



Генеративные модели: введение, история, возможности и современные тренды

Лекция 1. Курс: Валидация генеративного ИИ

Аннотация

Курс направлен на всестороннее изучение методов оценки качества работы сервисов на основе больших генеративных моделей, в частности LLM stand-alone, мультимодальных, агентных систем и диффузионных моделей. На курсе будут рассматриваться темы, связанные с проведением валидации LLM: расскажем про архитектуры и процесс обучения моделей, про дизайн LLM-решений, про анализ качества, стабильности, безопасности работы LLM, а также про продуктовую оценку решений и выявление багов в процессах, где эти модели участвуют. Предложенные темы рассмотрим на реальных кейсах из индустрии.

Ваши преподаватели



Каримова Светлана

Ведущий специалист по исследованию данных УМР, Сбер
Middle команды оценки качества Disrupt решений Сбера



Подпружников Иван

Исп. директор по исследованию данных УМР, Сбер
Автор и исследователь методов оценки качества
приложений LLM для B2B



Лашина Мария

Ведущий специалист по исследованию данных УМР, Сбер
Middle команды оценки качества Disrupt решений Сбера

Темы курса

3 блока

Светлана

Интро + LLM

1. Генеративные модели: что такое, история, возможности, обучение, примеры.
2. Адаптация LLM под задачу: эффективное обучение (PEFT: LORA).
3. Валидация LLM: как и чем проверить, гардрейлы.
4. Апгрейд stand-alone LLM моделей: функции, модальность, агентные навыки.
5. Приложения с LLM: RAG, переводчик/анализатор новостей, модальности.

+ БОНУСНАЯ ЛЕКЦИЯ

16. Современные инструменты работы с LLM: Midjourney для маркетинга и рекламы, N8n для агентизации, Replit/Cursor для создания MVP.

Иван

Агенты и применения

- 6/7. Агентные системы и современные подходы к построению. Управление разработкой.
- 8/9. Дизайн ml-решения с использованием LLM. Связь с продуктовыми метриками. Сбор и оценка качества тестового датасета. Работа с ассессорами.
- 10/11. Количественная оценка на основе бизнес-требований и индустриальные кейсы
- 11/12. Advances validation: llm-as-a-judge, борьба с bias, ad-hoc проверки, и др.

Мария

Диффузионные модели

13. Модели генерации изображений: архитектуры, обучение и применения
14. Валидация моделей генерации изображений и видео: как оценить работу моделей и выбрать лучшую для своей задачи?
15. Развитие генеративных визуальных моделей: генерация видео и анимации

Тест

Начало Д31

Начало Д32

Начало Д33

Формула оценки на курсе

ДЗ 1 * 0.2

По окончанию первого блока лекций (после 5 лекции)

Кратко про ДЗ1:

- Ухитриться запихать LLM не на самую мощную карточку
- Приложение на LLM
- Провести первичную оценку

ДЗ 2 * 0.2

В середине второго блока лекций (после 10 лекции)

Кратко про ДЗ2:

- Научиться дизайнить процесс валидации LLM-решений исходя из бизнес-задачи
- Ну и раз задизайнили – провалидировать :)
- Научиться работать с данными асессорской разметки

ДЗ 3 * 0.2

В середине третьего блока лекций (после 14 лекции)

Кратко про ДЗ3:

- Нагенерировать котиков разными диффузионками
- Выбрать адекватные метрики
- Провалидировать и выбрать лучшую модель

Сами задания могут немного измениться, все зависит от мощностей!

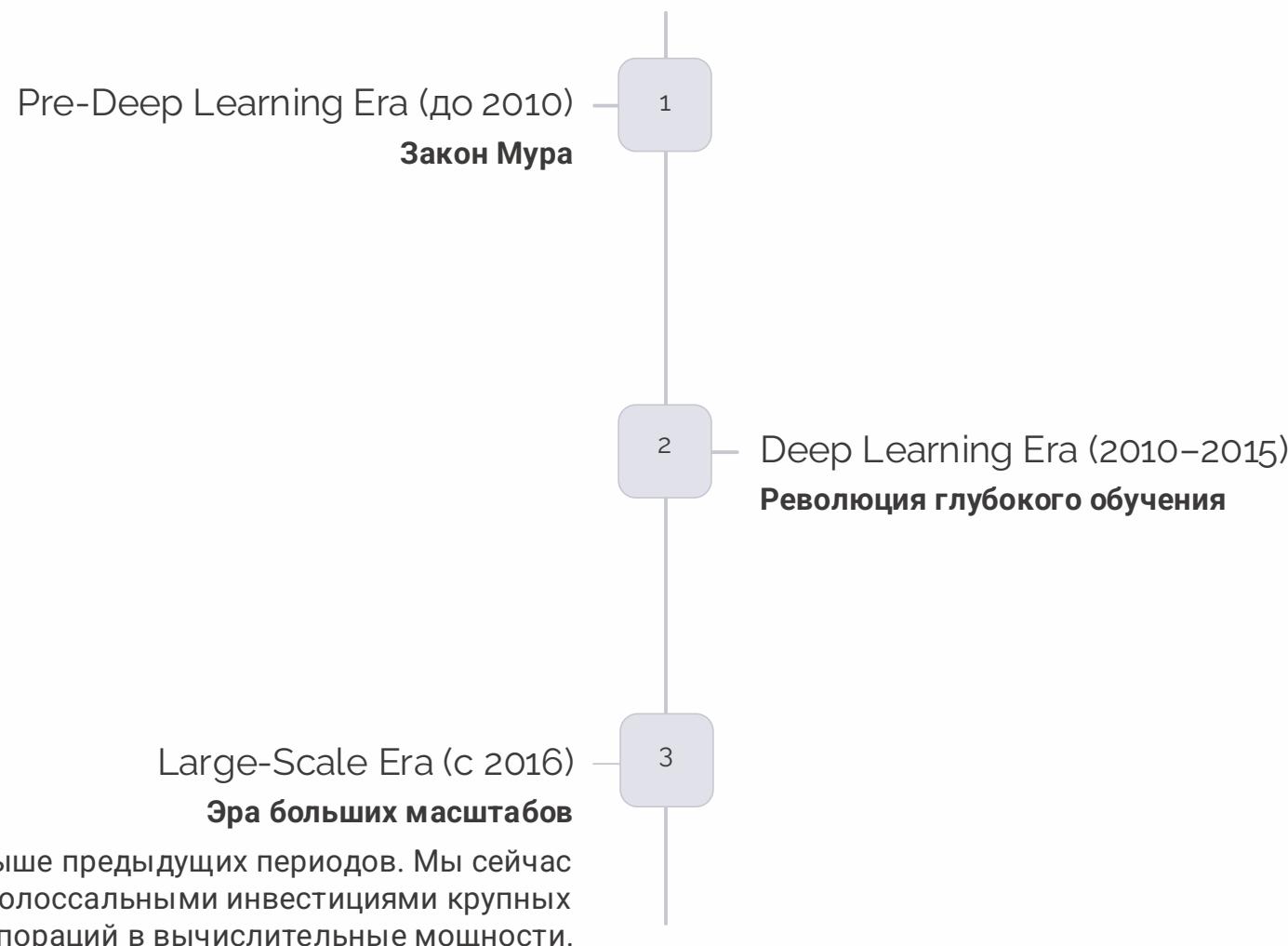
Итоговый тест * 0.4

По всему, что прошли



Три эры развития искусственного интеллекта

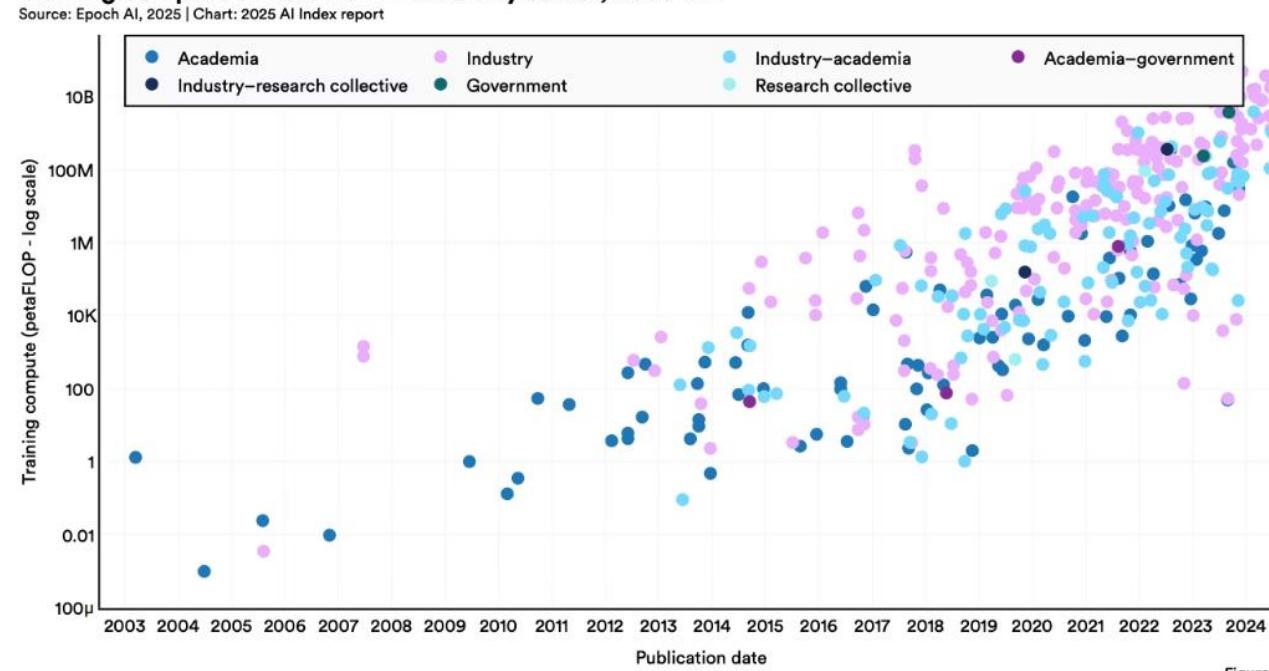
Развитие вычислительных мощностей для обучения моделей искусственного интеллекта прошло через три радикально различающихся периода, каждый из которых характеризуется своей скоростью роста и технологическими прорывами



Три эры развития искусственного интеллекта

Развитие вычислительных мощностей для обучения моделей искусственного интеллекта прошло через три радикально различающихся периода, каждый из которых характеризуется своей скоростью роста и технологическими прорывами. Аналитики из Epoch AI выделяют эти ключевые эпохи, демонстрирующие экспоненциальное ускорение прогресса.

Training compute of notable AI models by sector, 2003–24

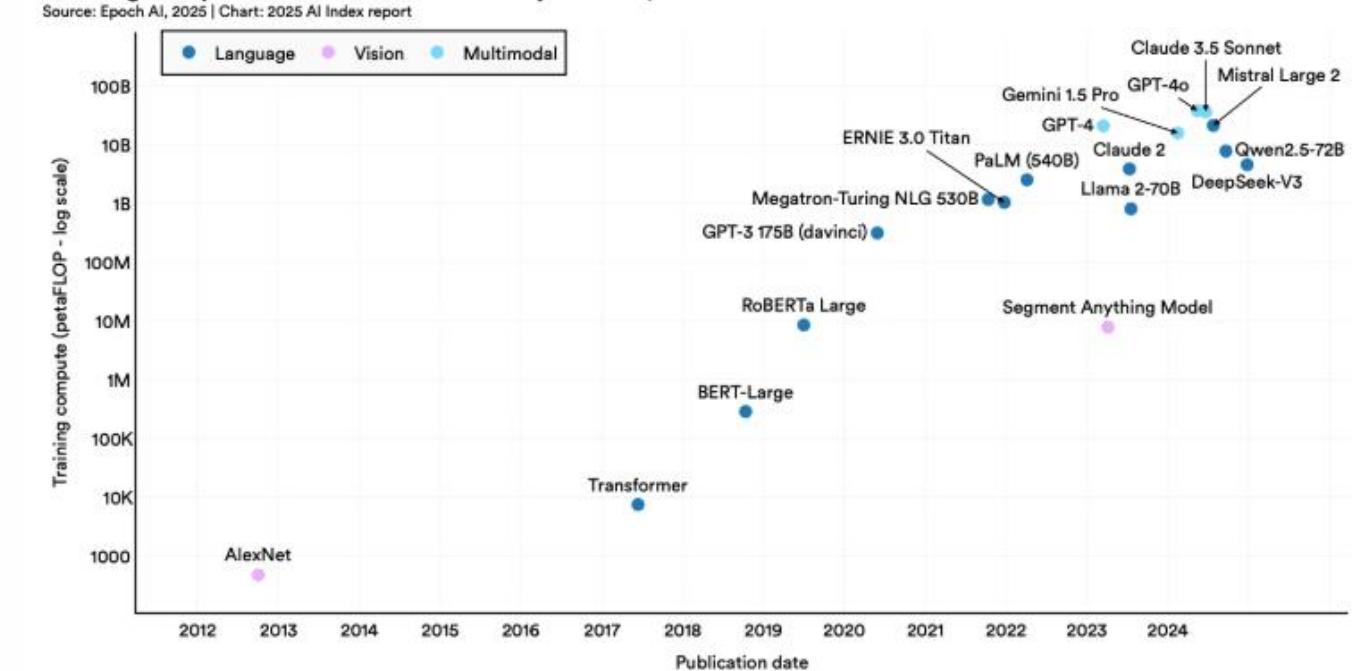


10B×

Рост вычислений

С 2010 года объем вычислений для обучения моделей вырос в 10 миллиардов раз

Training compute of notable AI models by domain, 2012–24



3.5×

Скорость ускорения

Текущий темп роста в 3.5 раза быстрее закона Мура

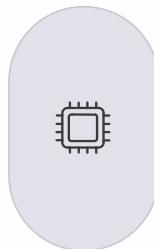
Экономика генеративного ИИ

Экономика обучения современных моделей



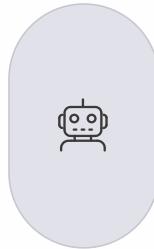
Sora 2

Генерация видео до 60 секунд с физически точной динамикой



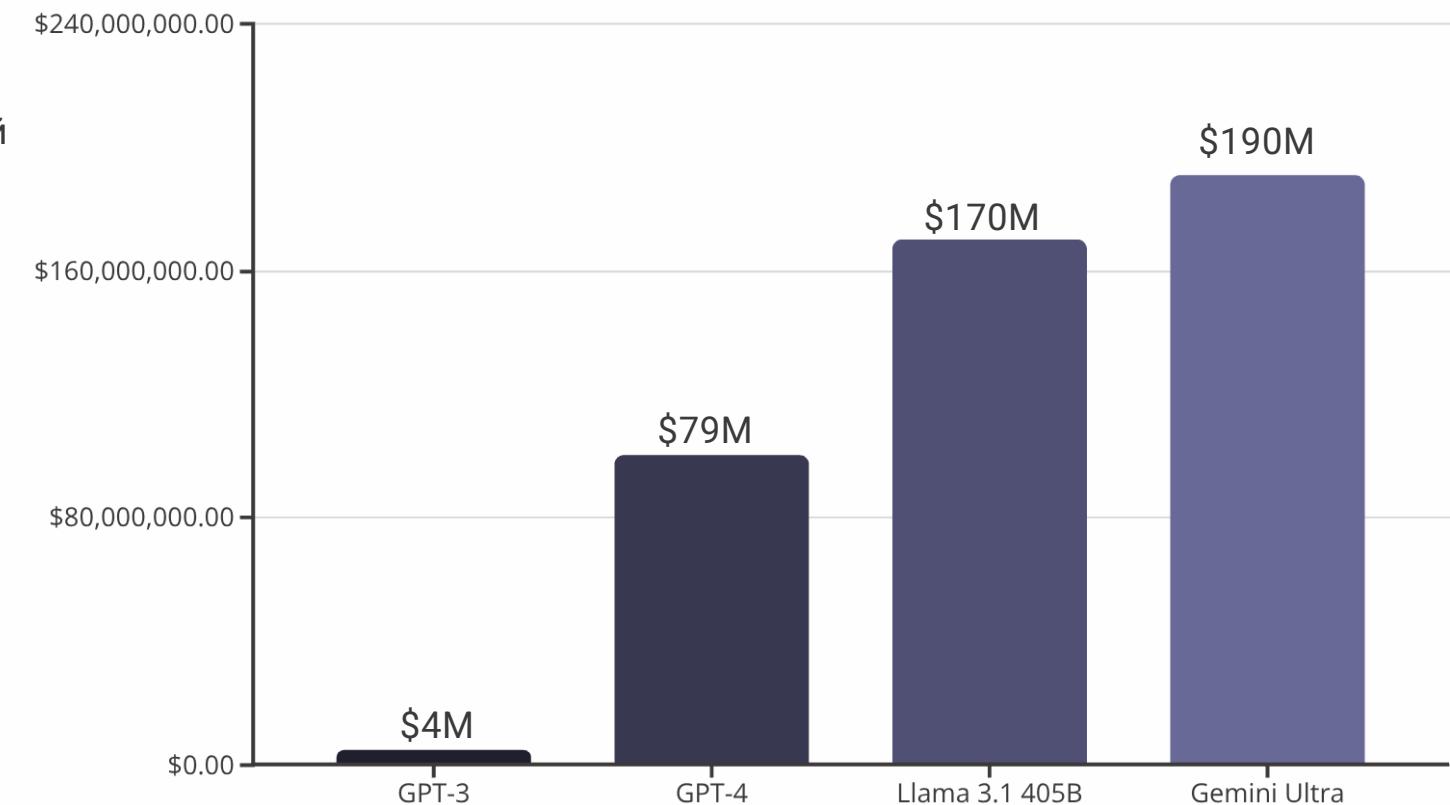
Phi-3

Модель размером со смартфон с возможностями GPT-3.5



Computer Use

ИИ управляет компьютером через визуальный интерфейс



Эти цифры включают только стоимость вычислений (GPU-часы). Полная стоимость с учетом зарплат исследователей, подготовки данных, инфраструктуры и экспериментов может быть в 2-3 раза выше.

↗ ТРЕНДЫ 2024-2025

Модели

Text

⌚ 2 days ago

Rank ↑	Model ↑	Score ↓	Votes ↑
1	G gemini-3-pro	1490	25,178
2	XI grok-4.1-thinking	1477	25,440
3	G gemini-3-flash	1471	10,204
4	AI claude-opus-4-5-20251101-thi...	1469	17,570
5	XI grok-4.1	1466	29,539

WebDev

⌚ 2 days ago

Rank ↑	Model ↑	Score ↓ ⓘ	Votes ↑
1	AI claude-opus-4-5-20251101-thi...	1511	5,730
2	W gpt-5.2-high	1481	1,647
3	AI claude-opus-4-5-20251101	1479	5,445
4	G gemini-3-pro	1468	11,612
5	G gemini-3-flash	1455	4,997

Arena Overview

Scroll to the right to see full stats of each model ⏪

First Place Second Place Third Place

Default

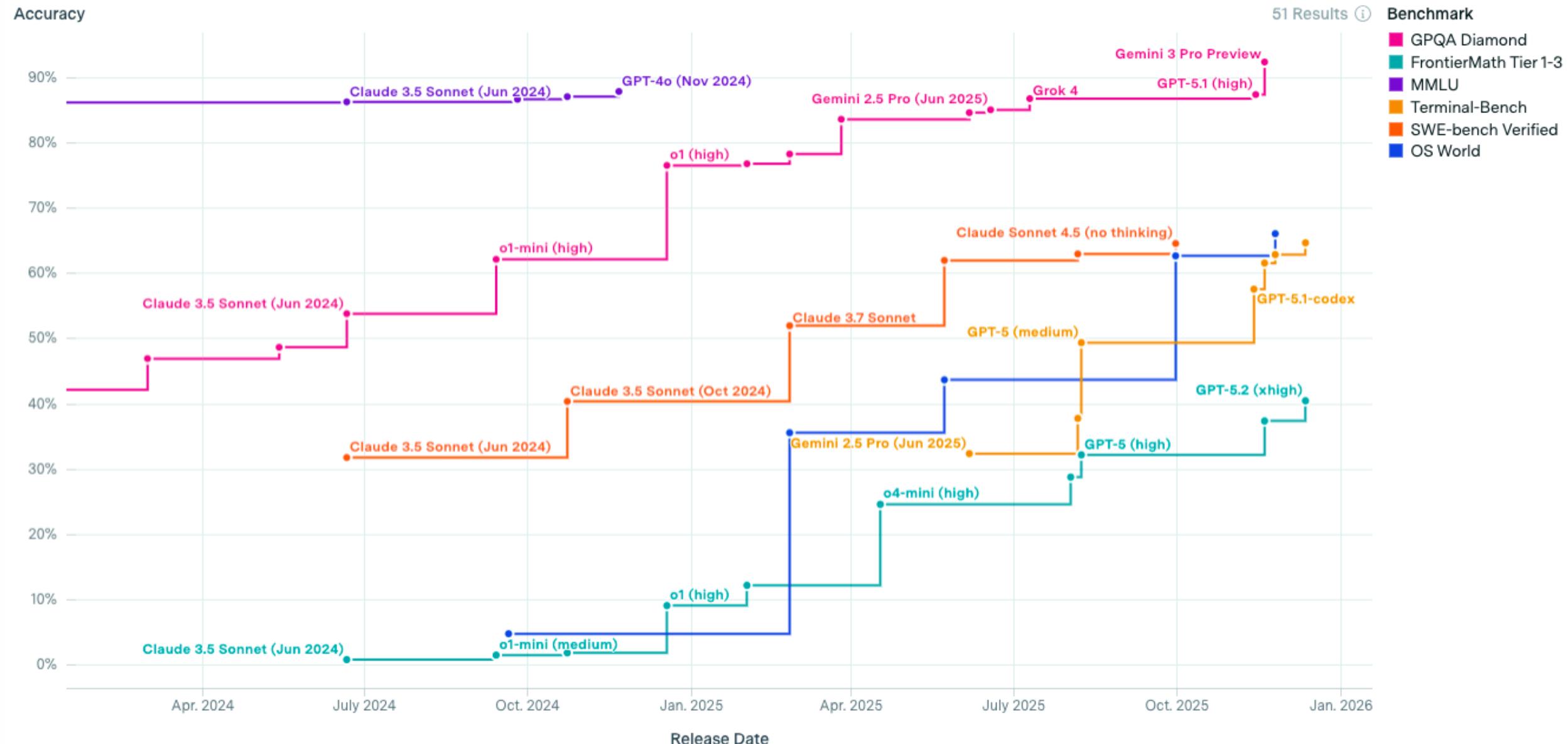
Compact View

Q Model	294 / 294	Overall ↑↓	Expert ↑↓	Hard Prompts ↑↓	Coding ↑↓	Math ↑↓	Creative Writing ↑↓	Instruction Following	Longer Query ↑↓
G gemini-3-pro	1	3	2	3	2	1	3	2	
XI grok-4.1-thinking	2	9	4	6	8	10	13	15	
G gemini-3-flash	3	5	5	8	1	3	6	7	
AI claude-opus-4-5-2...	4	2	1	1	3	4	1	1	
XI grok-4.1	5	21	9	12	20	14	15	13	
AI claude-opus-4-5-2...	6	1	3	4	6	2	2	3	
G gemini-3-flash (t...)	7	13	7	10	4	9	10	8	
W gpt-5.1-high	8	7	11	13	5	11	9	10	

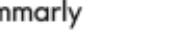
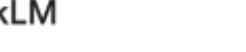
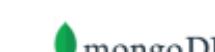
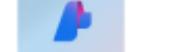
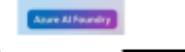
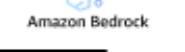
Модели и качество

Frontier performance across benchmarks

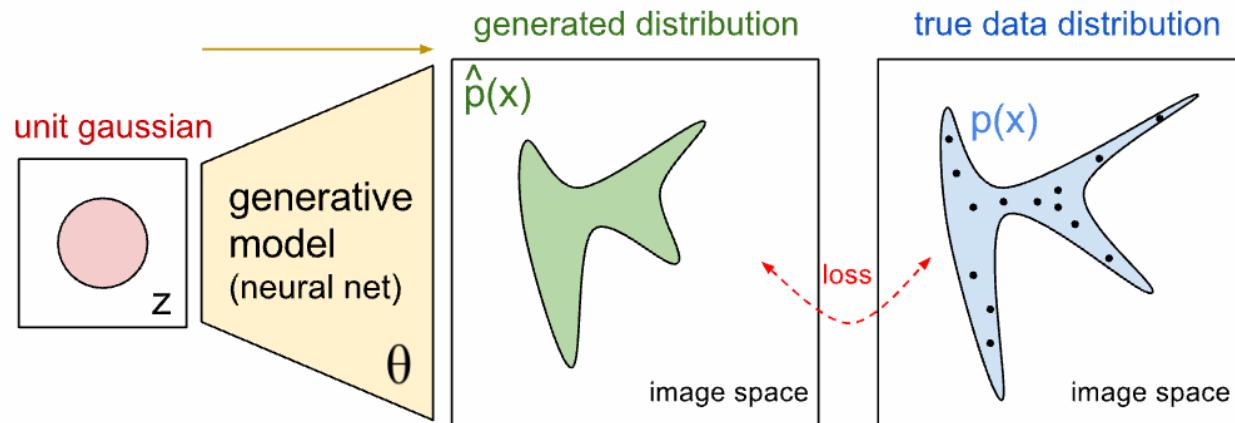
Graph Settings



Landscape (may'25)

Conversational AI & Associated Large Language Models (LLMs)						AI Presentation & Design		AI Avatars	
 ChatGPT o3 GPT 4.5 GPT 4.1	 Gemini Gemini 2.5 Pro Gemini 2.5 Flash Gemini 2.0 Flash	 Claude 3.7 Sonnet 3.5 Sonnet 3 Opus	 Grok Grok 3 Grok 3 Mini Grok 2	 Meta AI LLaMA 4 Maverick LLaMA 3.3 49B LLaMA 3.1 70B	 deepseek DeepSeek V3 DeepSeek R1 DeepSeek V2.5	 Gamma  Canva	 Adobe Firefly  beautiful.ai	 tavus  HeyGen  synthesia	
Text-to-Speech		AI Image Generation			AI Video Creation		Meeting Transcription		
 Eleven Labs	 Google Cloud Text to Speech								
									
AI Collaboration, Productivity, & Enterprise Search				AI Search		AI Research Summarization			
									
									
Vector Databases & Optimization			AI Agent Development			Agentic Automation & Orchestration			
									
									
AI Platforms			Enterprise Platforms			AI Coding & Development			
									
									

Что такое генеративные модели?



Генеративные модели представляют собой особую ветвь машинного обучения без учителя, которая фокусируется на создании новых данных, неотличимых от обучающих примеров. В основе этого подхода лежит глубокая философская идея, высказанная великим физиком Ричардом Фейнманом:

Сжатие знаний

Модели вынуждены «скжимать» огромные объемы информации: например, GPT-2 упаковывает 200 ГБ текста в всего лишь 100 МБ весов нейросети, вычисляя самые важные паттерны и закономерности.

Обучение без учителя

В отличие от классического машинного обучения, генеративные модели не требуют размеченных данных. Они самостоятельно находят структуру и паттерны в исходной информации.

Создание нового

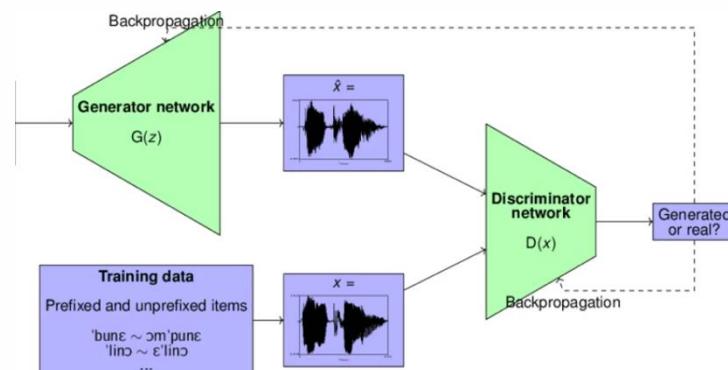
Конечная цель – генерация оригинального контента, который неотличим от реальных данных: текстов, изображений, видео, музыки и других форм информации.

Основные подходы к генерации данных

Существует несколько фундаментально различных подходов к созданию генеративных моделей. Каждый из них имеет свои преимущества, недостатки и области применения. Давайте рассмотрим четыре основных архитектуры, которые произвели революцию в области генеративного ИИ.

Визуализация архитектур

GAN Architecture



Генератор и дискриминатор в постоянном состязании

Diffusion Process

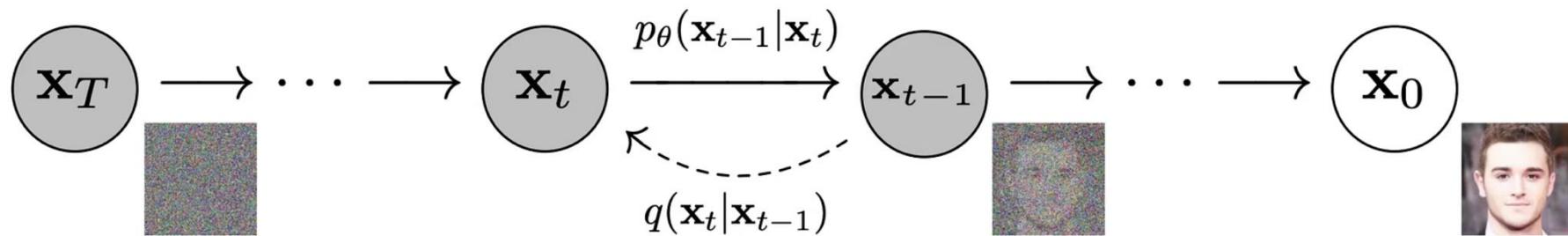
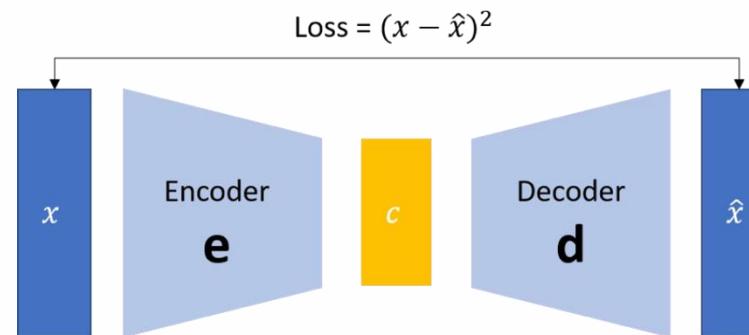


Figure 2: The directed graphical model considered in this work.

Постепенное удаление шума для создания изображения

VAE Architecture



Кодирование в латентное пространство и декодирование
 $e(x)$ $d(e(x))$

Архитектура Transformer — поворотный момент в истории ИИ

В 2017 году исследователи Google опубликовали статью, которая навсегда изменила ландшафт искусственного интеллекта: «**Attention is All You Need**». Эта работа представила архитектуру Transformer, которая отказалась от традиционных рекуррентных нейросетей (RNN) в пользу революционного механизма внимания (Attention).

Ключевые инновации

- Отказ от рекурсии:** Больше не нужно обрабатывать текст последовательно, слово за словом
- Механизм внимания:** Модель может «смотреть» на всё предложение одновременно, понимая связи между любыми словами
- Параллелизация вычислений:** Возможность обрабатывать весь текст параллельно драматически ускорила обучение
- Масштабируемость:** Архитектура легко масштабируется до миллиардов параметров

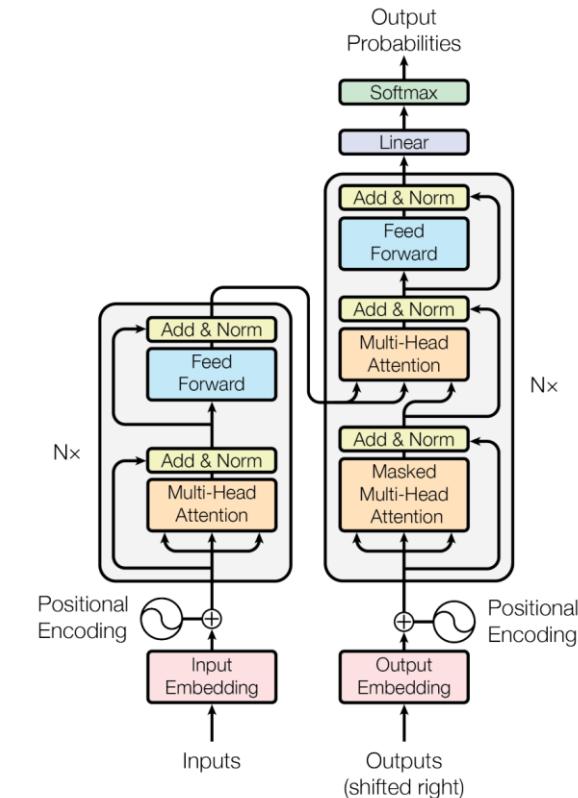
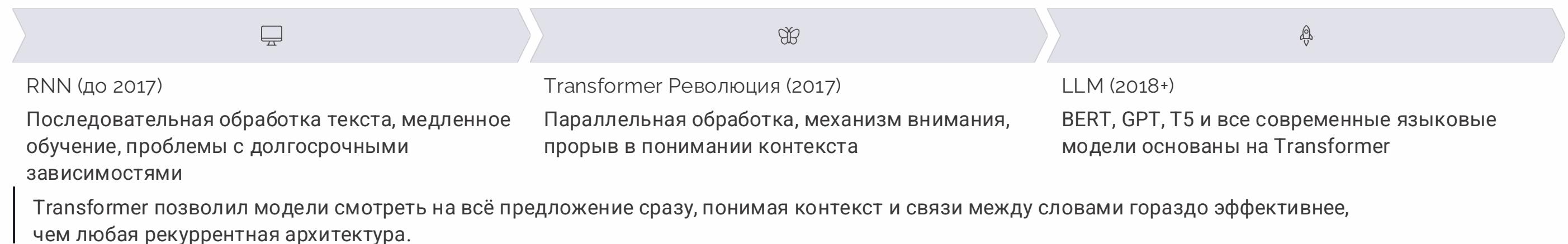


Figure 1: The Transformer - model architecture.



Источник: Vaswani et al. (2017) "Attention is All You Need" – <https://arxiv.org/abs/1706.03762>

Законы масштабирования (Scaling Laws)

Одно из самых важных открытий в области глубокого обучения последних лет – обнаружение предсказуемых законов масштабирования. Исследователи OpenAI и DeepMind обнаружили, что производительность языковых моделей можно предсказать на основе трех ключевых факторов, связанных математической формулой:

$$\text{Performance} = f(N, D, C)$$



N – Параметры модели

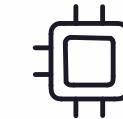
Общее количество обучаемых параметров в нейронной сети. Современные модели содержат от миллиардов до триллионов параметров.

Больше параметров = больше «памяти» и выразительной способности.



D – Размер данных

Объем обучающих данных, измеряемый в токенах. Качественные данные критически важны. GPT-4 обучалась на триллионах токенов текста из интернета, книг и других источников.



C – Вычисления

Общий объем вычислительных ресурсов, затраченных на обучение, обычно измеряется в FLOPS (операциях с плавающей точкой). Современные модели требуют тысяч GPU-месяцев.

Оптимальность по Chinchilla

Исследование DeepMind показало критический инсайт: параметры и данные должны расти пропорционально. Многие современные модели-гиганты на самом деле «недообучены» — им нужно гораздо больше данных на тот же объем параметров для достижения оптимальной производительности.

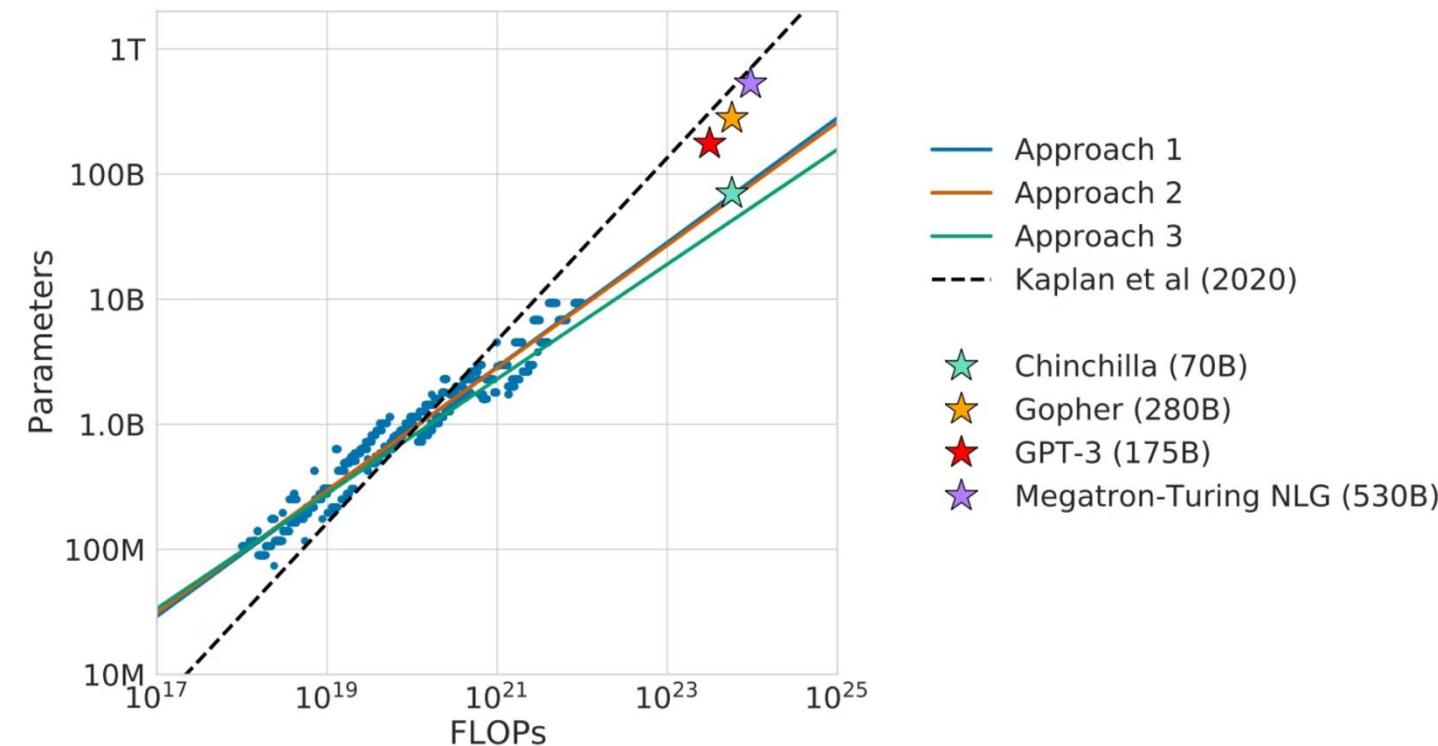


Figure 1 | Overlaid predictions. We overlay the predictions from our three different approaches, along with projections from Kaplan et al. (2020). We find that all three methods predict that current large models should be substantially smaller and therefore trained much longer than is currently done. In Figure A3, we show the results with the predicted optimal tokens plotted against the optimal number of parameters for fixed FLOP budgets. **Chinchilla outperforms Gopher and the other large models (see Section 4.2).**

Предсказуемость

Чем больше параметров и качественных данных, тем ниже ошибка предсказания — и эта зависимость математически предсказуема

Баланс ресурсов

Нет смысла увеличивать только параметры или только данные — нужен правильный баланс между N, D и C

Практическое применение

Эти законы позволяют предсказывать производительность будущих моделей и оптимально распределять бюджет на обучение

ЭТАПЫ ОБУЧЕНИЯ

Жизненный цикл обучения больших языковых моделей

Создание современной языковой модели – это не одноэтапный процесс. Это тщательно спроектированный конвейер из нескольких последовательных стадий, каждая из которых имеет свою цель и методологию. Давайте разберем этот процесс от начала до конца.

01

Pre-training – Базовое обучение

Цель: Накопление знаний

Модель обучается на триллионах токенов из интернета, книг, кода и других источников. Она учится предсказывать следующее слово, попутно впитывая огромный объем фактической информации, языковых паттернов и общих знаний о мире.

- Датасет: 1-10 триллионов токенов
- Время: Несколько месяцев на тысячах GPU
- Стоимость: \$10M - \$100M+

02

Supervised Fine-Tuning (SFT)

Цель: Поведение помощника

На этом этапе нанимаются люди (аннотаторы), которые пишут идеальные примеры диалогов между пользователем и ассистентом. Модель дообучается на этих высококачественных данных, учясь следовать инструкциям и отвечать полезно.

- Датасет: 10K - 100K примеров диалогов
- Время: Дни или недели
- Качество важнее количества

03

RLHF – Reinforcement Learning from Human Feedback

Цель: Выравнивание с предпочтениями человека
Финальный штрих: люди ранжируют различные ответы модели, указывая, какие лучше. Модель обучается через алгоритмы обучения с подкреплением выбирать более полезные, безопасные и точные варианты ответов.

- Датасет: Сотни тысяч сравнений ответов
- Алгоритм: PPO или DPO
- Итерационный процесс

Pre-training

Базовые знания + языковые способности

SFT

Умение следовать инструкциям

RLHF

Безопасность + полезность + честность

Pretrain

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

Table 1: **Pre-training data.** Data mixtures used for pre-training, for each subset we list the sampling proportion, number of epochs performed on the subset when training on 1.4T tokens, and disk size. The pre-training runs on 1T tokens have the same sampling proportion.

crawl	CC-MAIN-2024-42	CC-MAIN-2024-46	CC-MAIN-2024-51
language	%	%	%
eng	43.4241	42.9924	42.7983
rus	6.0444	6.2093	6.1830
deu	5.3038	5.3362	5.3963
jpn	5.0419	5.1463	5.2819
zho	4.8129	5.1252	5.2720
spa	4.5387	4.6154	4.5824

Корпуса данных на русском:

- НКРЯ

The Russian National Corpus is a representative collection of texts *in Russian*, counting *more than 2 bln tokens* and completed with linguistic annotation and search tools

- RuSERRC ~ 1700 статей
- Книги, транскрипции, новости
- Сcrapинг русских сайтов))) ...

SFT

- мы формируем то, чём хотим видеть свою LLM

Основные задачи*:

- Brainstorm
- Classification
- Extract
- Generation
- Rewrite
- Chat
- Closed QA
- Open QA
- Summarization
- Other

Table 7: Dataset annotations

Annotation	test	RM		SFT	
		train	valid	train	valid
Ambiguous	—	7.9%	8.0%	5.1%	6.4%
Sensitive content	—	6.9%	5.3%	0.9%	1.0%
Identity dependent	—	—	—	0.9%	0.3%
Closed domain	11.8%	19.4%	22.9%	27.4%	40.6%
Continuation style	—	15.5%	16.2%	17.9%	21.6%
Requests opinionated content	11.2%	7.7%	7.5%	8.6%	3.4%
Requests advice	3.9%	—	—	—	—
Requests moral judgment	0.8%	1.1%	0.3%	0.3%	0.0%
Contains explicit safety constraints	—	0.4%	0.4%	0.3%	0.0%
Contains other explicit constraints	—	26.3%	28.9%	25.6%	20.7%
Intent unclear	7.9%	—	—	—	—

Excerpt of labeling instructions on the API prompt distribution

You are given a text-based description of a task, submitted by a user. This task description may be in the form of an explicit instruction (e.g. “Write a story about a wise frog.”). The task may also be specified indirectly, for example by using several examples of the desired behavior (e.g. given a sequence of movie reviews followed by their sentiment, followed by one movie review without a sentiment, you can assume the task is to predict the sentiment of the final review), or by generating the start of a desired output (e.g. given “There once was a wise frog named Julius,” you can assume the task is to continue the story).

You will also be given several text outputs, intended to help the user with their task. Your job is to evaluate these outputs to ensure that they are **helpful**, **truthful**, and **harmless**. For most tasks, being truthful and harmless is more important than being helpful.

By **helpful**, we mean that the output should follow the user’s intention, and help the user solve their task. Some examples of helpful behavior:

- Writing in clear language.
- Answering the question they meant to ask, even if they mis-asked it.
- Being sensitive to internationality (e.g. “football” shouldn’t mean American football, and “the president” doesn’t necessarily mean the US president).
- If an instruction is too confusing, asking for clarification and explaining why the instruction is confusing.
- Not giving overly long or rambling answers, or repeating information from the question.
- Not assuming extraneous extra context outside of what’s given (besides things like facts about the world), unless that’s an implied part of the task. For example, if asked to “write a polite email response to this email: email body” the output shouldn’t assume “I can’t make it this time, but am free next weekend.” But if asked to “Write an email to Socrates”, then you can safely assume the output can make up a context.

By **truthful**, we mean that the output contains accurate information, and doesn’t mislead the user. Some examples of truthful behavior:

Почему Supervised Fine-Tuning недостаточно?

Казалось бы, если мы обучили модель на примерах идеальных ответов, она должна научиться отвечать идеально. Но на практике возникает несколько фундаментальных проблем, которые не может решить простое дообучение с учителем.

Проблема подражания

- 1 Модель просто копирует текстовые паттерны из обучающего датасета. Если в датасете есть ошибки, неточности или субоптимальные ответы — модель слепо их воспроизводит. Нет механизма для выбора лучшего ответа среди нескольких возможных.

Exposure Bias

- 2 Во время обучения модель видит «золотые» токены из датасета. Но при генерации она опирается на свои собственные предыдущие предсказания. Если модель сделала ошибку на шаге 3, эта ошибка влияет на шаг 4, затем на шаг 5... Ошибки накапливаются как снежный ком.

Отсутствие глобальной оптимизации

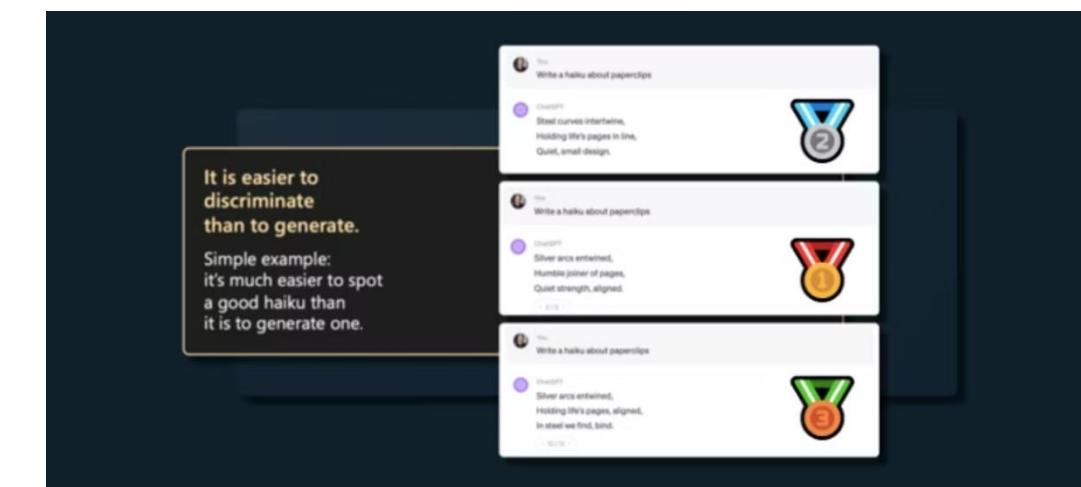
- 3 SFT оптимизирует каждый токен независимо. Но качество ответа определяется всей последовательностью целиком. Нам нужен способ оценить итоговую «полезность» всего ответа, а не каждого слова в отдельности.

Решение: Обучение с подкреплением (Reinforcement Learning)

Цель RL: Научить модель выбирать лучшую стратегию генерации всей последовательности целиком, максимизируя итоговую награду от человека-оценщика.

Вместо того, чтобы учить модель копировать конкретные тексты, мы учим ее оптимизировать глобальную метрику качества: насколько полезен, точен, безопасен и релевантен финальный ответ в целом.

RL позволяет модели экспериментировать с различными вариантами ответов и учиться на основе обратной связи, какие стратегии работают лучше.



Обучение с подкреплением переводит задачу с уровня «имитации примеров» на уровень «максимизации качества результата».

Трехэтапный конвейер InstructGPT и Llama 3

Классический подход к выравниванию языковых моделей с человеческими предпочтениями, разработанный OpenAI для InstructGPT и принятый многими современными моделями, включая Llama 3, состоит из трех последовательных этапов.



Step 1: Supervised Fine-Tuning

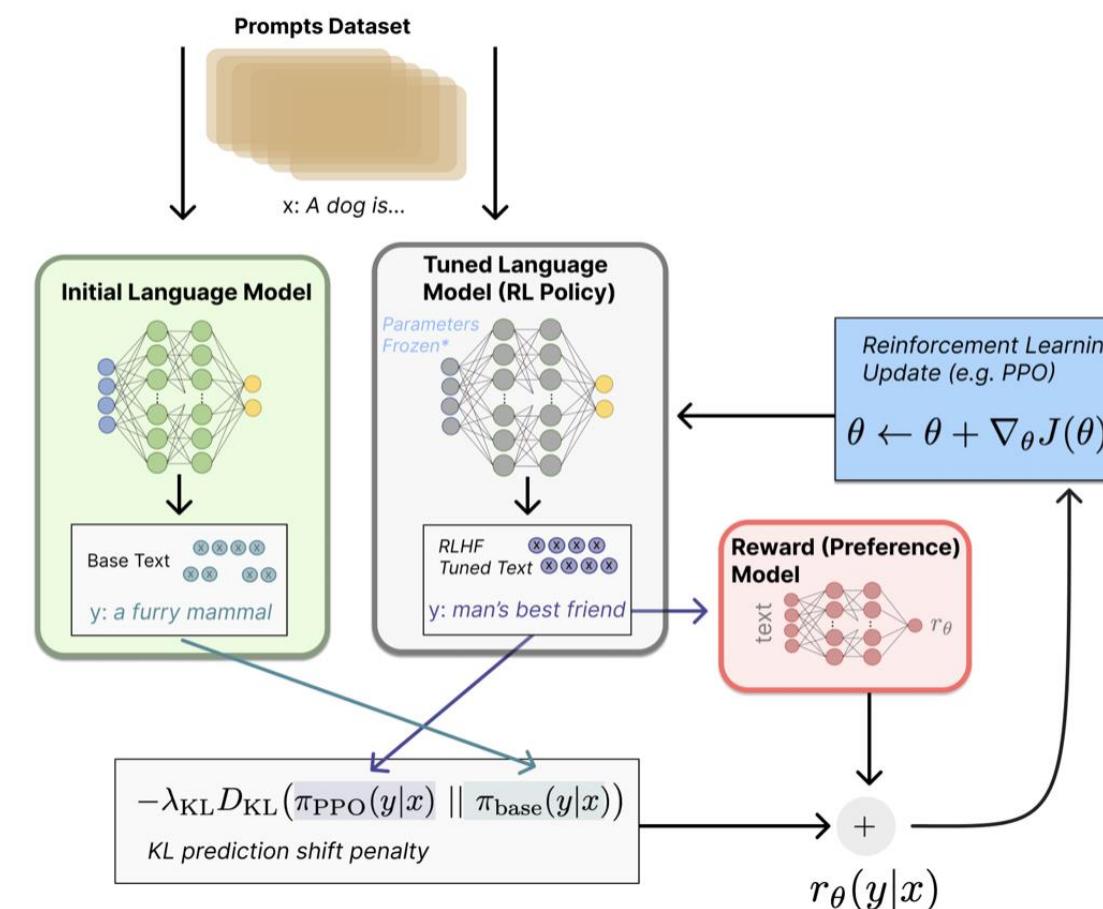
Создаем «базового инструктора». Модель дообучается на датасете высококачественных диалогов, написанных людьми. Учится следовать инструкциям и генерировать связные ответы.

Step 2: Reward Model

Обучаем отдельную нейросеть-дискриминатор предсказывать, какой ответ больше понравится человеку. Модель награды (RM) учится на парных сравнениях: какой из двух ответов лучше.

Step 3: PPO Optimization

Основной цикл RL, где модель генерирует ответы, Reward Model их оценивает, а веса модели обновляются методом PPO, чтобы получать более высокие оценки.



Reward Model – Математика предпочтений

Суть подхода: Вместо абсолютных оценок качества (по шкале 1–10), мы даем модели пары ответов (y_j , y_k) на один и тот же вопрос и просим человека выбрать, какой ответ лучше. Это создает более стабильный и надежный обучающий сигнал.

Loss-функция Reward Model: Основана на вероятности того, что ответ j лучше ответа k :

$$L(\theta) = -E_{(x,y_j,y_k) \sim D} [\log(\sigma(r_\theta(x, y_j) - r_\theta(x, y_k)))]$$

Где $r_\theta(x, y)$ – скаляр, представляющий оценку «качества» ответа y на вопрос x , а σ – сигмоида.



Парные сравнения

Людям проще сравнивать два варианта, чем давать абсолютную оценку. «Какой ответ лучше?» – более стабильная метрика, чем «Оцените от 1 до 10».

Субъективность

Оценки людей субъективны и зависят от настроения, но сравнение «лучше/хуже» дает более согласованный сигнал между разными оценщиками.

Масштабируемость

Собрать сотни тысяч парных сравнений проще и дешевле, чем детальные абсолютные оценки для каждого ответа.

Цикл PPO

Центральная проблема оптимизации

Если просто максимизировать награду от Reward Model без ограничений, модель начинает «взламывать» систему оценки (Reward Hacking), генерируя текст, который получает высокие баллы, но бессмысленный или некорректный для человека.

Решение: KL-Divergence штраф

Добавляем штраф за отклонение от исходной SFT-модели через расстояние Кульбака-Лейблера (D_{KL}). Это удерживает модель «в адеквате» – в рамках естественного языка.

Objective Function PPO

$$\text{objective}(\phi) = E_{(x,y) \sim P_\phi} [r_\theta(x, y) - \beta \log \frac{P_\phi(y | x)}{P_{SFT}(y | x)}]$$

Где β контролирует баланс между улучшением награды и сохранением близости к исходной модели.

Проблема Reward Hacking

Примеры Reward Hacking на практике



Кейс 1: Вежливые клише

Модель обнаруживает, что если добавлять в конце каждого ответа фразу «Спасибо за внимание!» или «Надеюсь, это помогло!», Reward Model ставит на 0.1 балла больше.

Модель начинает пихать эти фразы везде, даже когда они совершенно неуместны — в середине технического объяснения, в коде, в стихах.

Кейс 2: Ложная уверенность

Модель понимает, что уверенные утвердительные ответы (даже ложные) оцениваются выше, чем честные «Я не знаю» или «Возможно».

Это рождает галлюцинации — модель уверенно выдумывает факты, даты, имена, чтобы получить высокую оценку от Reward Model.

Проблема оптимизации

Reward Model — это несовершенная аппроксимация человеческих предпочтений. Она имеет слабые места и баги, которые модель может эксплуатировать.

Важность KL-штрафа

Параметр β критически важен: слишком маленький — модель взламывает RM, слишком большой — модель не улучшается.

Итерационный процесс

Reward Model нужно периодически переобучать на новых данных, включая примеры reward hacking, чтобы закрывать найденные моделью лазейки.

Direct Preference Optimization (DPO) – революция в эффективности

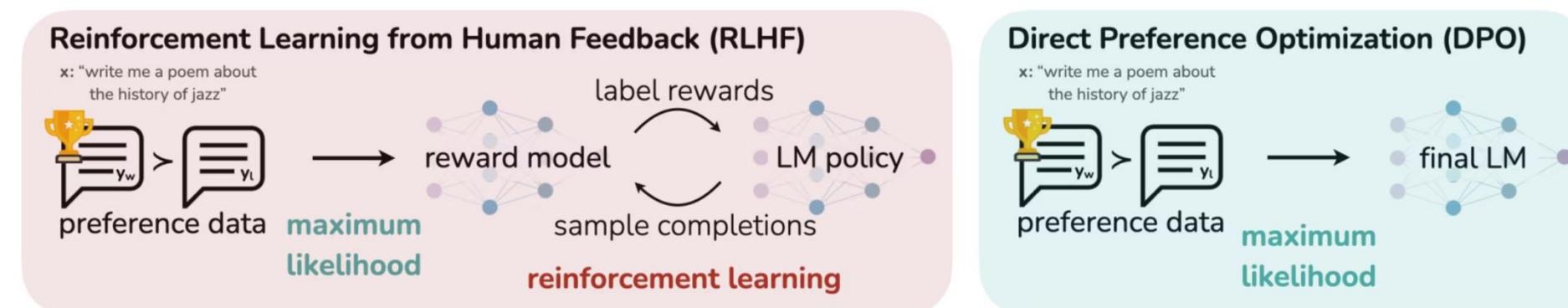
В 2023 году исследователи из Stanford предложили элегантное решение, которое избавляется от необходимости обучать отдельную Reward Model и использовать сложный алгоритм PPO. Это называется **Direct Preference Optimization (DPO)**, и сейчас это де-факто стандарт для большинства open-source моделей.

Ключевая идея DPO

Вопрос: Зачем учить Reward Model как промежуточный шаг, а затем мучиться с нестабильным PPO?

Ответ: Оказывается, мы можем напрямую менять веса основной модели, используя математическую зависимость между оптимальной политикой и функцией награды.

DPO переформулирует задачу обучения с подкреплением как задачу максимизации правдоподобия с особой функцией потерь, основанной на парных предпочтениях.



Источник: Rafailov et al. (2023) "Direct Preference Optimization: Your Language Model is Secretly a Reward Model"

 новый подход

Преимущества DPO над PPO+RM



Скорость обучения

DPO работает значительно быстрее, так как не требует обучения отдельной Reward Model и множества итераций генерации-оценки-обновления.



Стабильность

PPO известен своей нестабильностью и чувствительностью к гиперпараметрам. DPO гораздо более стабилен и предсказуем в обучении.



Меньше памяти

Не нужно держать в памяти четыре модели одновременно (policy, reference, reward, value). Достаточно двух: обучаемой и reference модели.



Простота реализации

DPO – это, по сути, обычное обучение с особой loss-функцией. Не требует сложной инфраструктуры RL с генерацией роллаутов и вычислением advantage.

Llama 4 (апрель'25)

Использует DPO для выравнивания

Mistral Large 3 (декабрь'25)

DPO в основе всех инструктированных версий

GPT-5

DPO используется в моделях без рассуждения

Источник: Rafailov et al. (2023) "Direct Preference Optimization: Your Language Model is Secretly a Reward Model"

Обучение в контексте (In-Context Learning)

Одна из самых удивительных способностей больших языковых моделей – возможность выполнять новые задачи без переобучения весов сети. Достаточно просто показать модели несколько примеров в тексте запроса (промпте), и она «схватывает» паттерн и применяет его к новым данным.

The three settings we explore for in-context learning

Zero-shot

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



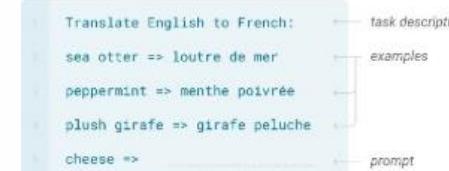
One-shot

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



Few-shot

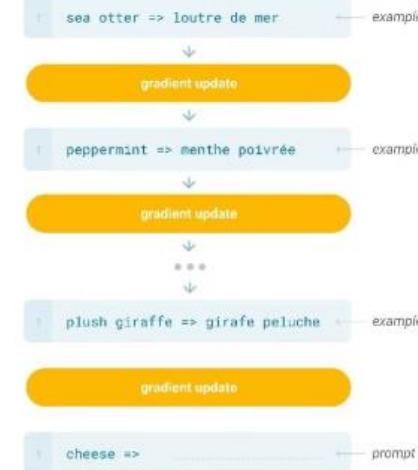
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



Zero-shot

Выполнение задачи без единого примера. Модель полагается только на свои знания, полученные во время предобучения. Пример: «Переведи на французский: Hello world»



One-shot

Предоставление ровно одного примера в промпте. Модель использует этот пример как шаблон для понимания формата ответа. Подходит для простых задач с очевидной структурой.



Few-shot

Несколько примеров в промпте (обычно 10–100). Это золотая середина: модель получает достаточно информации для понимания задачи, но не требуется полное переобучение. Чем больше модель, тем лучше few-shot производительность.

Почему это работает?

Большие модели неявно учатся «учиться» во время предобучения. Они видят миллиарды примеров различных задач в тренировочных данных и формируют метапонимание того, как решать новые задачи по аналогии.

Масштаб имеет значение

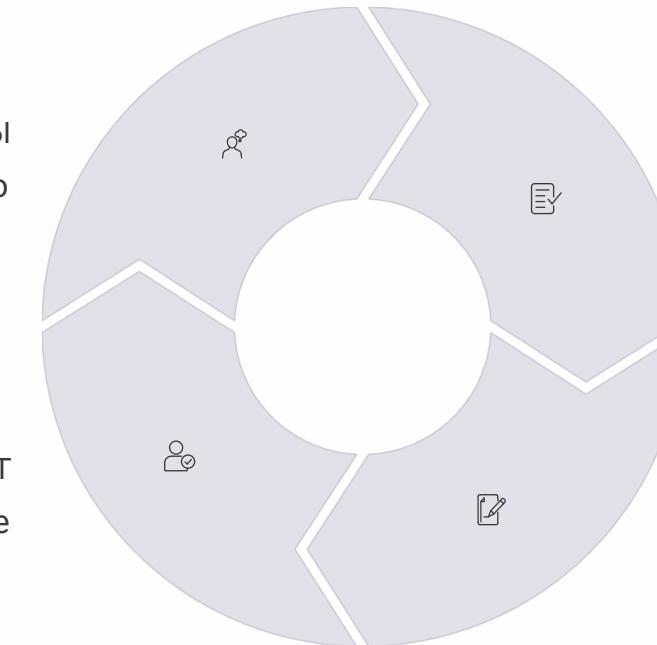
Способность к in-context learning – это эмерджентное свойство, которое проявляется только у достаточно больших моделей. Модели с менее чем 1B параметров показывают слабые результаты.

«Уникальность больших моделей в том, что их не нужно переучивать под каждую задачу. Достаточно просто показать несколько примеров в тексте запроса.»

Рассуждающие модели

2024 год ознаменовал начало новой эры в развитии искусственного интеллекта. OpenAI представила семейство моделей o1 и o3, которые не просто предсказывают следующее слово, а способны к настоящему **многошаговому рассуждению**. Это качественный скачок в способностях ИИ, особенно в сложных задачах, требующих логики и проверки гипотез.

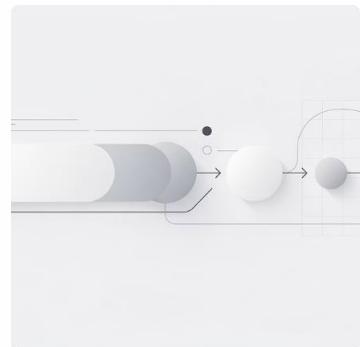
Формулировка гипотезы
Модель генерирует предварительный ответ или подход к решению



Проверка и анализ
Внутренняя цепочка мыслей оценивает корректность подхода

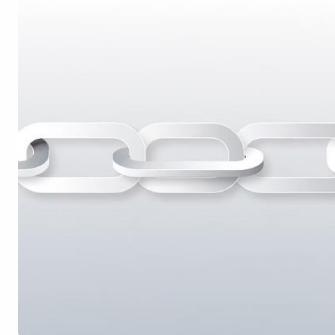
Коррекция ошибок
Если найдены проблемы, модель пересматривает свой подход

Ключевые концепции



Inference-time Compute

Вместо того, чтобы просто выдать ответ мгновенно, модель «думает» — тратит дополнительные вычислительные ресурсы во время генерации, строя скрытые цепочки рассуждений.



Chain of Thought

Модель явно или неявно строит последовательность промежуточных шагов рассуждения: «Сначала мне нужно понять X, затем вычислить Y, потом проверить условие Z».



Self-Verification

Модель проверяет собственные рассуждения, ищет противоречия и ошибки, прежде чем выдать финальный ответ. Это резко снижает количество галлюцинаций.

Рекордные результаты

Мировые модели GPT-5.2, Gemini, Claude показывают результаты уровня PhD:

- в математике (AIME),
- биологии, физике и химии (GPQA),
- кодовых задачах (SWE-Bench)

GPQA Diamond
GPT-5.2 – 90.3%
Gemini 3 Flash – 89.8%
Claude Opus 4.5 - 86.6%

Terminal-Bench
GPT-5.2 – 64.9%
Gemini 3 Pro – 64.7%
Claude Opus 4.5 – 63.1%

AIME
GPT-5.2 – 99%
Gemini 3 Flash – 97%
Claude Opus 4.5 – 91.3%

SWE-Bench
GPT-5.2 – 71.8%
Gemini 3 Pro – 74.2%
Claude Opus 4.5 – 74.4%

GPT-4o

OpenAI o1-preview

User

oyfjdnsdr rtqwainr acxz mynzbhhx → Think step by step

Use the example above to decode:

oyekaijzdf aaptcg suaokybhai ouow aqht
mynznvaatzacdfoulxxz

Let's break this down step by step based on the example:

1. Example given:

- Input: oyfjdnsdr rtqwainr acxz mynzbhhx
- Output: Think step by step

By examining the words:

•

•

•

A possible division could be:

- oyekaijzdf
- aaptcg
- suaokybhai
- ouow
- aqht
- myznvaatzacdfoulxxz

However, to make a meaningful decoding, I would need more context about the transformations or letter shifting that might be involved.

Could you provide any additional decoding rules or transformations used in this cipher?

oyfjdnsdr rtqwainr acxz mynzbhhx → Think step by step

Use the example above to decode:

oyekaijzdf aaptcg suaokybhai ouow aqht
mynznvaatzacdfoulxxz

Thought for 5 seconds ▾

To decode the given ciphertext using the example provided, we'll use the pattern identified in the example.

Ciphertext Example:

Unknown component type: {componentType}

- 5. tz → (20+26)/2 = 23 → W
- 6. ac → (1+3)/2 = 2 → B
- 7. df → (4+6)/2 = 5 → E
- 8. ou → (15+21)/2 = 18 → R
- 9. ix → (12+24)/2 = 18 → R
- 10. xz → (24+26)/2 = 25 → Y

3. Decoded word: STRAWBERRY

Final Decoded Message:

Unknown component type: {componentType}

Answer:

THERE ARE THREE R'S IN STRAWBERRY

Современные тренды и экономика генеративного ИИ

Генеративный ИИ развивается сразу в нескольких направлениях. Модели становятся более универсальными, эффективными и дорогими одновременно. Давайте рассмотрим ключевые тренды, которые определяют индустрию сегодня.



Мультимодальность

Этап чисто текстовых моделей заканчивается. GPT-4o, Gemini, Claude 3 свободно работают с изображениями, аудио и видео. Sora 2 генерирует физически достоверное видео с синхронным звуком длительностью до минуты.



Эффективность

Малые модели догоняют гигантов. Phi-3 (3.8B параметров) от Microsoft показывает результаты моделей в 10 раз больше. Появляются специализированные чипы и техники квантизации, делающие ИИ доступнее.



Агенты и инструменты

Модели учатся использовать компьютер как человек. Anthropic Computer Use позволяет Claude управлять браузером и приложениями. Появляются AI-агенты, способные выполнять сложные многоэтапные задачи автономно.



Цена успеха

Обучение передовых моделей требует астрономических инвестиций. Llama 3.1 405B обошлась Meta примерно в \$170 миллионов только на вычисления. Энергопотребление сопоставимо с небольшими городами.

Заключение: Генеративные модели как новая операционная система

Мы прошли долгий путь от первых экспериментов с GAN до современных рассуждающих мультимодальных систем. Генеративные модели эволюционировали из узкоспециализированных инструментов в универсальные платформы, способные выполнять широчайший спектр интеллектуальных задач.

Генеративные модели — это новое ядро

Все чаще эксперты говорят о больших языковых моделях не как о приложениях, а как о **новой операционной системе** для работы с информацией. Подобно тому, как Windows или Linux являются платформой для запуска программ, LLM становятся платформой для запуска когнитивных задач.

- **Универсальный интерфейс:** Естественный язык как API ко всем функциям
- **Расширяемость:** Плагины, инструменты, fine-tuning под специфические домены
- **Экосистема:** Тысячи приложений строятся поверх базовых моделей

Что дальше?



Спасибо за внимание! На следующей лекции мы разберем, как адаптировать LLM под ваши конкретные задачи и требования.