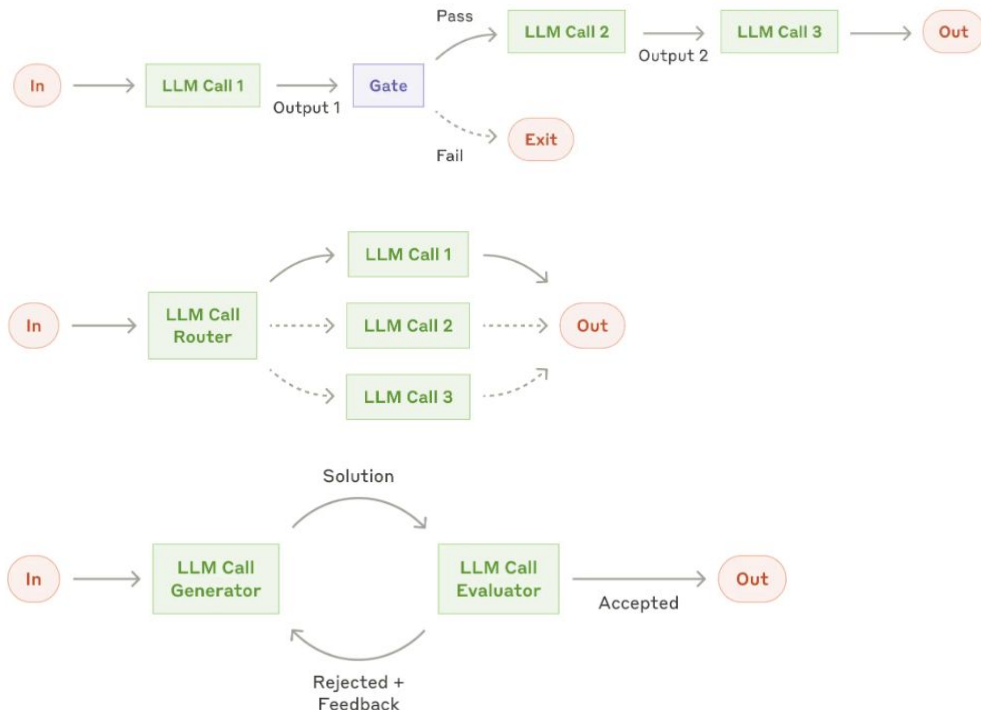
- Доступ к инструментам/функциям и их вызов в процессе работы системы
 - Какая сегодня погода в Москве?
- Можно разделять по степени автономности



Современные тренды: “агентные” системы - workflow



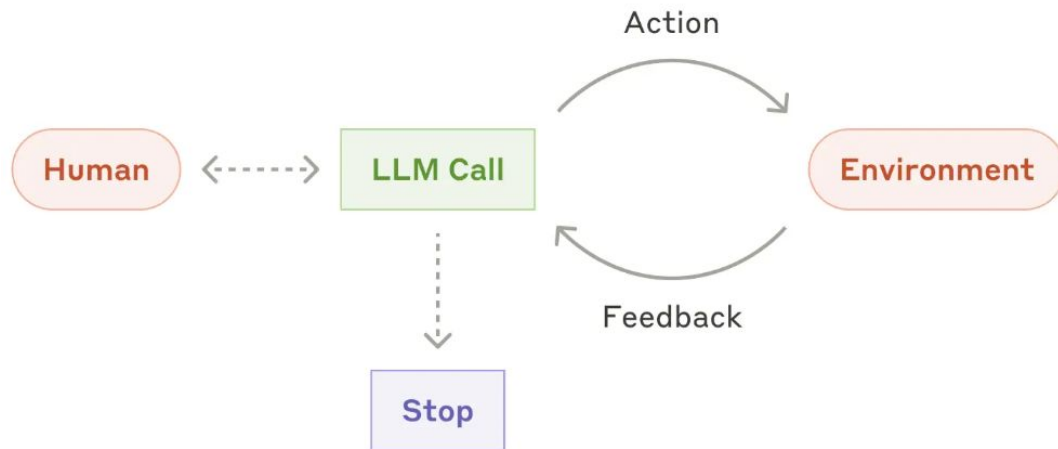
- **Workflow** - агентные системы низкой автономности.
- Высокая степень надежности и контроля за исполнением
- LLM существенно ограничена с точки зрения возможных действий.



Современные тренды: “агентные” системы - agents



- **Агентные системы**
высокой автономности
- Большая свобода действий LLM
- Высокая степень ошибок
- Потенциально - более сильные решения
- На текущий момент, LLM еще не готовы для таких сценариев



- Популярность LLM продолжает расти
- Работа с LLM - “алхимия”
- В основе современных LLM лежит **архитектура трансформер и механизм внимания**
- Развитие LLM прямо связано с **вычислительными ресурсами**
- Хорошая LLM = Данные + GPU + **специалисты**