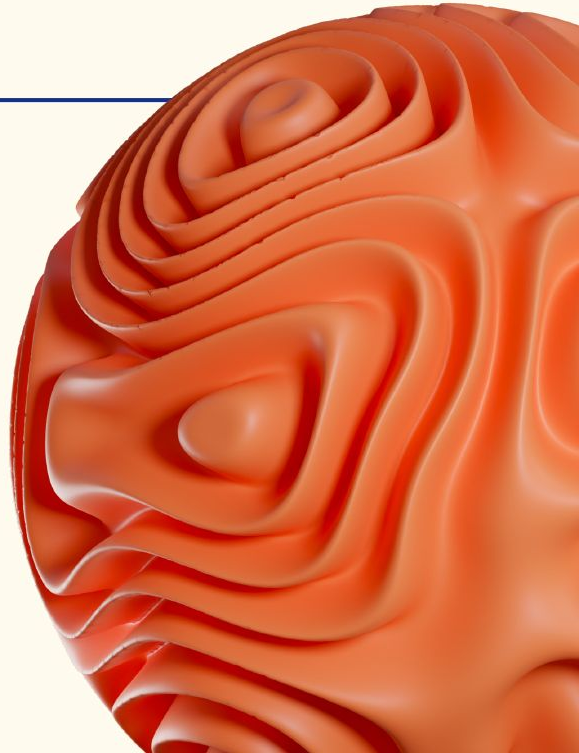
Задачи генерации в NLP

МФТИ × SKILLFACTORY

Задача суммаризации текста



Сидоров Никита, SberDevices

Цели занятия

- Изучить большие языковые модели (LLM)
- Рассмотреть примеры наиболее популярных моделей
- Рассмотреть методы Parameter Efficient Fine-Tuning`а моделей
- Познакомиться с методами борьбы с галюционированием генеративных моделей
- Рассмотреть альтернативные трансформерным архитектуры больших языковых моделей

План занятия

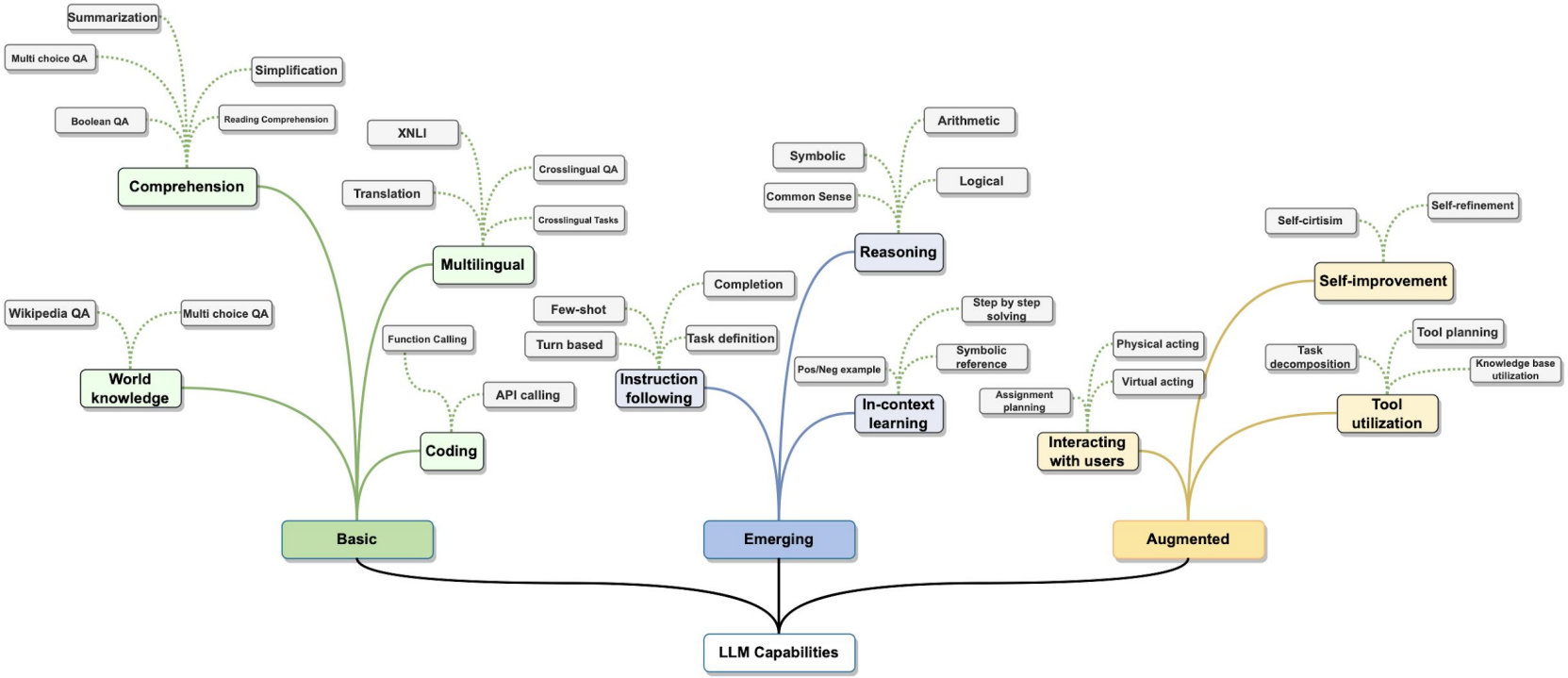
1. Большие языковые модели
2. PEFT методы обучения моделей
3. Retrieval-Augmented generation
4. Альтернативные архитектуры больших языковых моделей

Большие языковые модели(LLM)

В современном виде, это нейронные сети, предобученные на большом объеме информации и обладающие способностью к глубинному пониманию языка и продвинутой генерации.

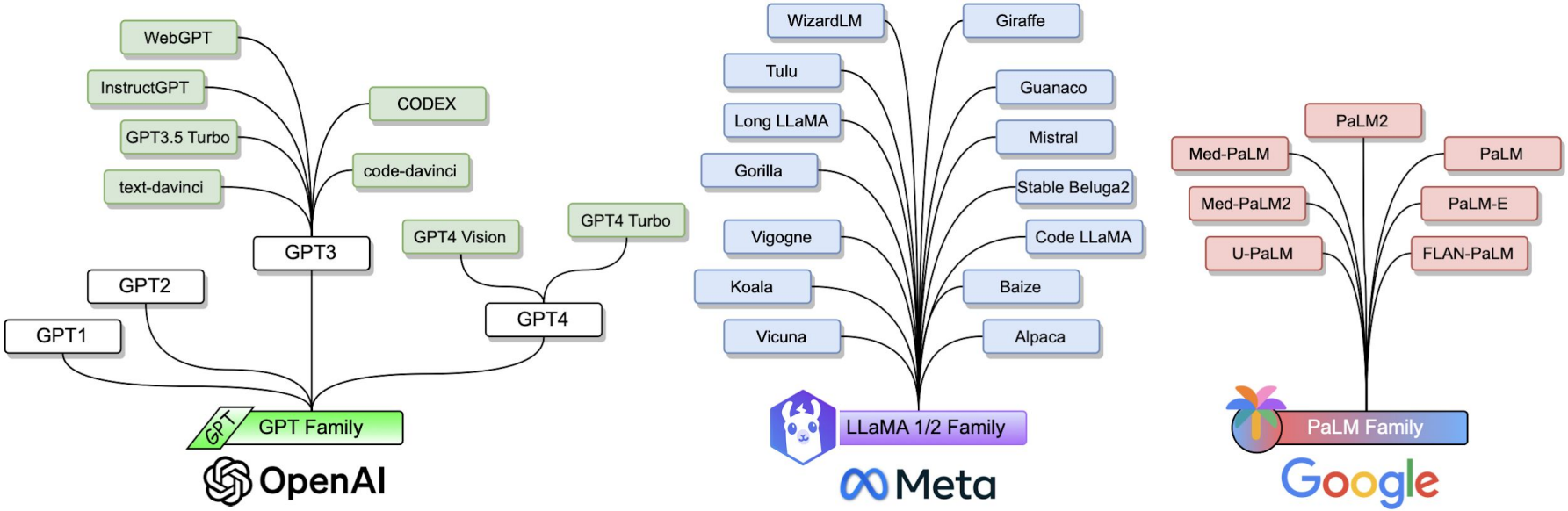
Помимо этого, они обладают новыми способностями, которые не были доступны в моделях меньшего размера или предыдущих поколений.

Возможности LLM



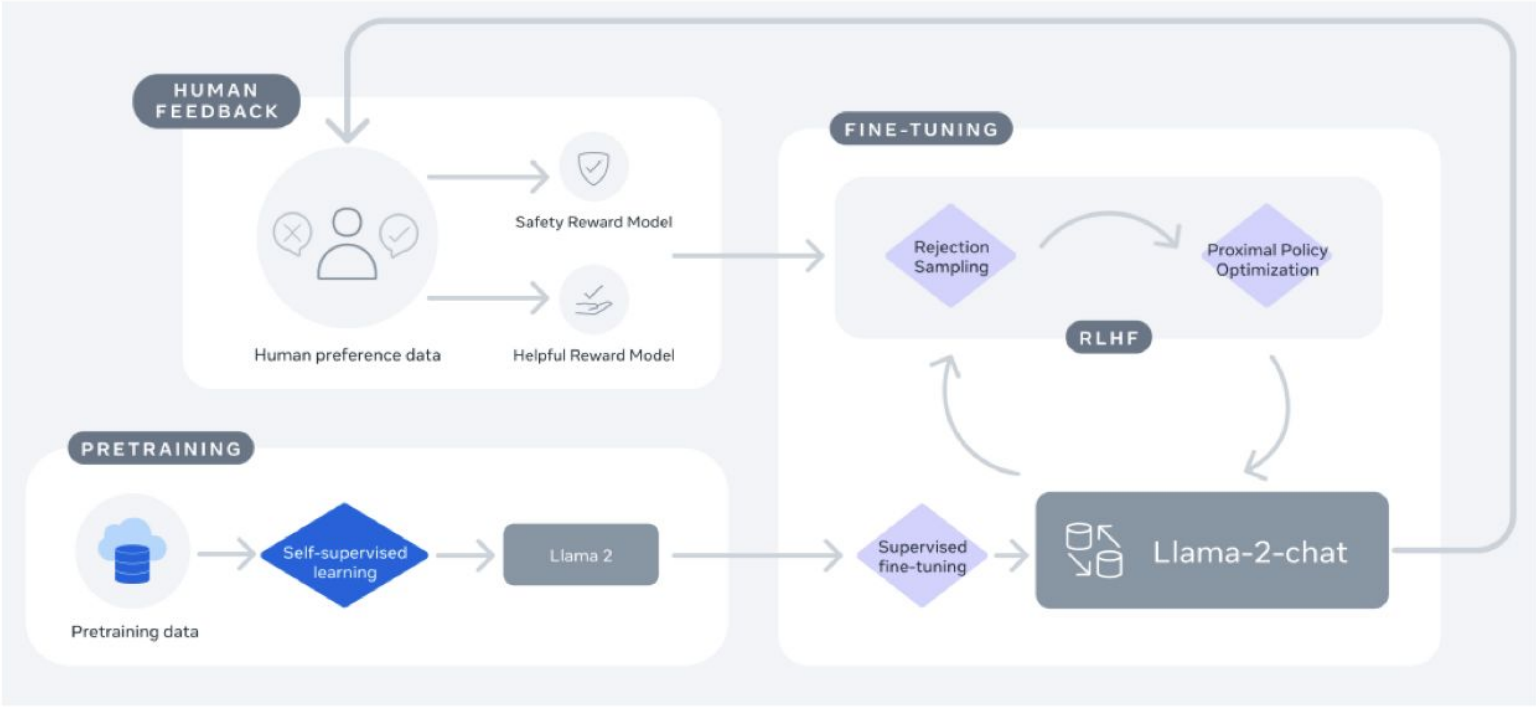
https://arxiv.org/pdf/2402.06196.pdf

Семейства современных LLM



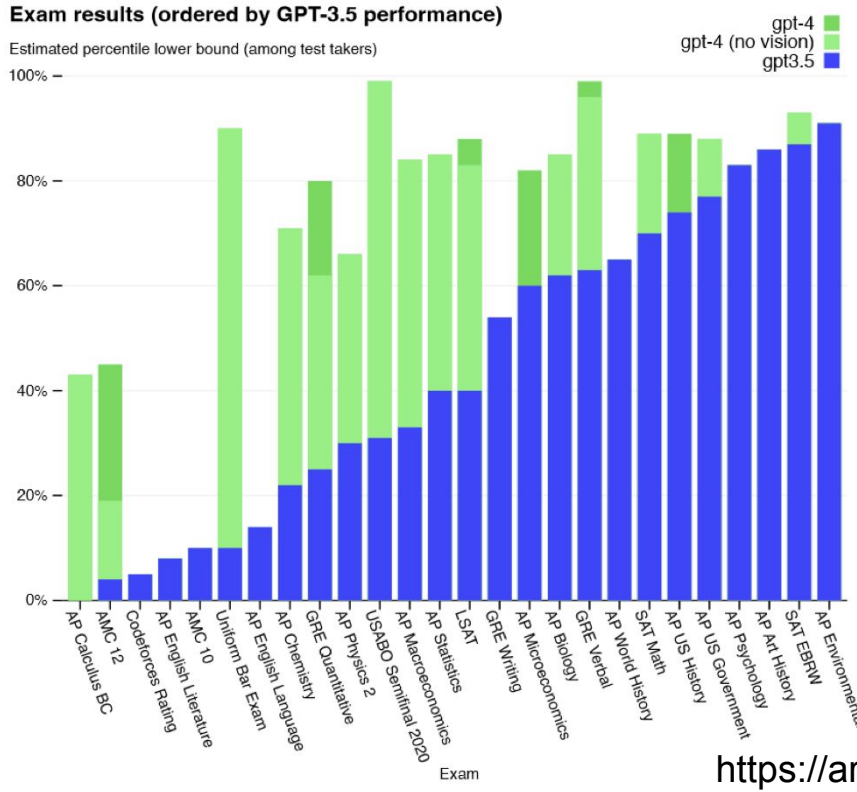
https://arxiv.org/pdf/2402.06196.pdf

От LLM к инстинктивным моделям



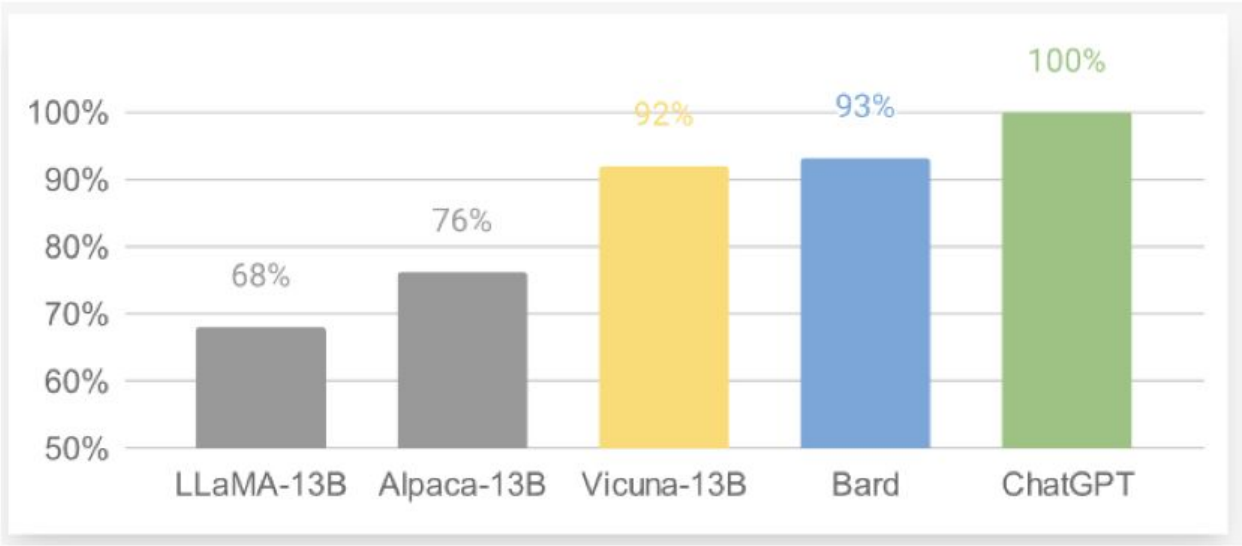
https://arxiv.org/pdf/2402.06196.pdf

Способности инструктивных моделей



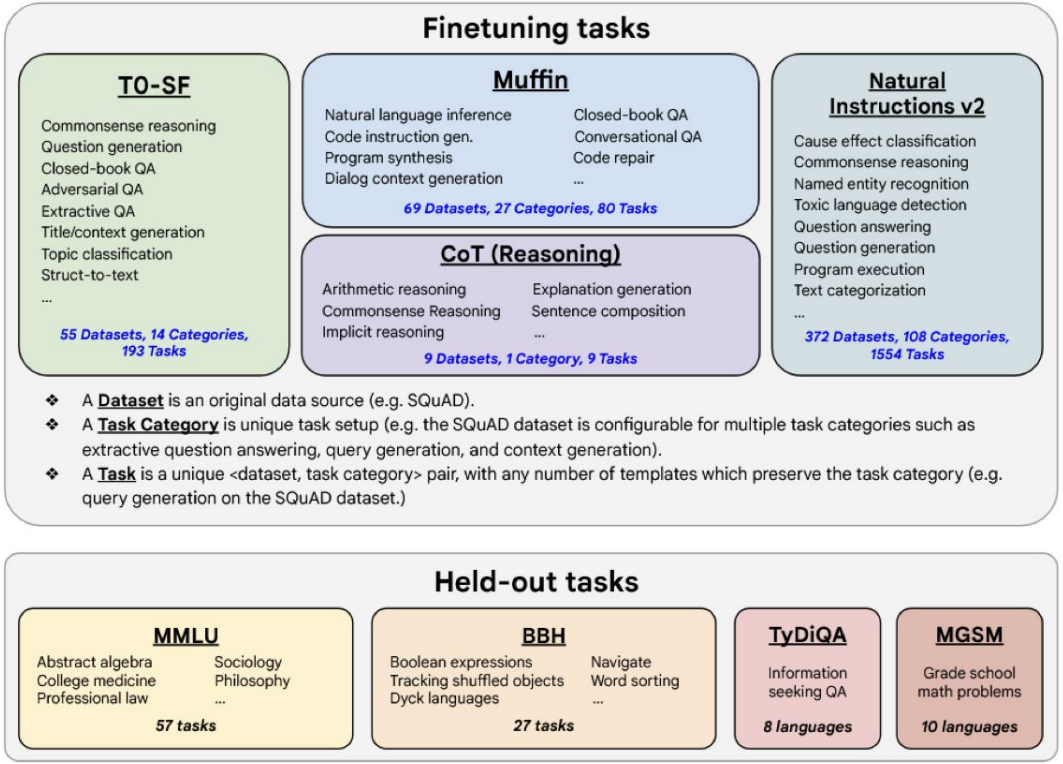
https://arxiv.org/pdf/2402.06196.pdf

Способности инструктивных моделей



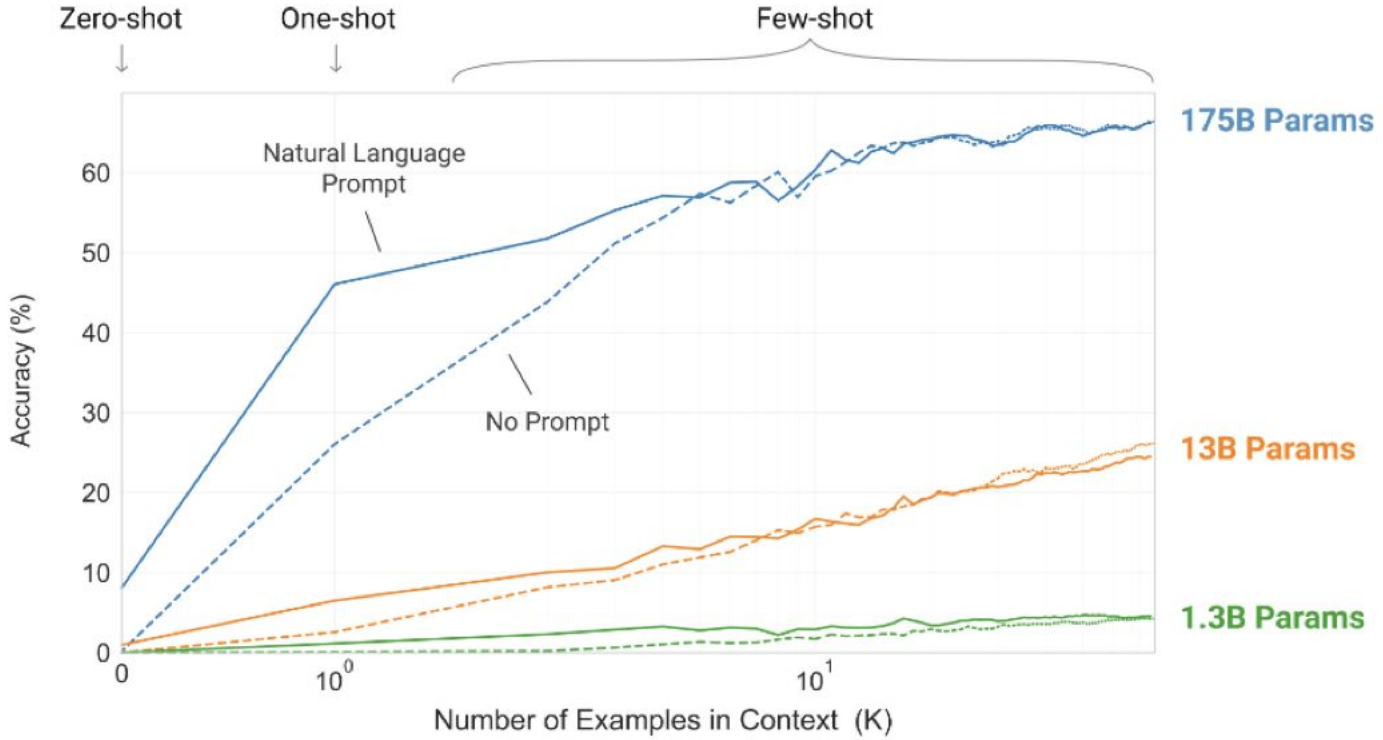
https://arxiv.org/pdf/2402.06196.pdf

Альтернативный подход к обучению инструктивных моделей - FLAN

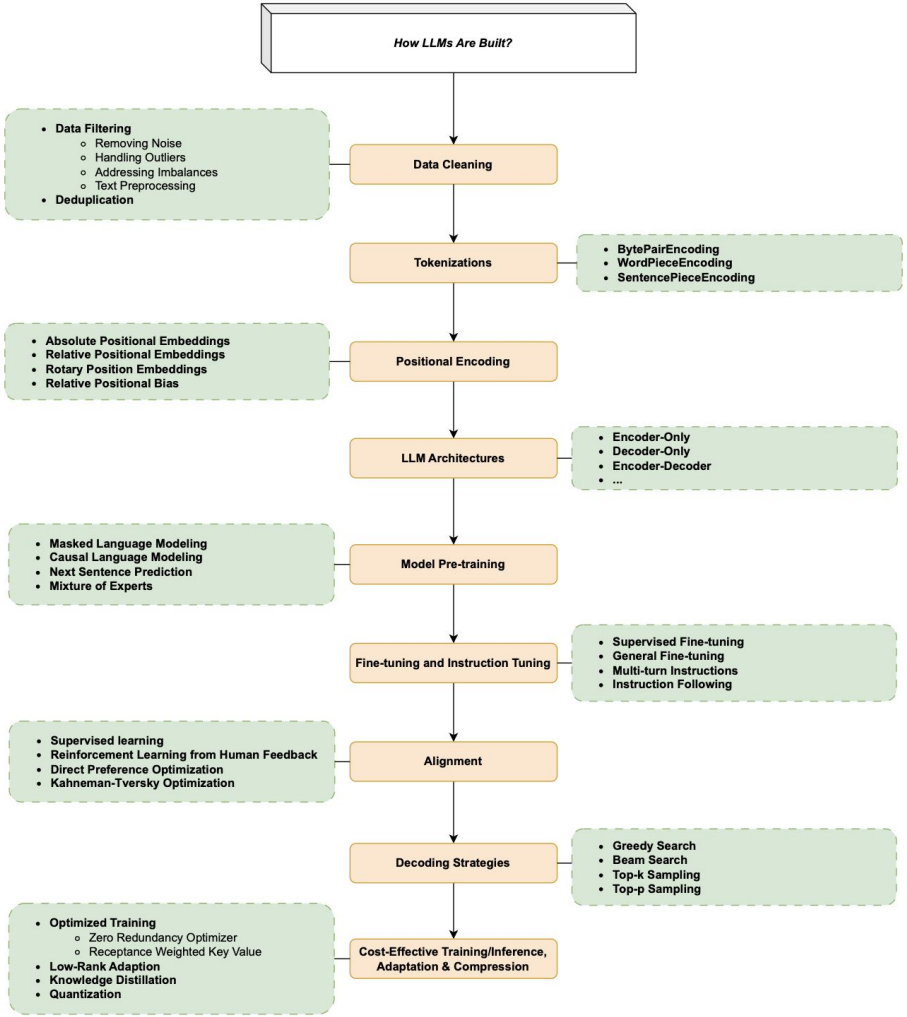


https://arxiv.org/pdf/2402.06196.pdf

Влияние промптинга на инструктивные модели

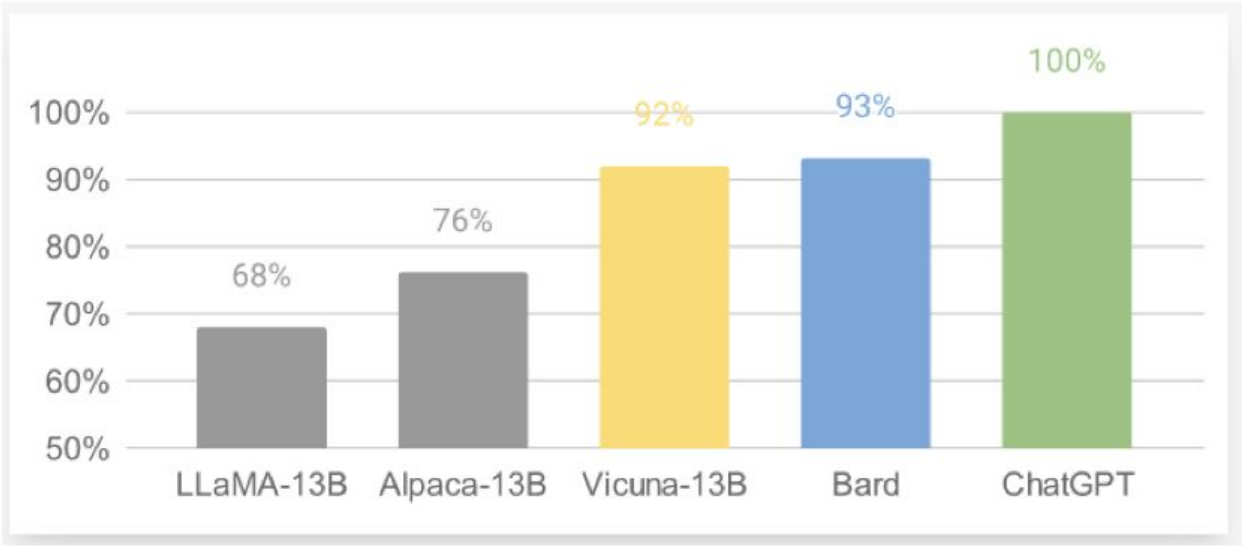


https://arxiv.org/pdf/2402.06196.pdf



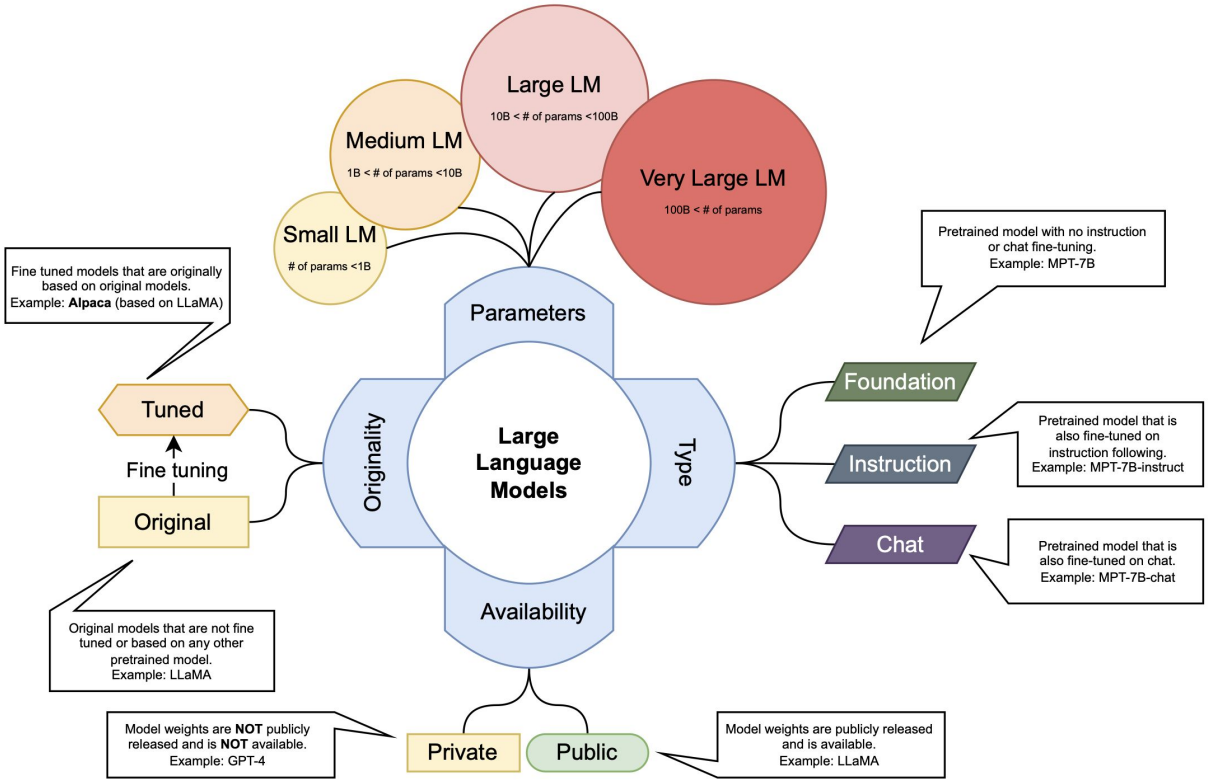
https://arxiv.org/pdf/2402.06196.pdf

Способности инструктивных моделей



https://arxiv.org/pdf/2402.06196.pdf

Категоризация LLM



https://arxiv.org/pdf/2402.06196.pdf

Ограничения при работе с LLM

- У LLM нет памяти/состояния
- В общем случае, генерация стохастична
- Они обучаются на устаревающей информации и не имеют доступа к внешним источникам данных
- Они очень большие, а значит требуют большие накладные расходы
- Они галлюцинирует

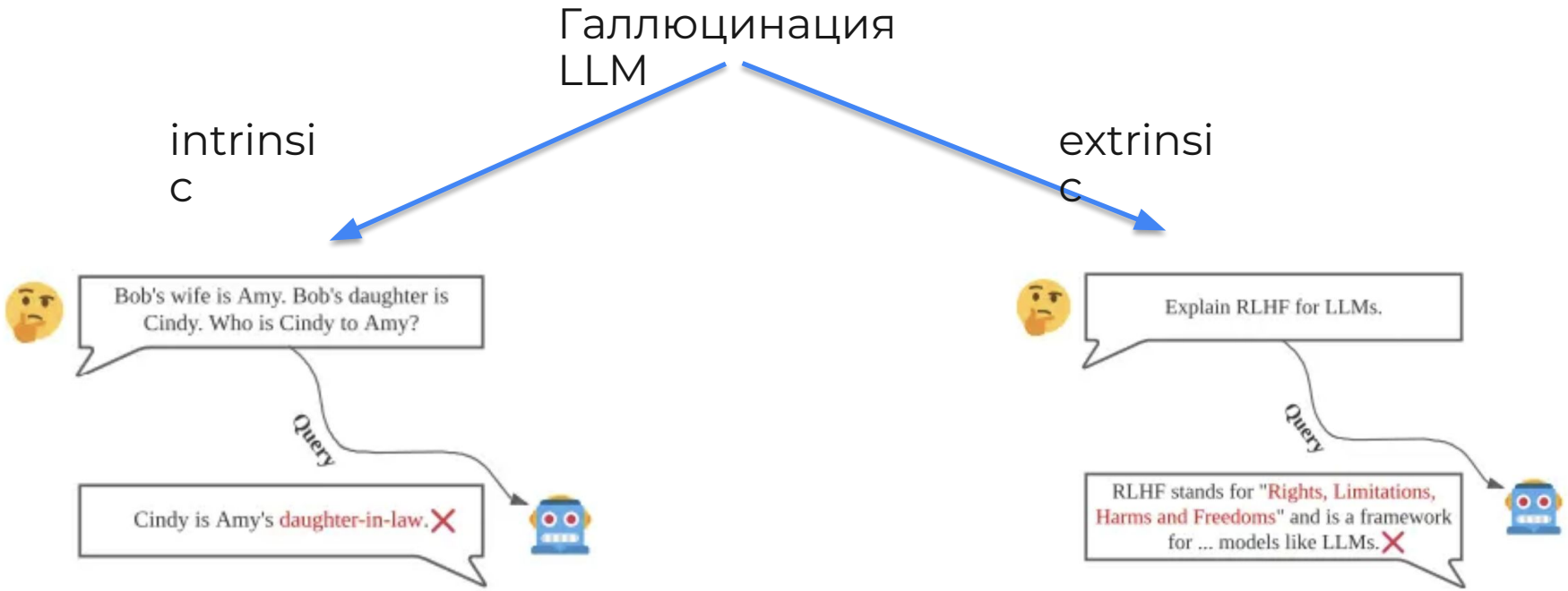
Галлюцинации LLM

Галлюцинация LLM - создание контента, который является бессмысленным или не соответствует предоставленному источнику.

Выделяют 2 вида галлюцинаций:

- Intrinsic (генерация не соответствует или противоречит исходному тексту)
- Extrinsic (сгенерированный вывод, не может быть проверен на основе исходного содержимого (т.е. вывод, который не может быть ни подтвержден, ни опровергнут источником)

Виды галлюцинаций LLM



RAG

Модели предобучаются на данных, которые постепенно устаревают. Также они не имеют доступа к частной или специфичной информации, которая требуется для конкретной задачи.

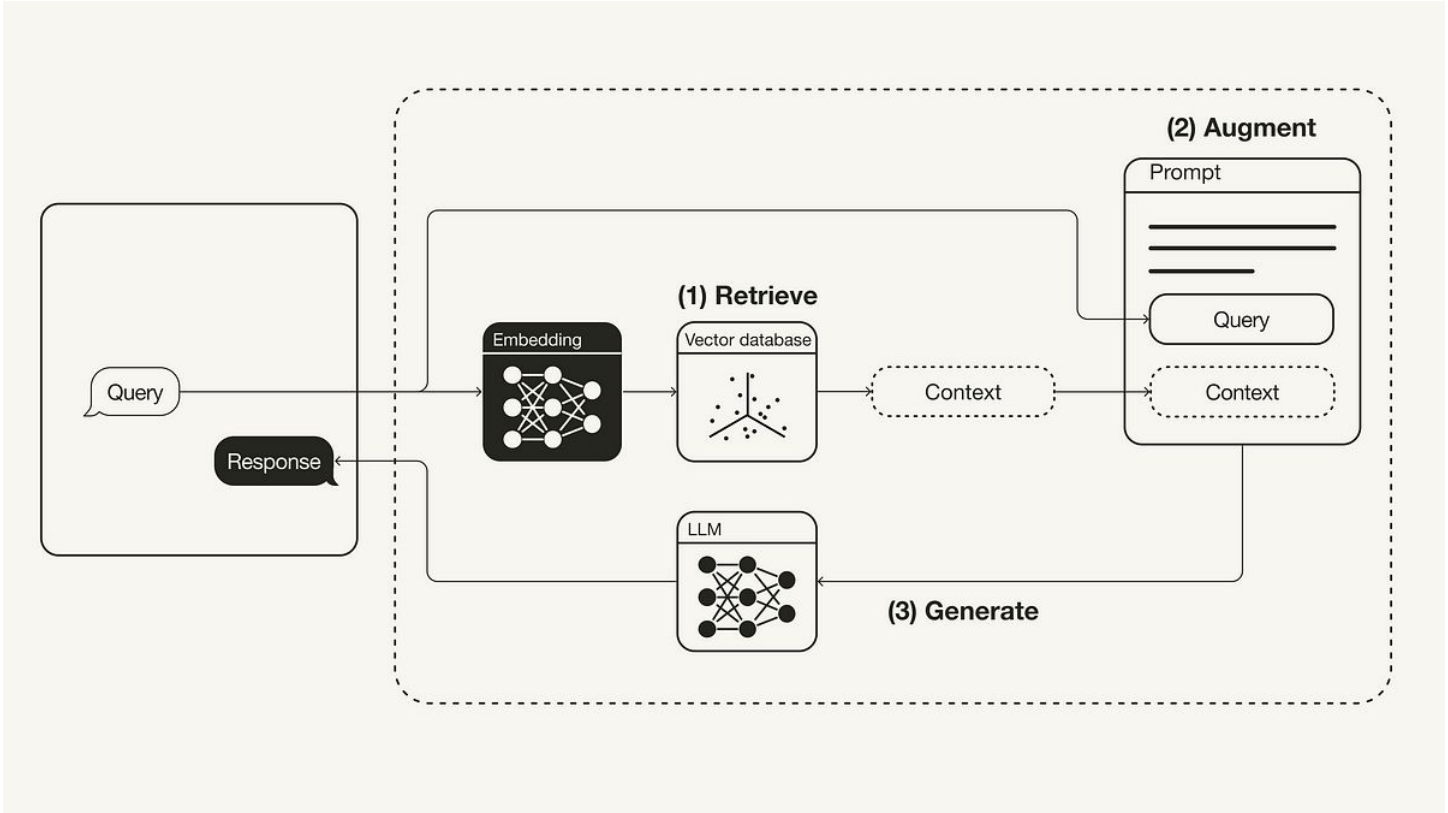
Для таких сценариев идеально подходит retrieval augmented generation(RAG).

RAG

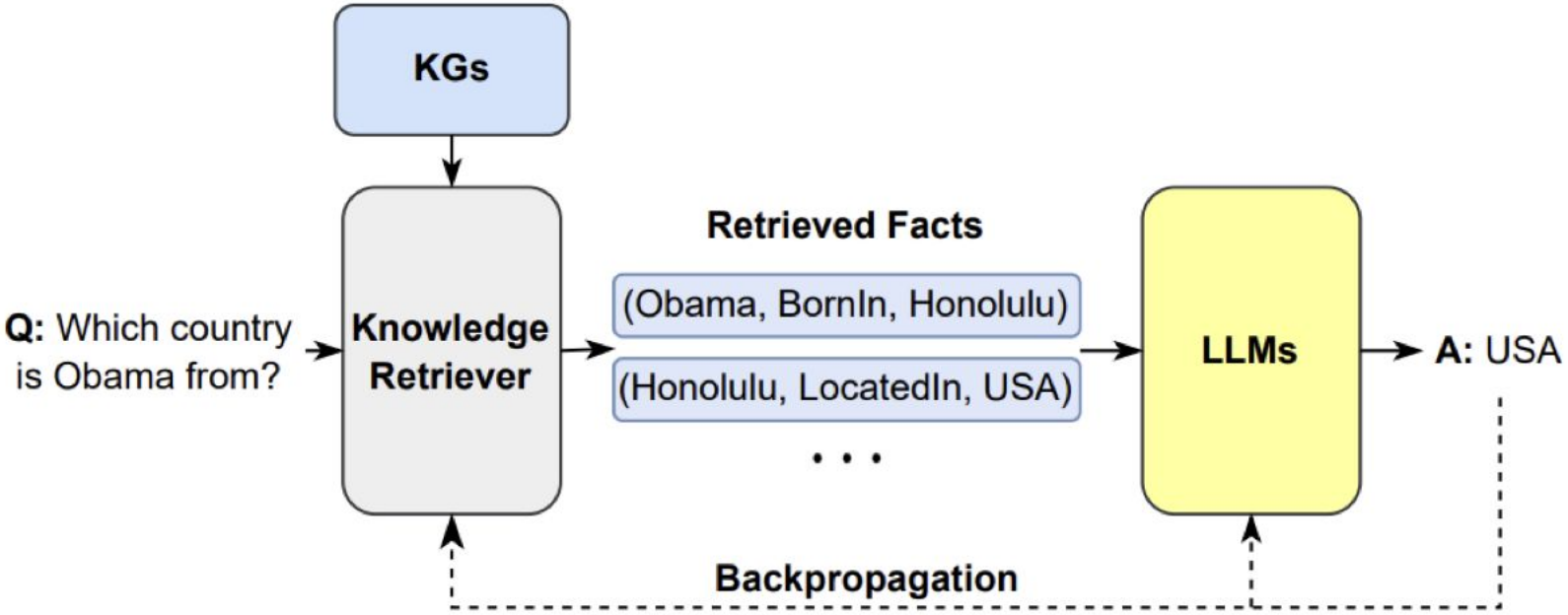
Чтобы модель отвечала на запрос по неизвестной или специализированной теме мы можем сделать инъекцию релевантной информации в запрос и подать такой расширенный промыт на вход в языковую модель.

Важно понимать, что при наличии контекстного окна достаточной вместимости, мы можем положить в контекст весь документ, по которому осуществляется запрос, однако этот случай уже не относится к RAG.

RAG

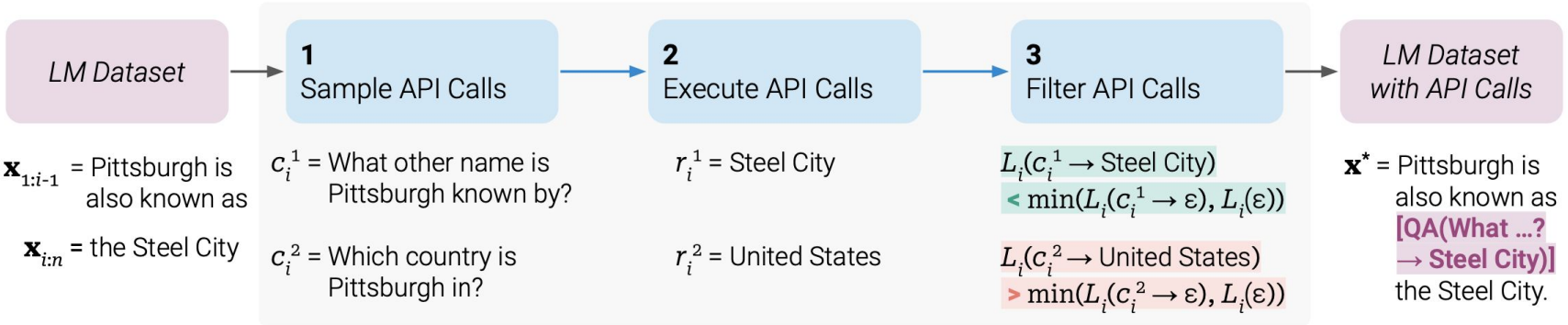

(2) Augment
Prompt
Query
Context
(1) Retrieve
Embedding
Vector database
Context
Query
Response
LLM
(3) Generate

RAG


KGs
Q: Which country is Obama from?
Knowledge Retriever
Retrieved Facts
(Obama, BornIn, Honolulu)
(Honolulu, LocatedIn, USA)
• • •
LLMs
A: USA
Backpropagation

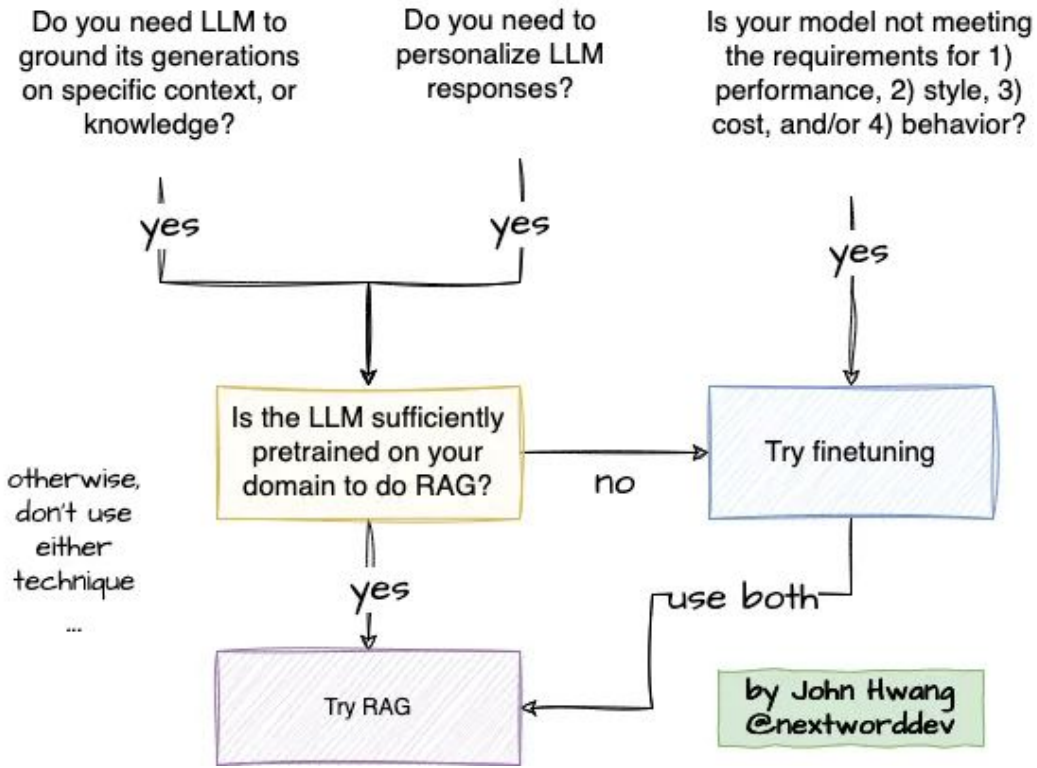
ToolFormer

Модели способны исходя из запроса вызывать сторонние инструменты. В данном контексте - это внешние функции или сервисы, которые могут использовать LLM



https://arxiv.org/pdf/2302.04761.pdf

Рецепт использования RAG



Ограничения по использованию RAG

Не забывайте, что использование техники RAG влечет за собой дополнительные накладные расходы на:

- Скорость работы вашей системы (механизм retrieve + расширение контекста)
- Потребляемые ресурсы и память (хранение БД для retrieve + отдельные сервисы)

Посчитаем потребление памяти для FT модели OPT (1.3B)

		OPT-1.3B, 16-bit float, seq 512
cuDNN and CUDA		~1Gb
Model weights	size(float) * N	2.6Gb
Gradients	size(float) * $N_{trainable}$	2.6Gb
Hidden states	*~size(float) L (20 h seq + 3 seq^2)*	1Gb per example
Optimizer states	2 * size(float) * $N_{trainable}$	5.2Gb
(maybe) fp32 copy of the gradients	4 * $N_{trainable}$	10.2Gb

Расчитаное значение: 12.4Gb, Реальное: 11.0Gb *(после empty_cache)*

Посчитаем потребление памяти для FT модели OPT (1.3B)

Обучим только 0.2M параметров

		OPT-1.3B, 16-bit float, seq 512
cuDNN and CUDA		~1Gb
Model weights	size(float) * N	2.6Gb
Gradients	size(float) * $N_{trainable}$	**0.4Mb**
Hidden states	*~size(float) L (20 h seq + 3 seq^2)*	1Gb per example
Optimizer states	2 * size(float) * $N_{trainable}$	0.8Mb
(maybe) fp32 copy of the gradients	4 * $N_{trainable}$	**1.6Mb**

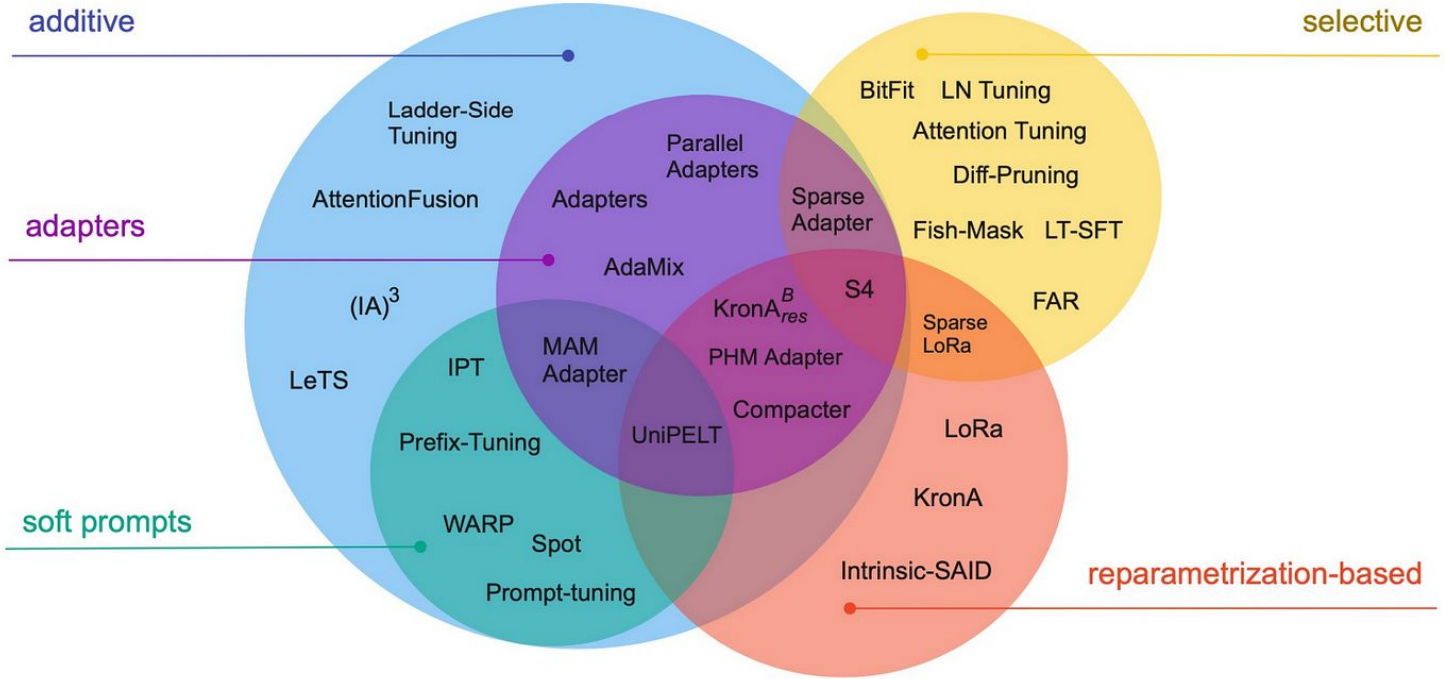
Расчитаное значение: 4.6Gb, Реальное: 5.7Gb *(после empty_cache)*

PEFT методы

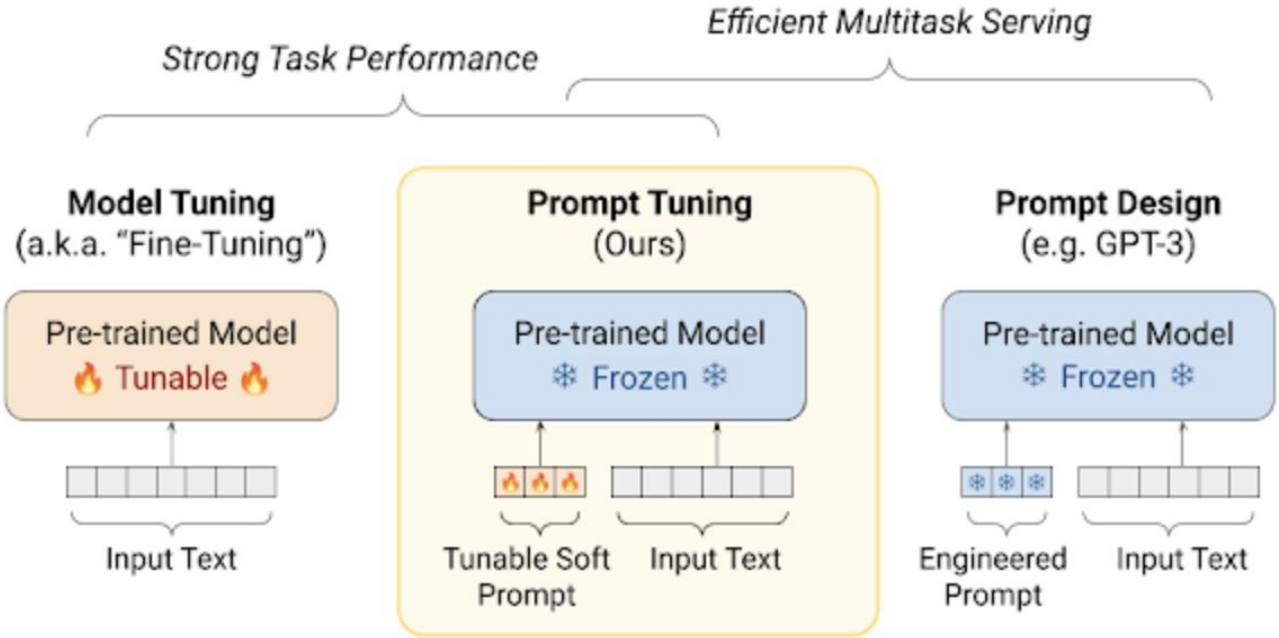
Parameter Efficient
Finu-Tuning

PEFT методы позволяют обучать только небольшое количество (дополнительных) параметров модели (<1%), что значительно снижает затраты на вычисления и хранение, обеспечивая при этом производительность, сопоставимую с полным Fine-Tuning`ом модели.

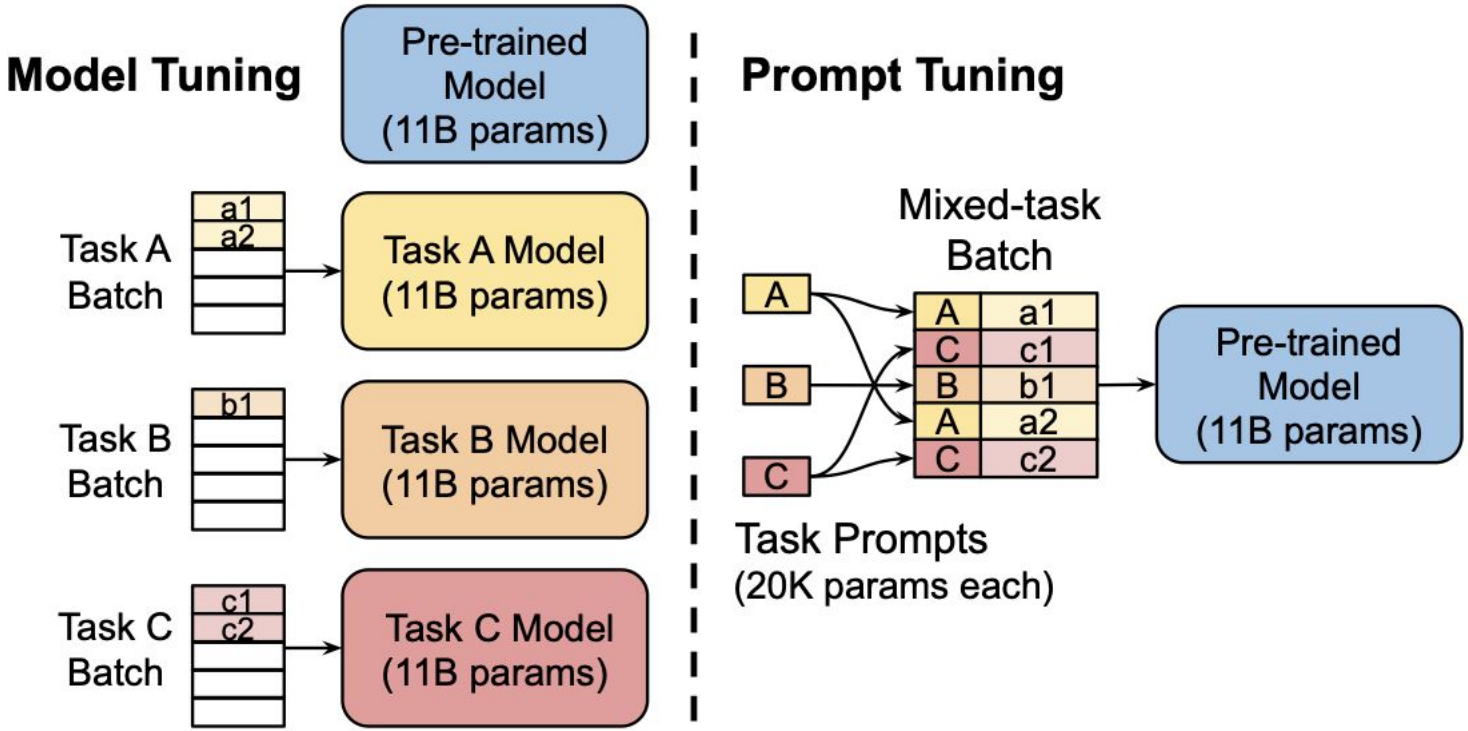
Семейство PEFT-методов



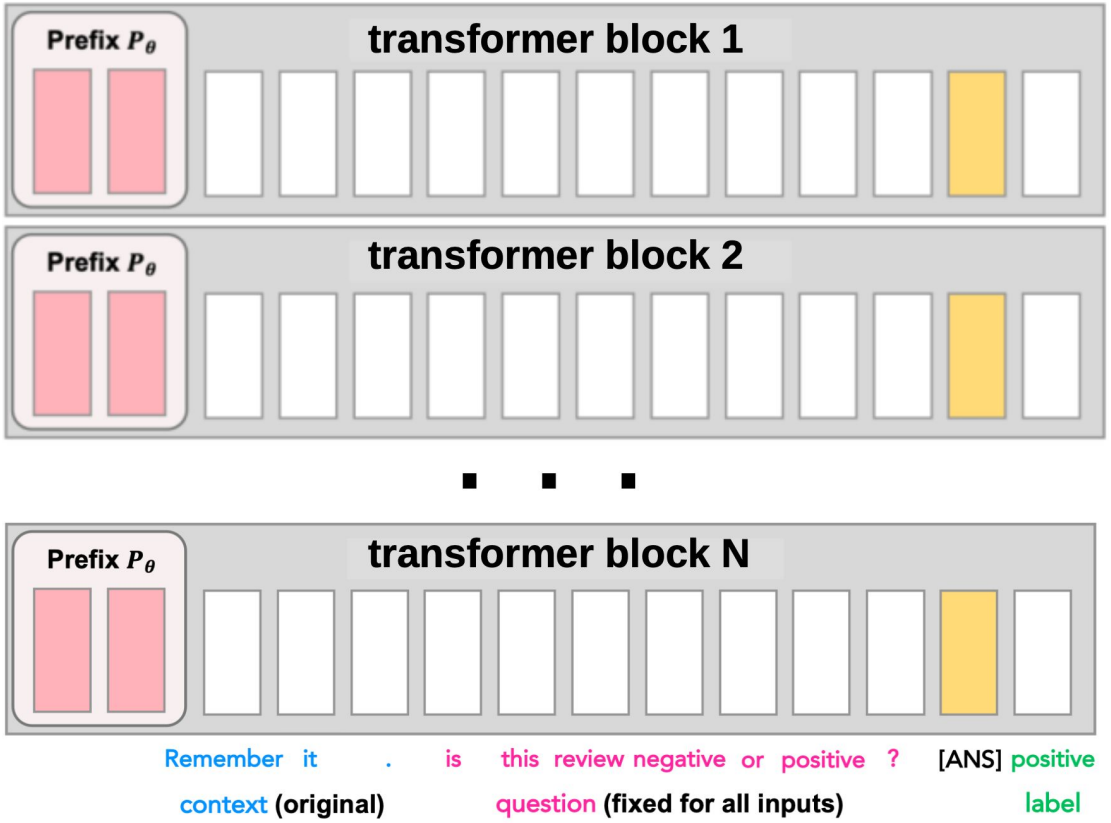
Prompt-Tuning (Additive)



Prompt-Tuning (Additive)



Prefix-Tuning (Additive)

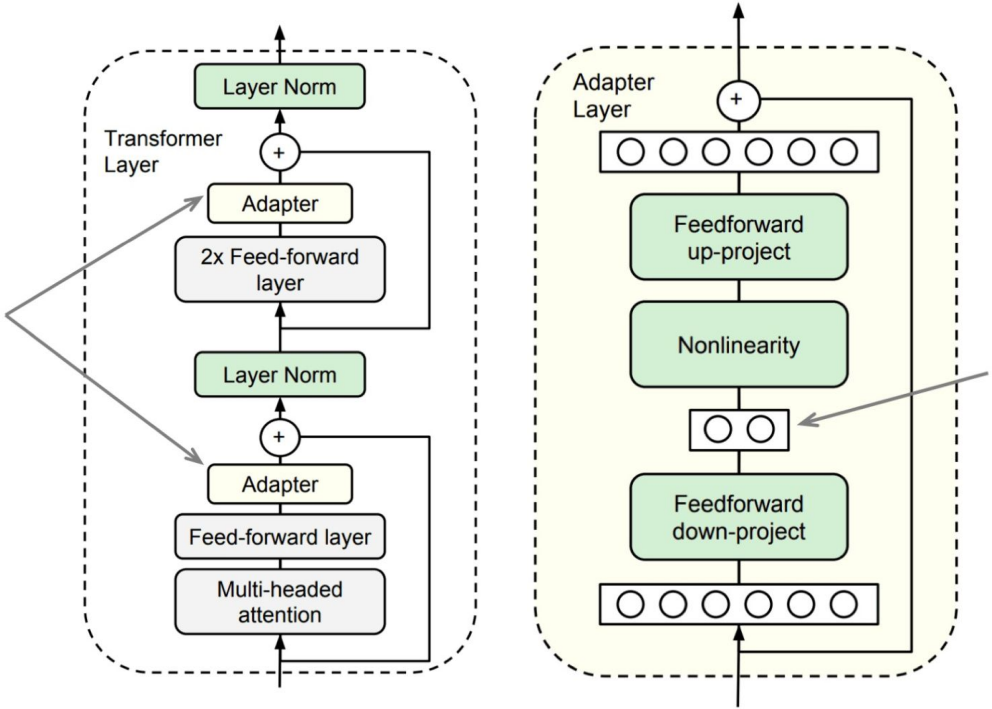


https://arxiv.org/abs/2101.00190 https://arxiv.org/abs/2110.07602

Adapters (Additive)

Обучаем только небольшие подсети

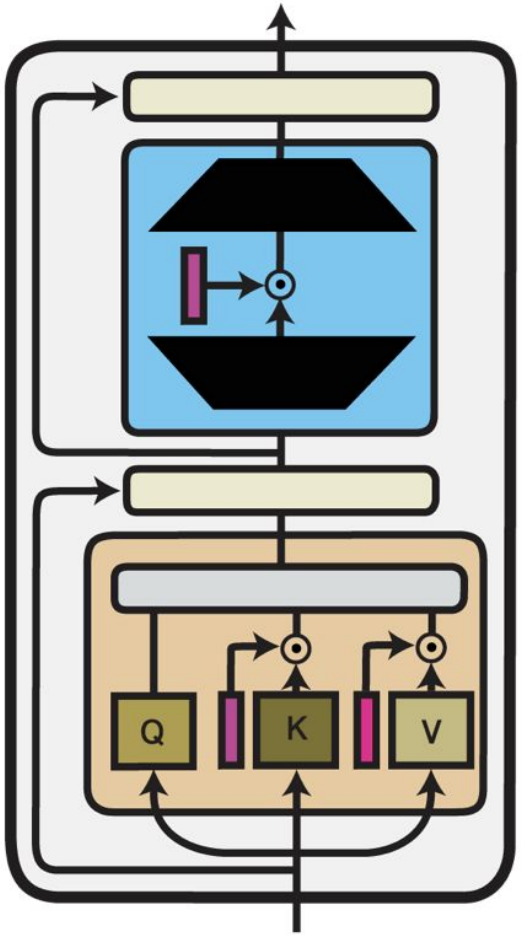
Only these are trained, everything else is fixed and is the same for all tasks



Small hidden size, i.e. an adaptor has only a few parameters (which is good!)

IA3 (Additive)

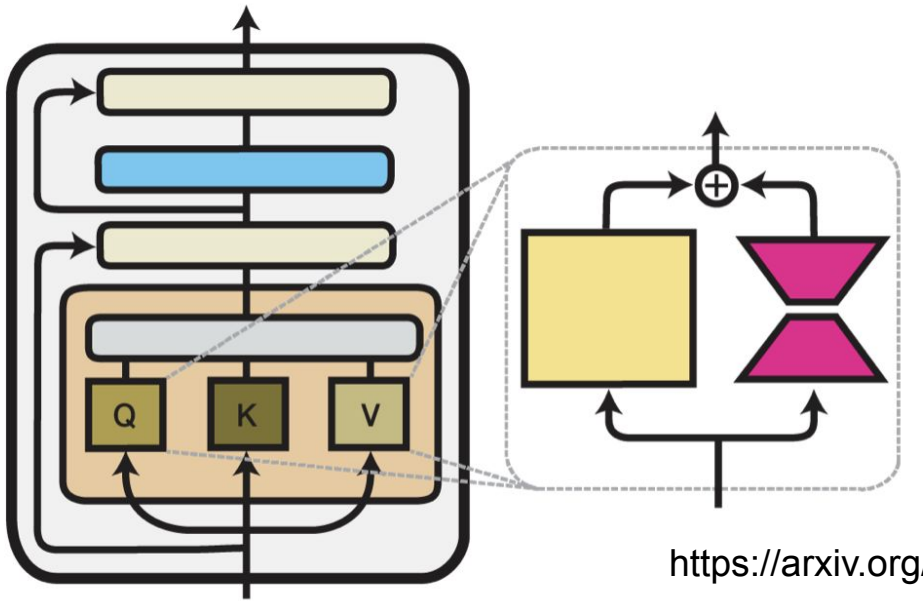
Идея: Сделаем rescale векторов key и value и скрытого состояния в FFN слое



https://arxiv.org/pdf/2205.05638.pdf

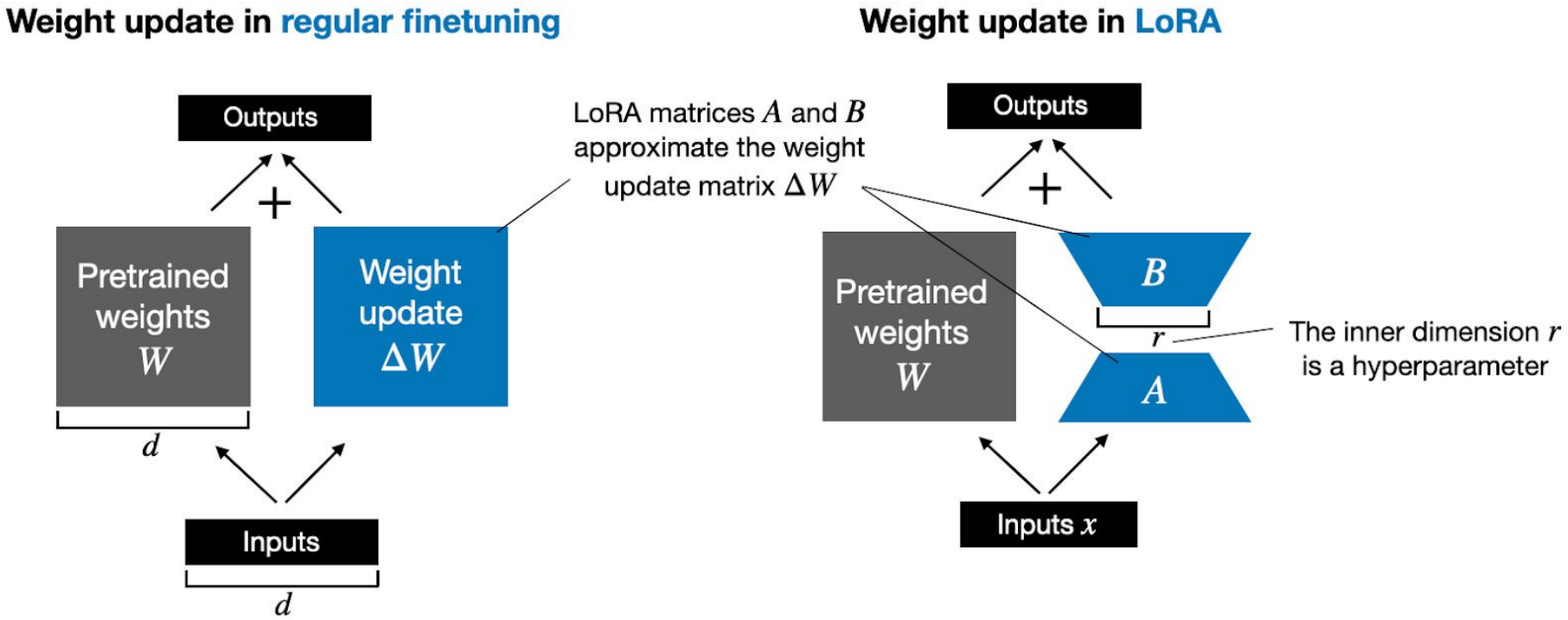
LoRA (Reparametrization)

Идея: Обновление параметров для любой матрицы весов в LoRA производится за счет произведения двух матриц низкого ранга (



https://arxiv.org/pdf/2106.09685.pdf

LoRA (Reparametrization)



https://arxiv.org/pdf/2106.09685.pdf

Сравнение методов

Method	Type	Storage	Memory	Backprop	Inference overhead
Adapters (Houlsby et al., 2019)	A	yes	yes	no	Extra FFN
AdaMix (Wang et al., 2022)	A	yes	yes	no	Extra FFN
SparseAdapter (He et al., 2022b)	AS	yes	yes	no	Extra FFN
Cross-Attn tuning (Gheini et al., 2021)	S	yes	yes	no	No overhead
BitFit (Ben-Zaken et al., 2021)	S	yes	yes	no	No overhead
DiffPruning (Guo et al., 2020)	S	yes	no	no	No overhead
Fish-Mask (Sung et al., 2021)	S	yes	maybe[5]	no	No overhead
LT-SFT (Ansell et al., 2022)	S	yes	maybe[5]	no	No overhead
Prompt Tuning (Lester et al., 2021)	A	yes	yes	no	Extra input
Prefix-Tuning (Li and Liang, 2021)	A	yes	yes	no	Extra input
Spot (Vu et al., 2021)	A	yes	yes	no	Extra input
IPT (Qin et al., 2021)	A	yes	yes	no	Extra FFN and input
MAM Adapter (He et al., 2022a)	A	yes	yes	no	Extra FFN and input
Parallel Adapter (He et al., 2022a)	A	yes	yes	no	Extra FFN
Intrinsinc SAID (Aghajanyan et al., 2020)	R	no	no	no	No overhead
LoRa (Hu et al., 2022)	R	yes	yes	no	No overhead
UniPELT (Mao et al., 2021)	AR	yes	yes	no	Extra FFN and input
Compacter (Karimi Mahabadi et al., 2021)	AR	yes	yes	no	Extra FFN
PHM Adapter (Karimi Mahabadi et al., 2021)	AR	yes	yes	no	Extra FFN
KronA (Edalati et al., 2022)	R	yes	yes	no	No overhead
KronA^{B}_{res} (Edalati et al., 2022)	AR	yes	yes	no	Extra linear layer
$(\text{IA})^3$ (Liu et al., 2022)	A	yes	yes	no	Extra gating
Attention Fusion (Cao et al., 2022)	A	yes	yes	yes	Extra decoder
LeTS (Fu et al., 2021)	A	yes	yes	yes	Extra FFN
Ladder Side-Tuning (Sung et al., 2022)	A	yes	yes	yes	Extra decoder
FAR (Vucetic et al., 2022)	S	yes	maybe[6]	no	No overhead
S4-model (Chen et al., 2023)	ARS	yes	yes	no	Extra FFN and input

https://arxiv.org/abs/2303.15647

Сравнение методов

Method	% Trainable parameters	% Changed parameters	Evaluated on <1B	Evaluated on <20B	Evaluated on >20B
Adapters (Houlsby et al., 2019)	0.1 - 6	0.1 - 6	yes	yes	yes
AdaMix (Wang et al., 2022)	0.1 - 0.2	0.1 - 0.2	yes	no	no
SparseAdapter (He et al., 2022b)	2.0	2.0	yes	no	no
BitFit (Ben-Zaken et al., 2021)	0.05 - 0.1	0.05 - 0.1	yes	yes	yes
DiffPruning (Guo et al., 2020)	200	**0.5**	yes	no	no
Fish-Mask (Sung et al., 2021)	0.01 - 0.5	0.01 - 0.5	yes	yes	no
Prompt Tuning (Lester et al., 2021)	0.1	0.1	yes	yes	yes
Prefix-Tuning (Li and Liang, 2021)	0.1 - 4.0	0.1 - 4.0	yes	yes	yes
IPT (Qin et al., 2021)	56.0	56.0	yes	no	no
MAM Adapter (He et al., 2022a)	0.5	0.5	yes	no	no
Parallel Adapter (He et al., 2022a)	0.5	0.5	yes	no	no
Intrinsinc SAID (Aghajanyan et al., 2020)	0.001 - 0.1	**∼0.1 or 100**	yes	yes	no
LoRa (Hu et al., 2022)	0.01 - 0.5	**∼0.5 or ∼30**	yes	yes	yes
UniPELT (Mao et al., 2021)	1.0	1.0	yes	no	no
Compacter (Karimi Mahabadi et al., 2021)	0.05-0.07	**∼0.07 or ∼0.1**	yes	yes	no
PHM Adapter (Karimi Mahabadi et al., 2021)	0.2	**∼0.2 or ∼1.0**	yes	no	no
KronA (Edalati et al., 2022)	0.07	**∼0.07 or ∼30.0**	yes	no	no
KronA^{B}_{res} (Edalati et al., 2022)	0.07	**∼0.07 or ∼1.0**	yes	no	no
$(\text{IA})^3$ (Liu et al., 2022)	0.02	0.02	no	yes	no
Ladder Side-Tuning(Sung et al., 2022)	7.5	7.5	yes	yes	no
FAR (Vucetic et al., 2022)	6.6-26.4	6.6-26.4	yes	no	no
S4-model (Chen et al., 2023)	0.5	**more than 0.5**	yes	yes	no

https://arxiv.org/abs/2303.15647

Что важно для PEFT?

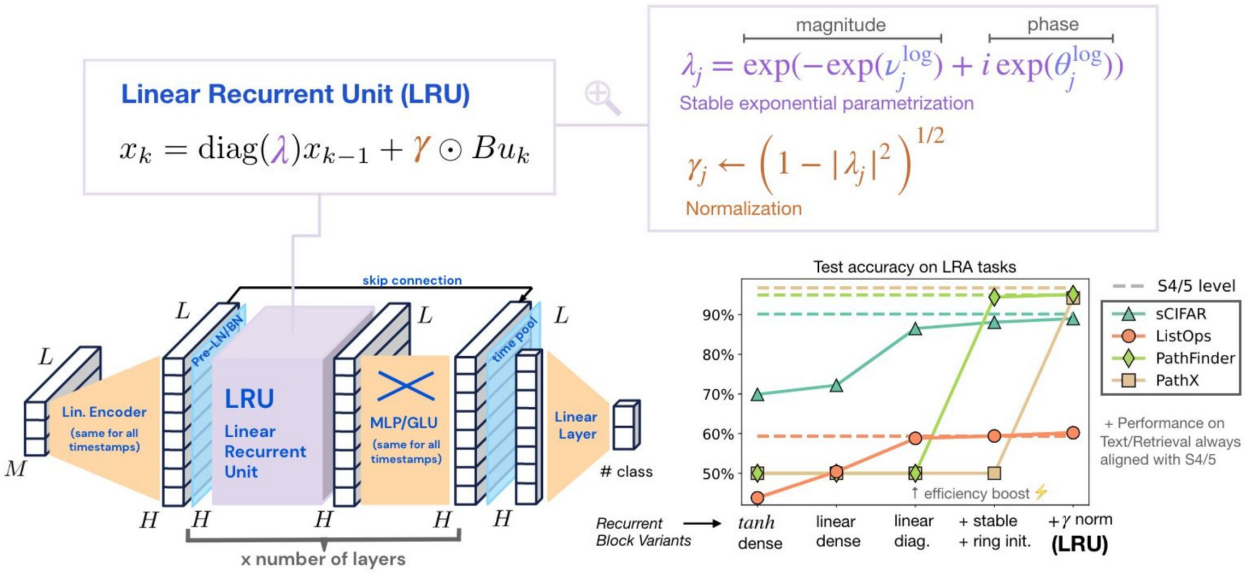
- Количество обучаемых параметров
- Эффективность тренировки (дополнительное количество параметров, необходимость backprop по оригинальной модели, утилизация GPU)
- Эффективность на инференсе
- Точность на финальных метриках

Что использовать?

- N*10-N*100 примеров –> Prompt engineering, IA3
- N*100/N*1000 примеров –> Prompt-tuning, IA3
- Больше примеров –> LoRA адаптеры

Альтернативные архитектуры

- RetNet
- RWKW
- LRU
- S4
- и др



Доклад Ильи Гусева про альтернативные архитектуры

Практические советы по тренировке Transformer`ов

- Используйте Learning Rate warmup / Inverse sort decay.
- Увеличивайте Batch size (можно аккумулировать по нескольким степам).
- Объединяйте в батч примеры одинаковой/похожей длины.
- Делайте скейлинг логитов attention`а на 1/sqrt(dim).
- Используйте Adam оптимизатор.
- Используйте warm-up batch-size, warm-up sequence length.
- И другие приемы из этих материалов.

https://www.borealisai.com/research-blogs/tutorial-17-transformers-iii-training/

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

Итоги занятия

- Познакомились с понятием LLM
- Изучили методы борьбы с галлюцинациями LLM
- Изучили PEFT-методы и способы их сравнения
- Бегло посмотрели на альтернативные трансформерам архитектуры
- Рассмотрели полезные рецепты для применения изученных методов

Спасибо
за внимание

МФТИ × SKILLFACTORY