



LLM fine-tuning для задачи персонализации рекламных сообщений

Октябрь 2024

Этим занимаются конкуренты

[PERSADO]

→ ↺ 📄 persado.com... ⌵ ☆ 📁 | ⬇️ 🌐

Evolution of Motivation AI

To create the Motivation AI platform, Persado has leveraged advanced machine learning techniques, such as **Supervised Fine-Tuning (SFT)** and Reinforcement Learning from Human Feedback (RLHF), to enhance the relevance and impact of its campaign messages across diverse industries, channels, and languages. By training

Generate Push Notification

Campaign & Offer Info

Apply for travel rewards card ✕

Product Info

travel rewards card ✕ highly rated ✕

Other Content Considerations

summer ✕ travel ✕ vacation ✕

Target Segment

Deal Seekers ▾

Generate

»

1

Could this be you? This summer, go farther with our highly-rated Travel Rewards Card. ✈️

91

2

Could this be you? This summer, go farther with our highly-rated Travel Rewards Card. ✈️ Enjoy exclusive travel perks and rewards with every trip!

87

3

This summer, take your vacation further! With our Travel Rewards Card, every destination gets closer. Highly-rated for a reason! ✈️

81

4

Summer adventures await! Use our highly-rated Travel Rewards Card to maximize your benefits and create unforgettable memories! ✈️

74

Deploy

Performance Prediction Breakdown

91

This message is expected to do better than 91% of messages based on data from 10,009,312 impressions.

87 Intro

63 Concept

76 Other

⚡ Insights

Contains CURIOSITY which is a top emotion for both your account and industry

🗨️ Tone

Fashion-forward

Conversational

Inclusive


Value Customer Loyalty

Модель для fine-tuning прямо не раскрывается, но упоминаются Llama и Mistral.



AdCreative.ai

📄 app.adcreative.ai/generated/ad-texts/375163/59731 📄 ⌵ ☆ 📁 | ⬇️ 🌐



AI-Generated Ad Texts

Your conversion-focused ad texts tailored to bring results.

How does it work? 🗨️

🕒 92 / 100

SUCCESS Method: Simple-Unexpected-Concrete-Credible-Emotional-Story Style Ad Texts

Представьте себе: вы оплачиваете поездку на такси, и за это получаете 1000 бонусных баллов 🚗💰. Это возможно с Ребс! 🌟

Для женщин поколения Z, начинающих свой бизнес, это шанс превратить каждую поездку в ваше преимущество. 🧠💼

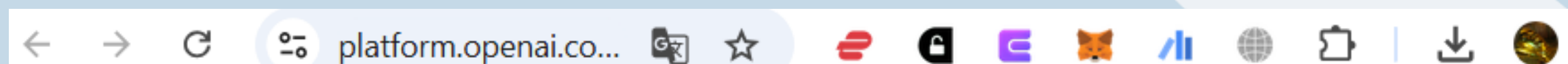
Что делает Ребс таким особенным? Это не просто карта – это ваш верный спутник в мире бизнеса. Надежный, удобный и выгодный. 🧭🌟

Представьте, как ваша подруга Лена, только что начавшая свое дело, зарабатывает бонусы на каждом шагу. Это не просто история – это ваша реальность с Ребс.

Присоединяйтесь к нам и начните зарабатывать бонусы уже на следующей поездке! 🚀

AdLLM Spark is a large language model, just like ChatGPT by OpenAI or Gemini by Google, but it is trained with over 840 million high-converting ad texts. This vast amount of data enables AdLLM to generate conversion-focused ad texts that bring results. This makes AdLLM the world's first large language model for advertising. AdLLM uses **Llama-3-70B as a base model** and was trained for over 1,000 hours using NVIDIA H100 Tensor Core GPUs on our dataset of over 1.5 billion tokens of high-conversion-rate ad texts from all major platforms such as Facebook, Instagram, Google, YouTube, LinkedIn, Microsoft, Pinterest, and TikTok.

Это перспективно для нашей задачи



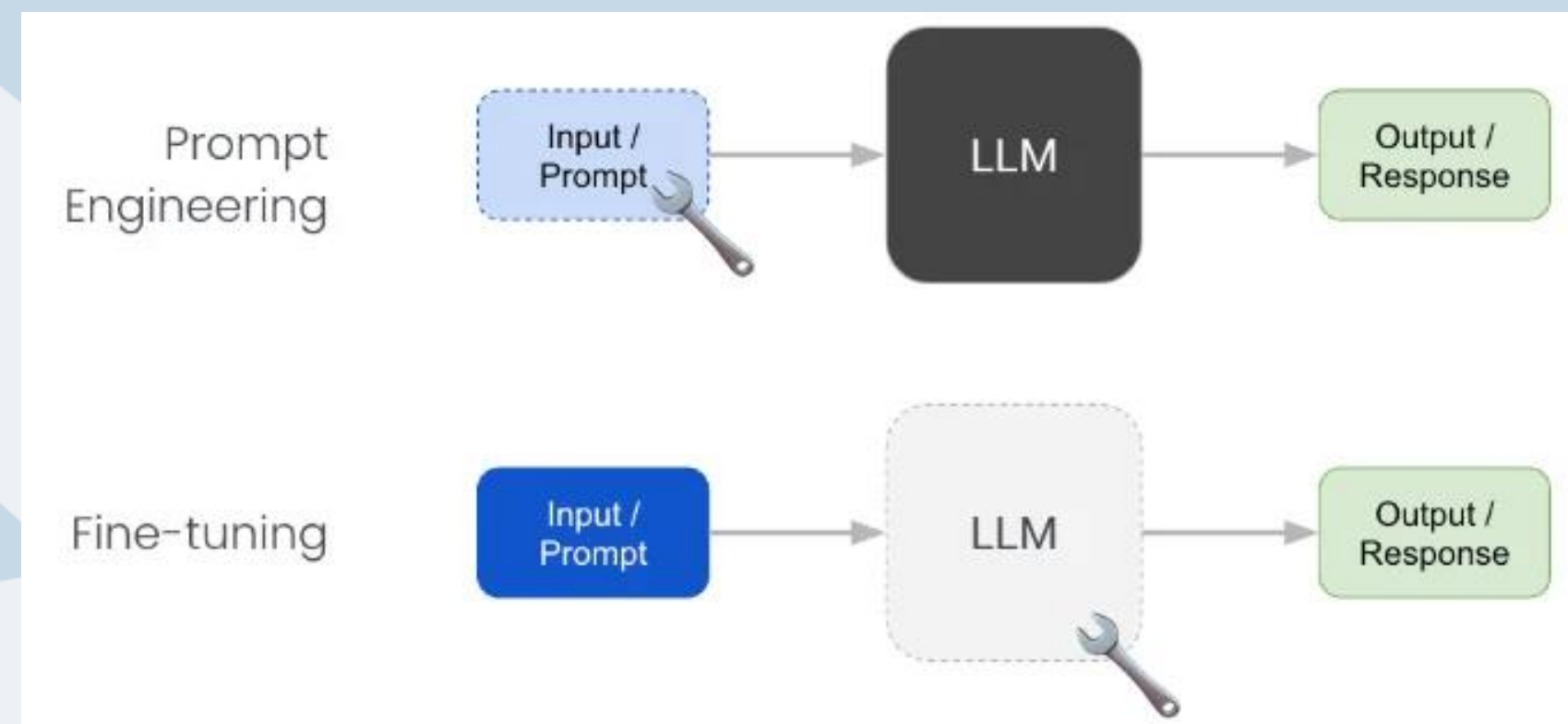
OpenAI Platform

Распространенные варианты использования

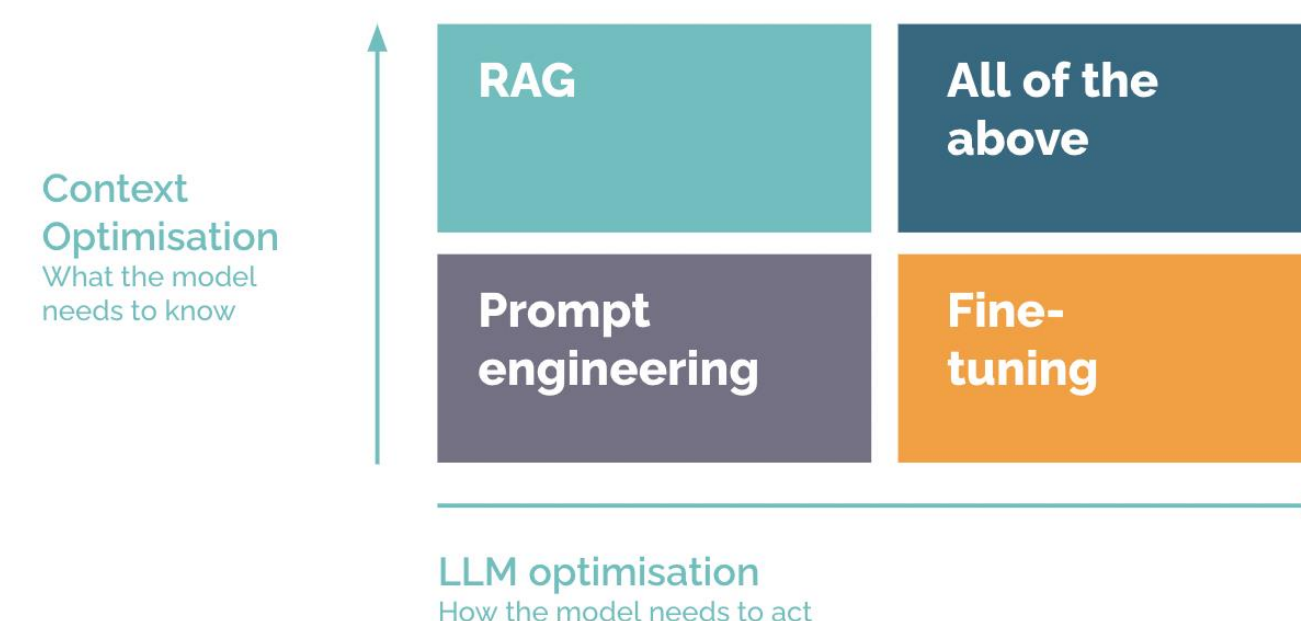
Вот несколько распространенных случаев, когда тонкая настройка может улучшить результаты:

- Установка стиля, тона, формата или других качественных аспектов
- Повышение надежности при получении желаемого результата
- Исправление ошибок при выполнении сложных подсказок
- Обработка множества пограничных случаев определенными способами
- Выполнение нового навыка или задачи, которую трудно сформулировать в подсказке

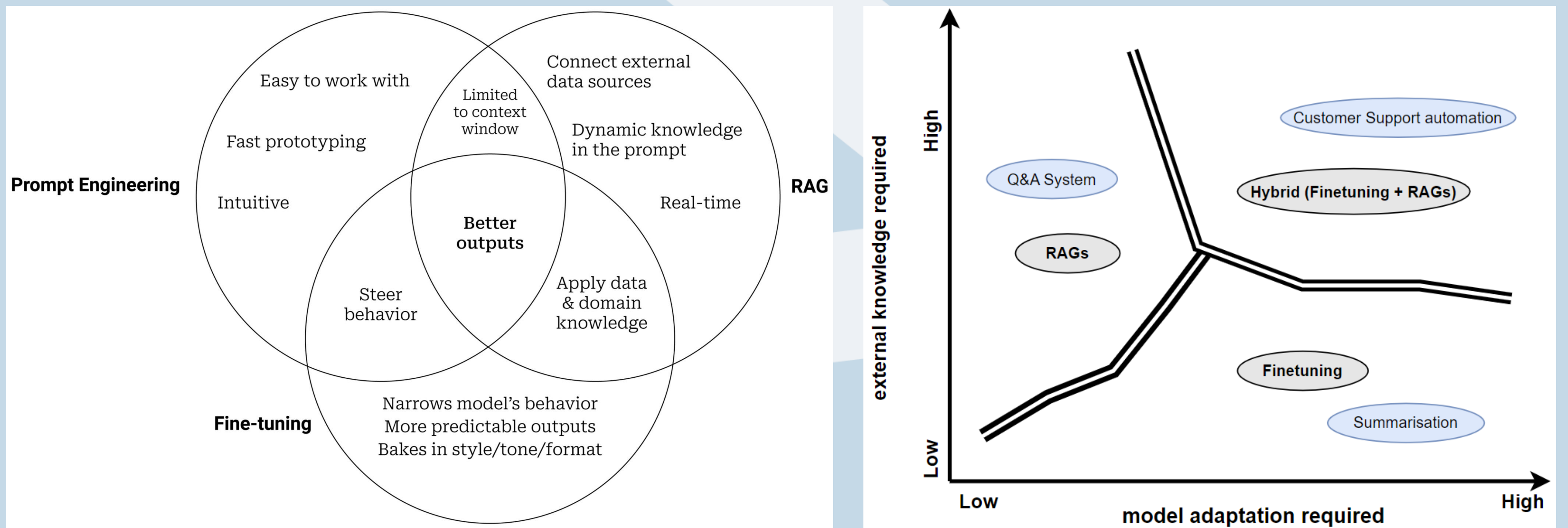
Один из высокоуровневых способов думать об этих случаях — когда проще «показать, а не рассказать». В следующих разделах мы рассмотрим, как настроить данные для тонкой настройки, и различные примеры, когда тонкая настройка улучшает производительность по сравнению с базовой моделью.



The optimisation flow

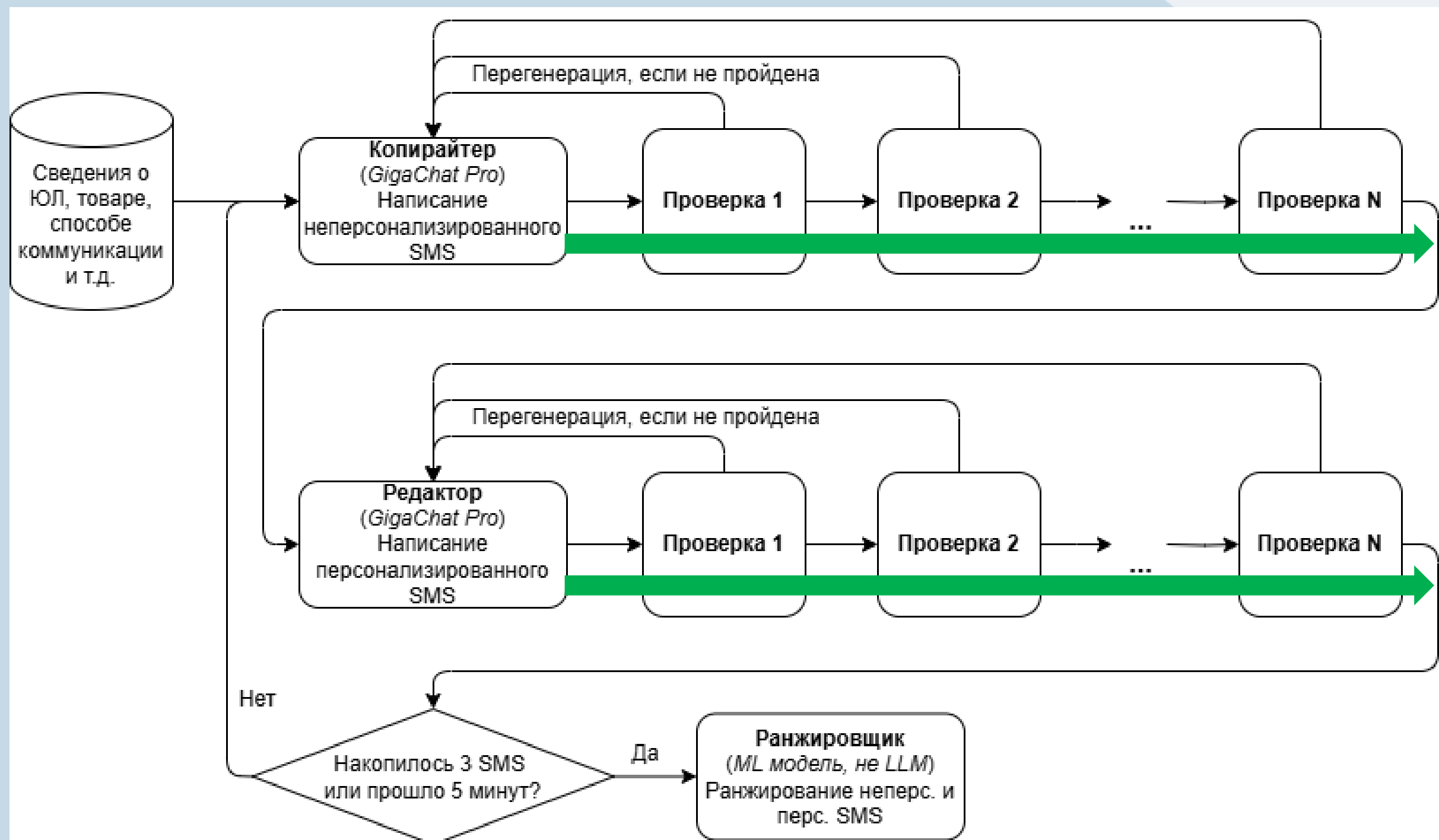


Ещё немного про prompt-engineering vs fine-tuning vs RAG



Совместимость с текущим прототипом

Fine-tuning не отменяет текущий прототип, а потенциально позволит достичь сообщений без ошибок за меньшее количество регенераций.



Как минимум, за счет fine-tuning можно попробовать научить модель генерировать сообщения без формализованных ошибок, поскольку мы можем создать большой синтетический датасет с «правильными» сообщениями.

Как максимум, можно попробовать научить модель создавать не только безошибочные, но и стилистически грамотные сообщения.

Актуальные источники

Исчерпывающий гайд по методам тонкой настройки больших языковых моделей.

Подробное руководство (<https://arxiv.org/pdf/2408.13296v1>) от Ирландского центра искусственного интеллекта CeADAR по практическому применению и оптимизации процесса тонкой настройки LLM.

В руководстве представлен анализ подходов обучения: контролируемые, неконтролируемые и инструктивные подходы. Гайд подробно рассматривает подготовку наборов данных, выбор подходящей модели, настройку параметров и оценку производительности.

Это руководство подходит как для начинающих, так и для опытных специалистов, которые хотят эффективно настраивать и использовать LLM для решения различных задач в области обработки естественного языка.

Несмотря на техническую сложность темы, авторы сделали материал доступным для широкой аудитории, используя понятный язык и наглядные примеры.

Содержание:

- Введение
- Семиэтапный конвейер тонкой настройки LLM
- Этап 1: Подготовка данных
- Этап 2: Инициализация модели
- Этап 3: Настройка обучения
- Этап 4: Выбор методов тонкой настройки и соответствующих конфигураций модели
- Этап 5: Оценка и валидация
- Этап 6: Развертывание
- Этап 6: Мониторинг и обслуживание
- Платформы и фреймворки для тонкой настройки LLM
- Мультимодальные LLM и их тонкая настройка
- Частые проблемы, этика и ответственность

arXiv > cs > arXiv:2408.13296

Computer Science > Machine Learning

[Submitted on 23 Aug 2024 (v1), last revised 21 Oct 2024 (this version, v2)]

The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs: An Exhaustive Review of Technologies, Research, Best Practices, Applied Research Challenges and Opportunities

Venkatesh Balavadhani Parthasarathy, Ahtsham Zafar, Aafaq Khan, Arsalan Shahid

This report examines the fine-tuning of Large Language Models (LLMs), integrating theoretical insights with practical applications. It outlines the historical evolution of LLMs from traditional Natural Language Processing (NLP) models to their pivotal role in AI. A comparison of fine-tuning methodologies, including supervised, unsupervised, and instruction-based approaches, highlights their applicability to different tasks. The report introduces a structured seven-stage pipeline for fine-tuning LLMs, spanning data preparation, model initialization, hyperparameter tuning, and model deployment. Emphasis is placed on managing imbalanced datasets and optimization techniques. Parameter-efficient methods like Low-Rank Adaptation (LoRA) and Half Fine-Tuning are explored for balancing computational efficiency with performance. Advanced techniques such as memory fine-tuning, Mixture of Experts (MoE), and Mixture of Agents (MoA) are discussed for leveraging specialized networks and multi-agent collaboration. The report also examines novel approaches like Proximal Policy Optimization (PPO) and Direct Preference Optimization (DPO), which align LLMs with human preferences, alongside pruning and routing optimizations to improve efficiency. Further sections cover validation frameworks, post-deployment monitoring, and inference optimization, with attention to deploying LLMs on distributed and cloud-based platforms. Emerging areas such as multimodal LLMs, fine-tuning for audio and speech, and challenges related to scalability, privacy, and accountability are also addressed. This report offers actionable insights for researchers and practitioners navigating LLM fine-tuning in an evolving landscape.

docs.together.ai/docs/fine-tuning-overview

together.ai

Documentation <> API Reference

GETTING STARTED

Introduction

Quickstart

OpenAI compatibility

MODELS

Serverless models

Dedicated models

CAPABILITIES

> Chat

Images

Vision

Code/Language

> Rerank

> Embeddings

< Finetuning

Finetuning

Learn how to use your own private data to fine-tune a custom LLM.

Fine-tuning LLM is the process of improving an existing LLM for a specific task or domain. You can improve an LLM by giving it a set of labeled examples for that task which it can then learn from. The examples can come from public datasets on the internet, or private datasets unique to your organization.

Together facilitates every step of the fine-tuning process. You can use our APIs for the following:

1. Uploading your own datasets to our platform
2. Starting a fine-tuning job that fine-tunes an existing LLM of your choice with your uploaded data
3. Monitoring the progress of your fine-tuning job
4. Hosting the resulting model on Together or download it so you can run it yourself locally

Together supports both [LoRA fine-tuning](#) and [Full fine-tuning](#) for next-token prediction. Get started fine-tuning a LLM with the following steps!

Choosing your model

platform.openai.co...

OpenAI Platform

Preparing your dataset

Once you have determined that fine-tuning is the right solution (i.e. you've optimized your prompt as far as it can take you and identified problems that the model still has), you'll need to prepare data for training the model. You should create a diverse set of demonstration conversations that are similar to the conversations you will ask the model to respond to at inference time in production.

Методы fine-tuning

Full fine-tuning vs

Сложнее, дороже,
есть проблема
«катастрофического
забывания»

Также может быть деление –
Supervised FT vs Unsupervised FT

PEFT Methods for LLMs

Additive Fine-tuning

Adapter Based Fine-tuning

- Adapter design
- Multi-task Adaptation
- Serial Adapter, Parallel Adapter, CIAT, CoDA
- AdapterFusion, AdaMix, PHA, AdapterSoup, MeRA, Hyperformer

Soft prompt-based Fine-tuning

- Soft Prompt design
- Training Speedup
- Prefix-tuning, Prefix-Propagation, p-tuning v2, APT, p-tuning, prompt-tuning, Xprompt, IDPG, LPT, SPPT, Aprompt
- SPoT, PTP, InfoPrompt, PTP, IPT, SMoP, DePT

Others

(IA)^3, MoV, SSF, IPA

Selective Fine-tuning

Unstructural Masking

U-Diff pruning, U-Bitfit, PaFi, FishMask, Fish-Dip, LT-SFT, SAM, Child-tuning

Structural Masking

S-Diff pruning, S-BitFit, FAR, Bitfit, XAttn Tuning, SPT

Layer-wise Adaptation

Token-specific Fine-tuning

Reparameterised Fine-tuning

Low-rank Decomposition

Intrinsic SAID, LoRA, Compacter, KronA, KAdaptation, HIW, VeRA, DoRA

LoRA Derivatives

- Dynamic Rank — DylORA, AdaLoRA, SoRA, CapaBoost, AutoLoRA
- LoRA Improvement — Laplace-LoRA, LoRA Dropout, PeriodicalLoRA, LoRA+
- Multiple LoRA — LoRAHub, MOELoRA, MoLoRA, MoA, MoLE, MixLoRA

Orthogonal Weight Regularization

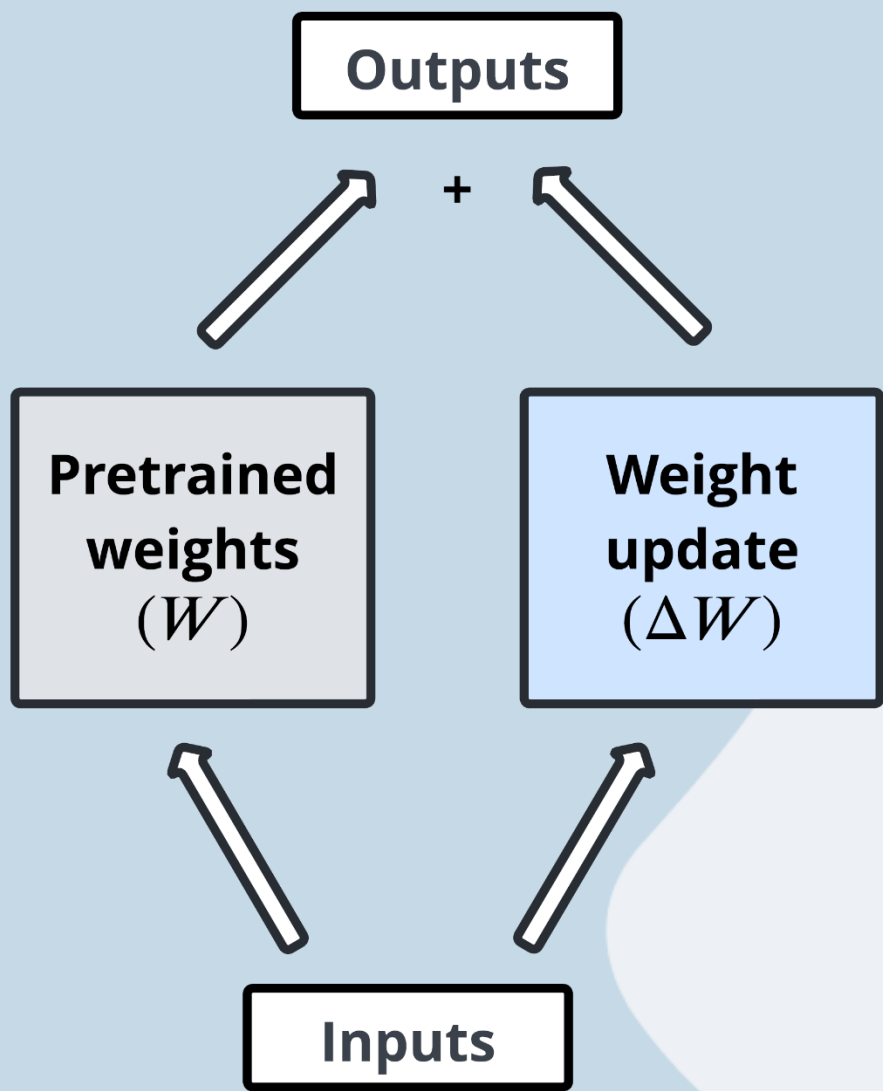
Adaptive Learning Rate Scheduling

Hybrid Fine-tuning

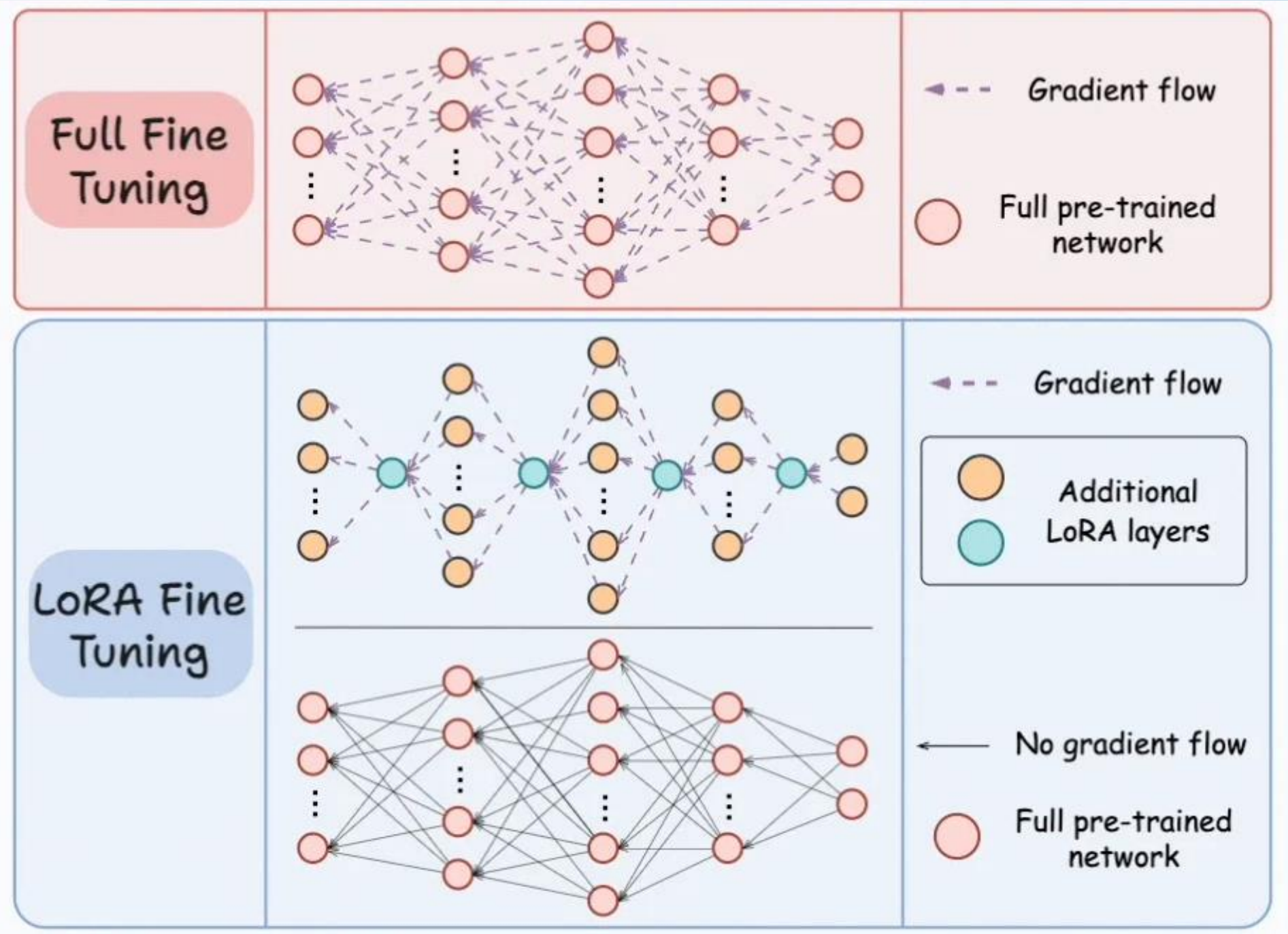
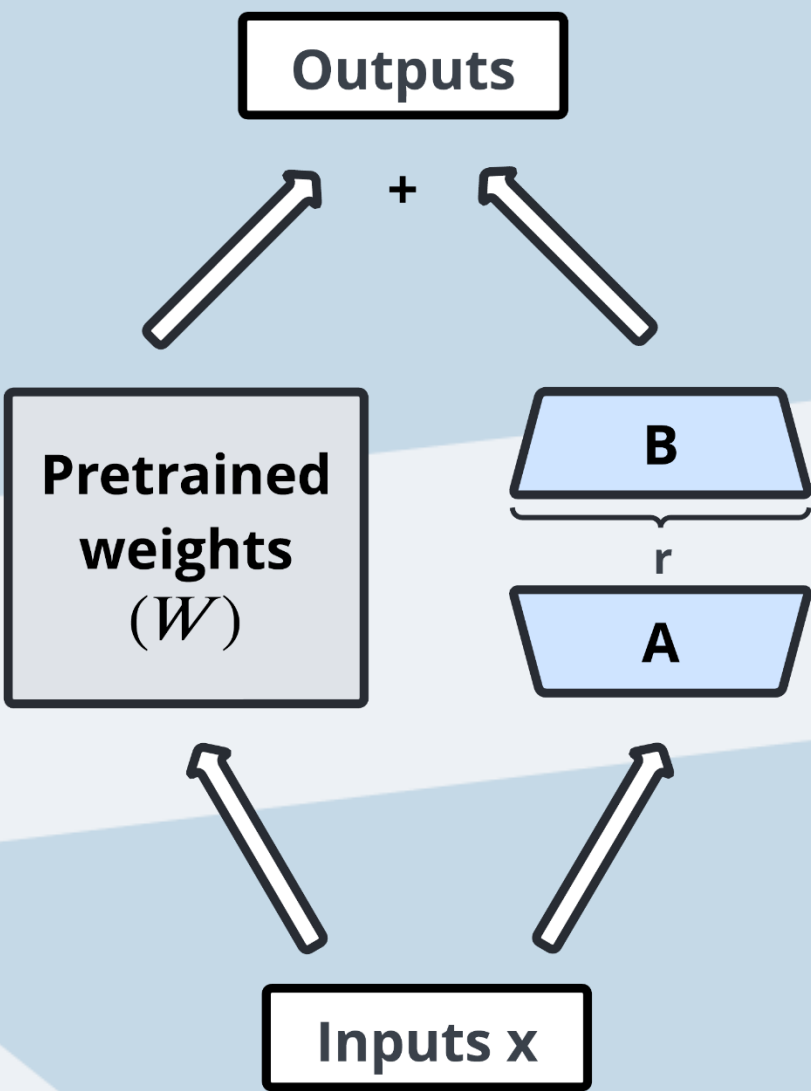
- UniPELT, S4, MAM Adapter, NOAH, AUTOPEFT, LLM-Adapter, SiPET
- Multi-adapter PEFT, LLM-PEFT, BiLingPET, CrossPET

LoRa

Weight update in regular finetuning



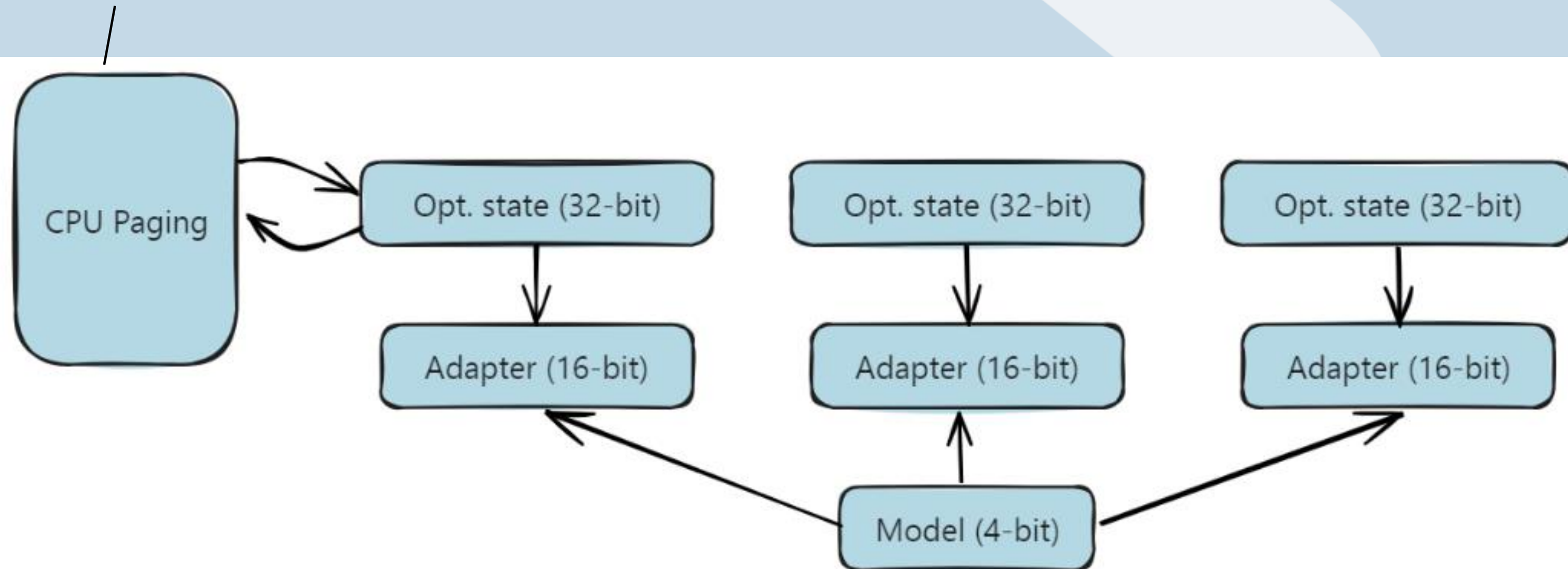
Weight update in LoRA



QLoRa

Оптимизация LoRa с точки зрения вычислительных ресурсов

подкачка памяти (часть дискового пространства как виртуальная оперативная память)

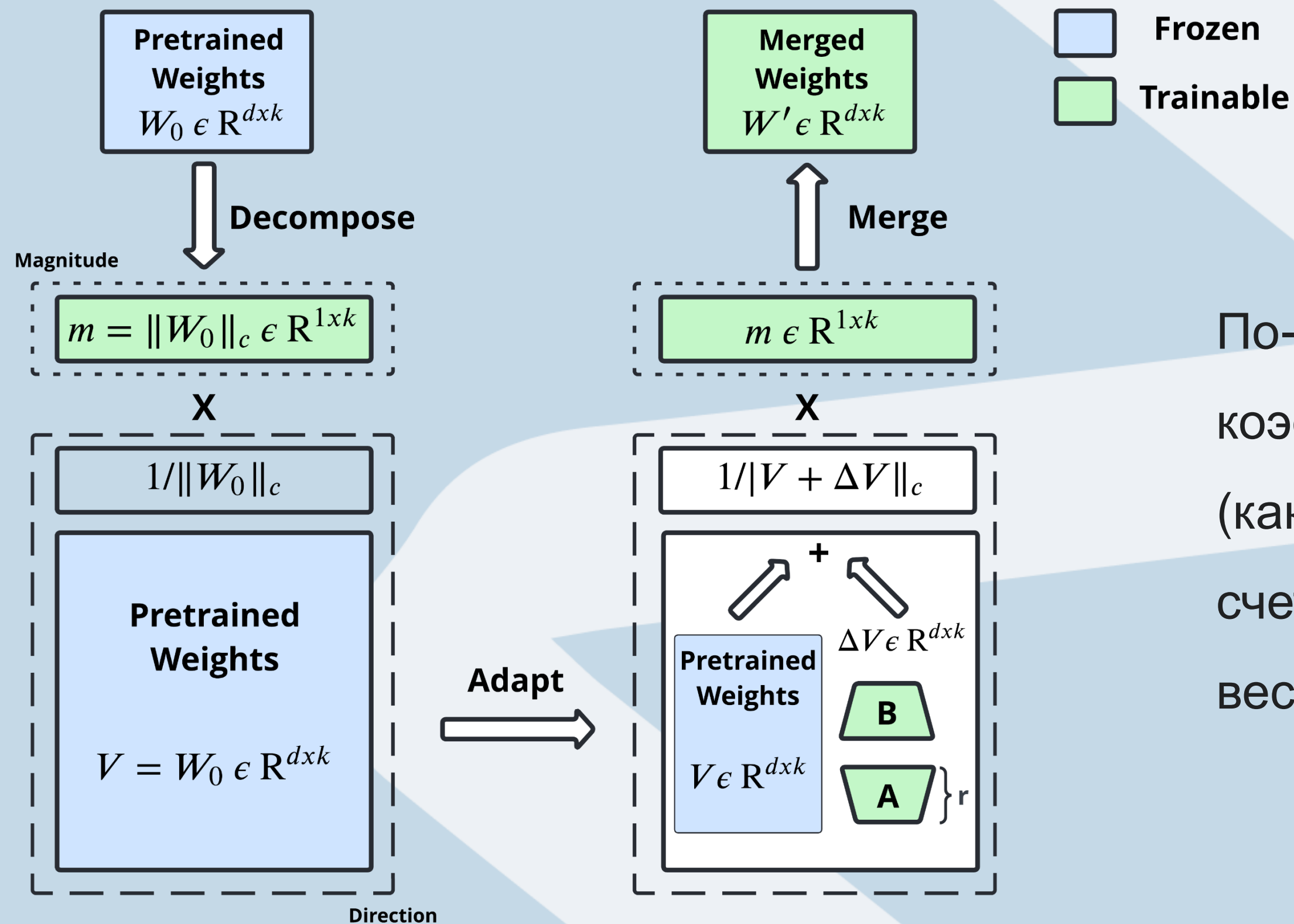


Состояния оптимизации
(градиенты, моменты и т.д.)

Коэффициенты матриц
LoRa

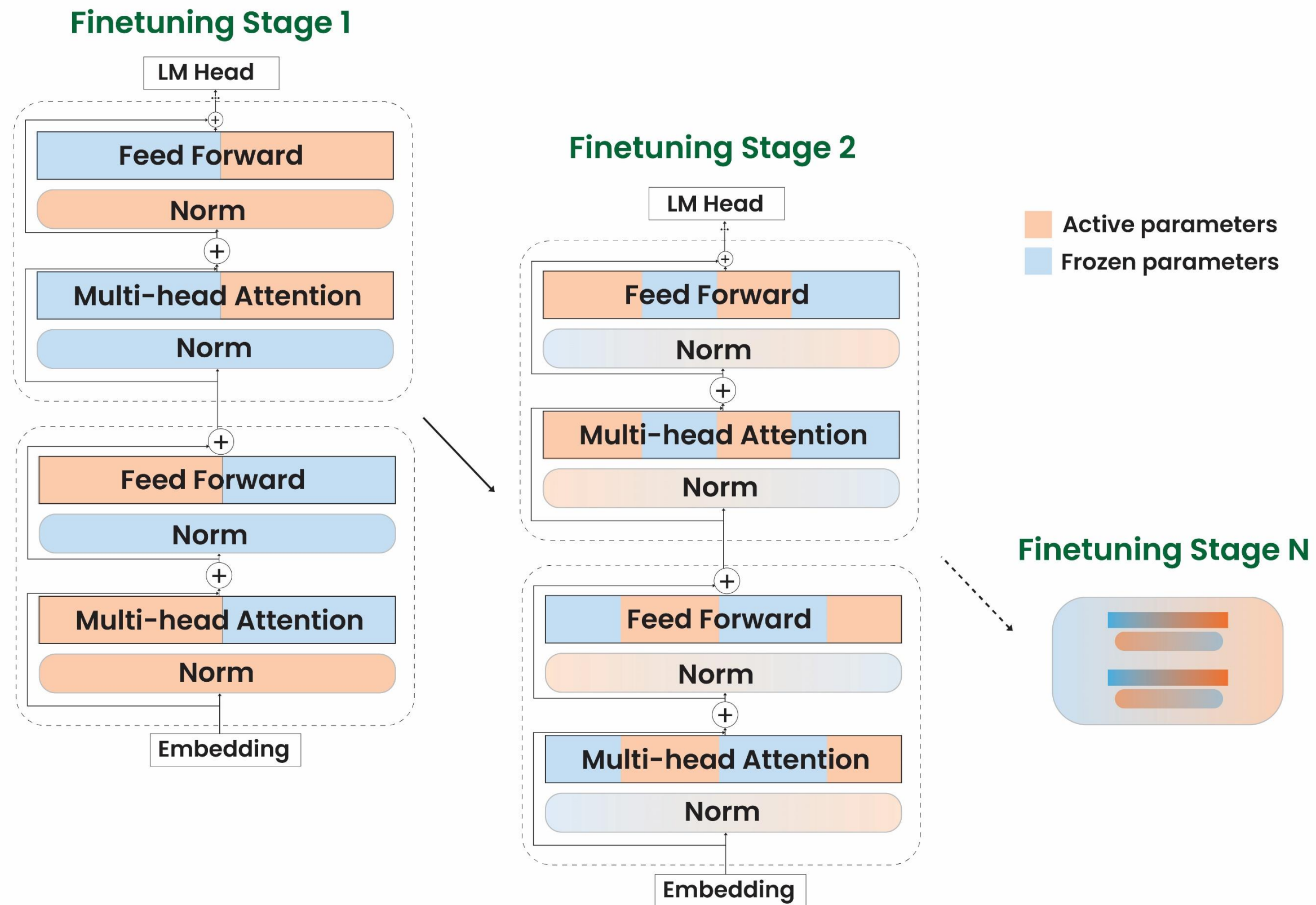
Основные веса модели

DoRA (Weight-Decomposed Low-Rank Adaptation)



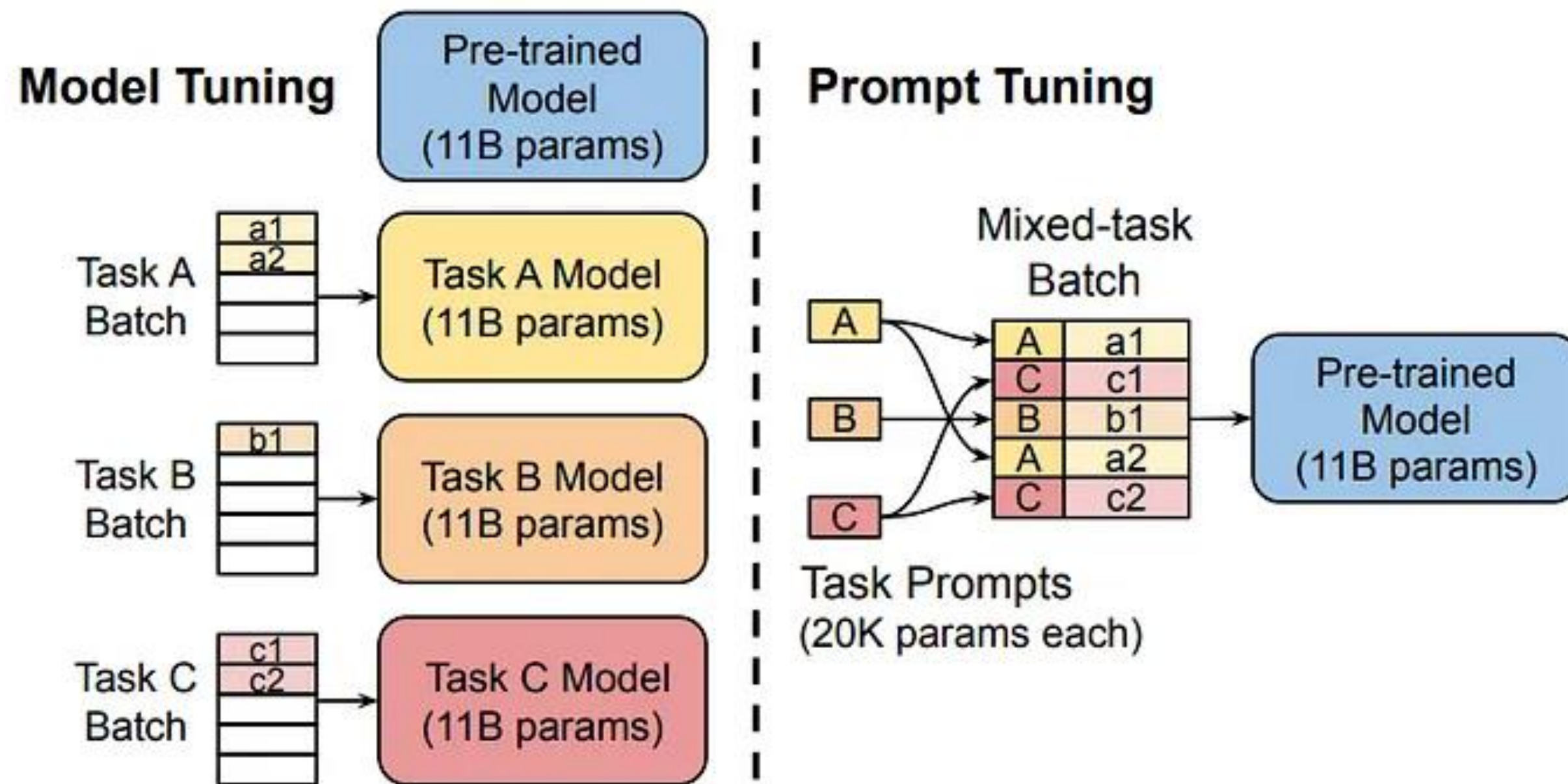
По-прежнему изменяем в ходе обучения коэффициенты низкоранговых матриц A и B (как в LoRa), но можем делать это точнее за счет дополнительного разложения исходных весов модели на величину и направление.

HFT (Half Fine-Tuning)



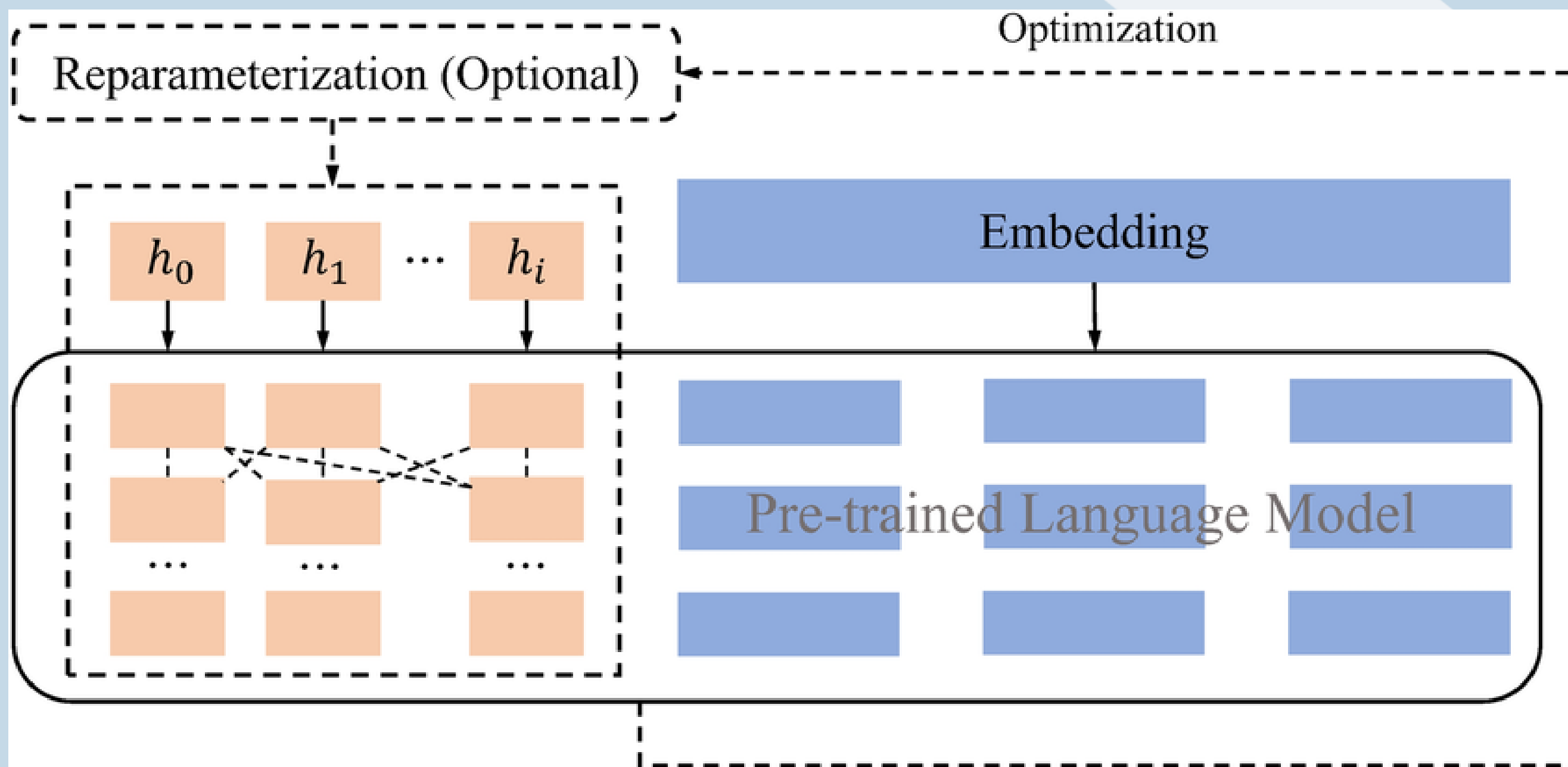
Не добавляем низкоранговые матрицы, а обновляем половину весов на каждой стадии.

Prompt-tuning (P-tuning)



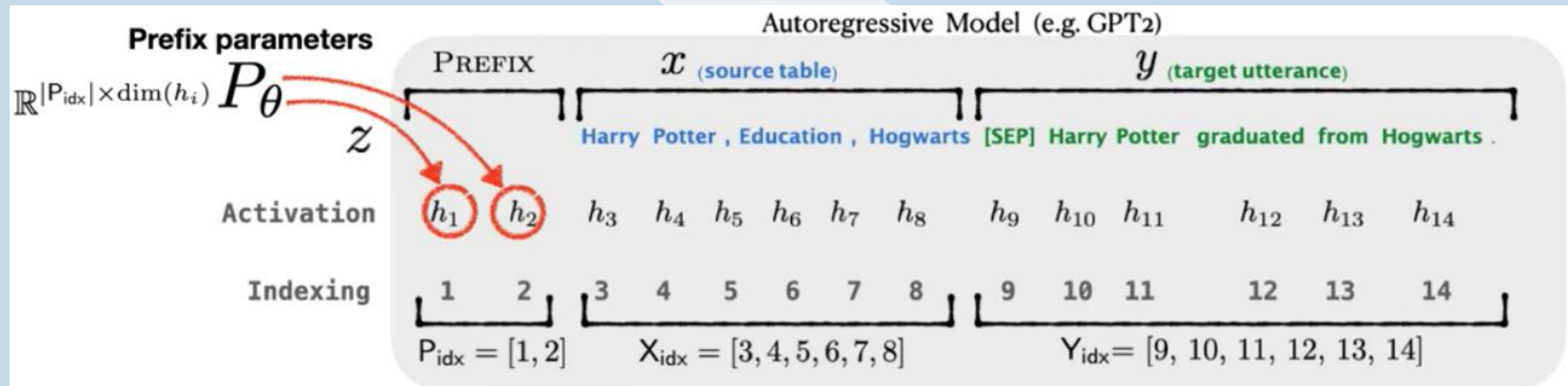
Крайне похоже на автоматический подбор промпта – но только в данном случае мы внедряем на уровне входных данных не текстовый промпт, а эмбединги, которые направляют модель. Эмбединги в данном случае влияют только на первые слои

P-tuning v2



Эмбеддинги также добавляются в начало входных данных, но влияют на все слои модели.

Prefix-tuning



Крайне похоже на p-tuning v2 – но только в данном случае префиксы внедряются не в виде эмбеддингов (т.е. не как входные данные, подаваемые вместе с запросом), а в виде добавок к функциям активации на каждом слое.

Существующие платформы, модели и методы для fine-tuning

→ ↻ docs.together.ai/docs/fine-tuning-models

together.ai

Documentation <> API Reference

GETTING STARTED

Introduction

Quickstart

OpenAI compatibility

MODELS

Serverless models

Dedicated models

CAPABILITIES

> Chat

Images

Vision

Code/Language

> Rerank

> Embeddings

> Finetuning

Data preparation

Instruct Formats

Fine-tuning Models

GUIDES

Fine-tuning Models

A list of all the models available for fine-tuning.

The following models are available to use with our fine-tuning API. Get started with [fine-tuning a model!](#)

- *Training Precision Type* indicates the precision type used during training for each model.
 - AMP (Automated Mixed Precision): AMP allows the training speed to be faster with less memory usage while preserving convergence behavior compared to using float32. Learn more about AMP in [this PyTorch blog](#).
 - bf16 (bfloat 16): This uses bf16 for all weights. Some large models on our platform uses full bf16 training for better memory usage and training speed.
- Long-context fine-tuning of Llama 3.1 (8B) Reference, Llama 3.1 (8B) Reference, Llama 3.1 (70B) Reference, Llama 3.1 Instruct (70B) Reference for context sizes of 32K-131K is only supported using the LoRA method.
- For Llama 3.1 (405B) Fine-tuning, please [contact us](#).

LoRA Fine-tuning

Organization	Model Name	Model String for API	Context Length	Max Batch Size	Min Batch Size	Training Precision Type*
Meta	Llama 3.1 (8B) Reference	meta-llama/Meta-Llama-3.1-8B-Reference	8192	32	8	AMP

openai.com/api/pricing/

Pricing

Fine-tuning models

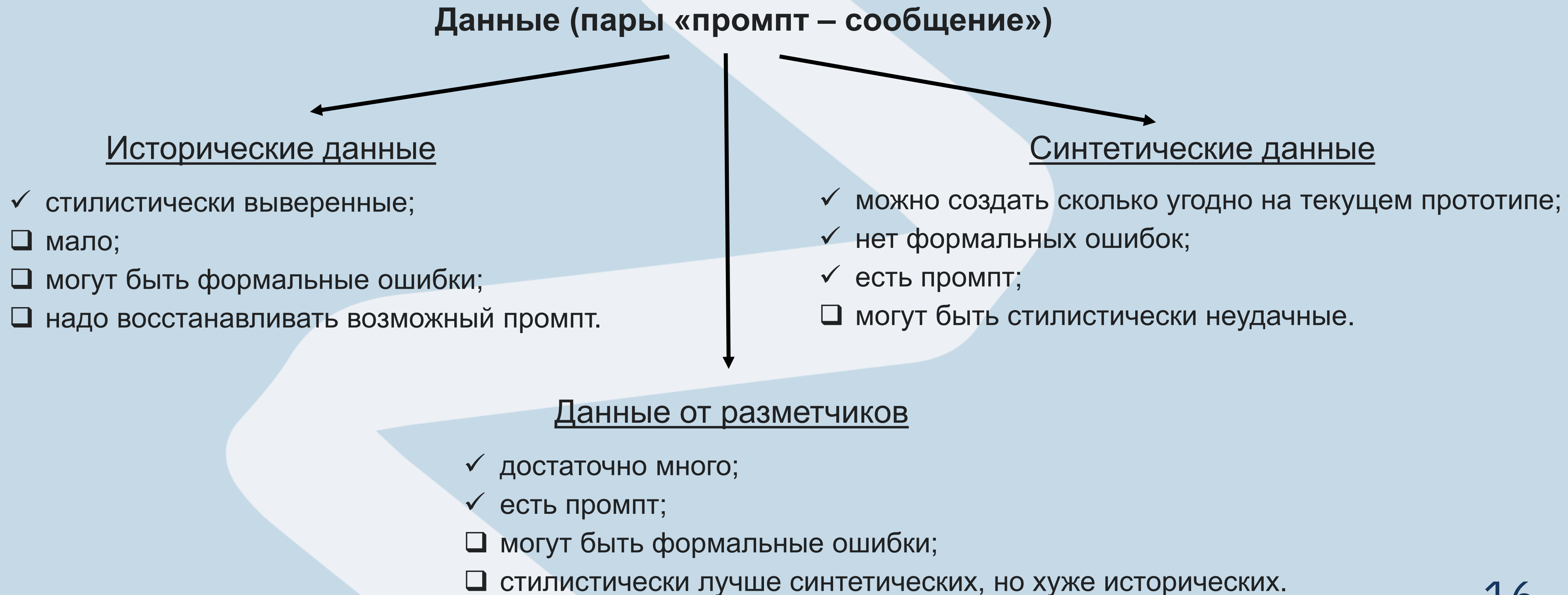
Create your own custom models by fine-tuning our base models with your training data. Once you fine-tune a model, you'll be billed only for the tokens you use in requests to that model.

Learn about fine-tuning ↗

Model	Pricing	Pricing with Batch API*
gpt-4o-2024-08-06**	\$3.750 / 1M input tokens	\$1.875 / 1M input tokens
	\$1.875 / 1M cached*** input tokens	
	\$15.000 / 1M output tokens	\$7.500 / 1M output tokens
	\$25.000 / 1M training tokens	
gpt-4o-mini-2024-07-18**	\$0.300 / 1M input tokens	\$0.150 / 1M input tokens
	\$0.150 / 1M cached*** input tokens	
	\$1.200 / 1M output tokens	\$0.600 / 1M output tokens
	\$3.000 / 1M training tokens	

LoRa?

На чем обучать



Что мы хотим увидеть при fine-tuning

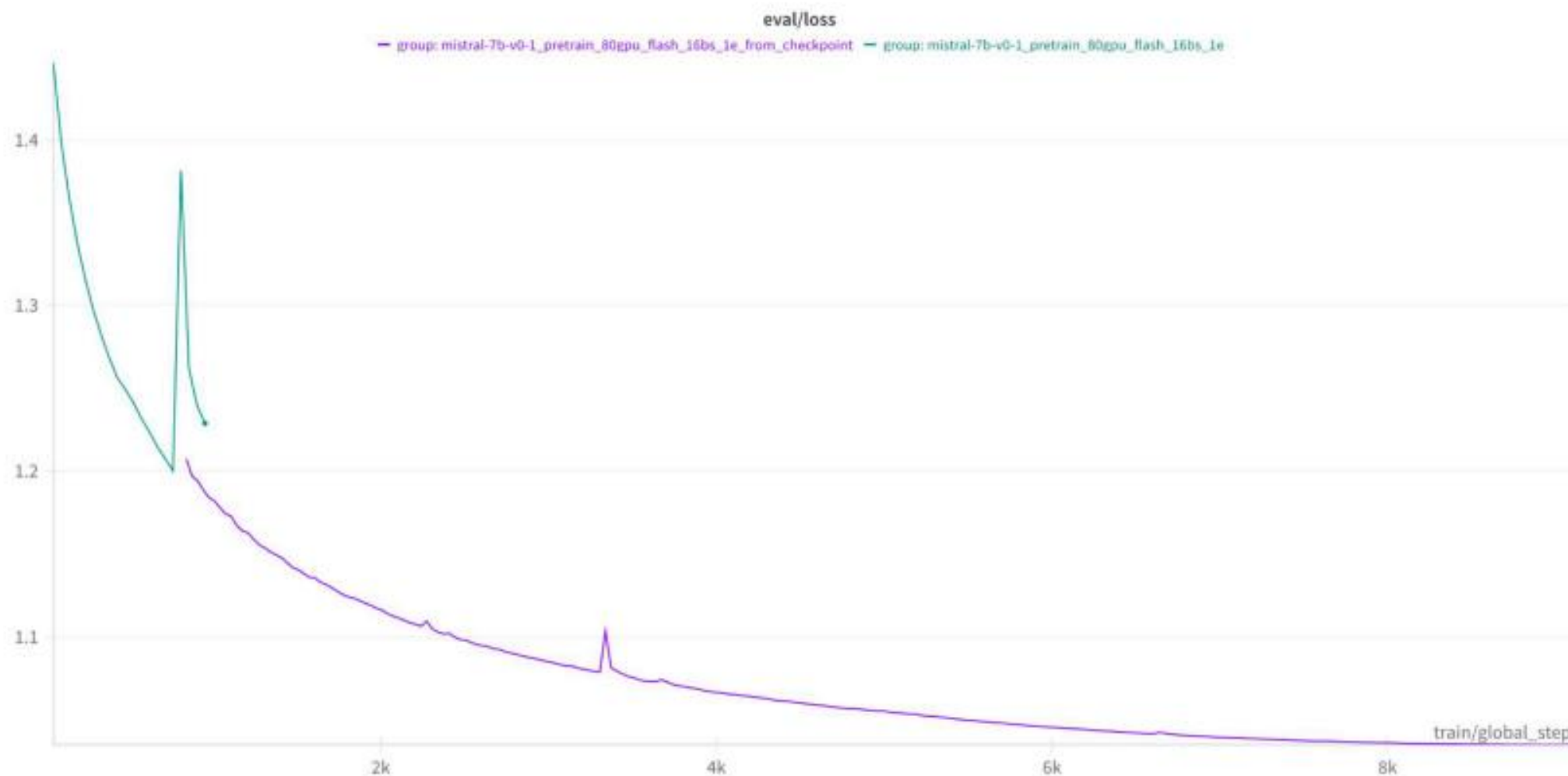
Anastasiya_Rysmyatova
24 окт в 09:00

Как мы обучили Mistral 7B русскому языку и адаптировали для объявлений Авито

Средний 7 мин 5.9K

Блог компании AvitoTech, IT-компании, Машинное обучение*, Искусственн

Мы знали, что может взрываться LOSS, и один раз он взорвался. Мы послушали много разных конференций и узнали, что делают коллеги в таких случаях — просто взяли один из последних чекпоинтов перед взрывом и пропустили часть данных после этого чекпоинта. Потом был ещё маленький взрыв, но мы решили пропустить его.



Для нашей задачи Loss может быть количеством регенераций / сработавших правил проверок.

При fine-tuning на together.ai можно бесплатно пользоваться BI от Weights & Biases/

Оценка стоимости fine-tuning

Example count recommendations

To fine-tune a model, you are required to provide at least 10 examples. We typically see clear improvements from fine-tuning on 50 to 100 training examples with `gpt-4o-mini` and `gpt-3.5-turbo`, but the right number varies greatly based on the exact use case.

We recommend starting with 50 well-crafted demonstrations and seeing if the model shows signs of improvement

Нижняя граница ~10 примеров
Верхняя граница ~10 000 примеров.

Model	Pricing	Pricing with Batch API*
gpt-4o-2024-08-06**	\$3.750 / 1M input tokens	\$1.875 / 1M input tokens
	\$1.875 / 1M cached*** input tokens	
	\$15.000 / 1M output tokens	\$7.500 / 1M output tokens
gpt-4o-mini-2024-07-18**	\$25.000 / 1M training tokens	
	\$0.300 / 1M input tokens	\$0.150 / 1M input tokens
	\$0.150 / 1M cached*** input tokens	
	\$1.200 / 1M output tokens	\$0.600 / 1M output tokens
	\$3.000 / 1M training tokens	

Пара «промпт – сообщение»: 300 слов на вход, 30 слов на выход.
1 слово ~1,5 токена.

500 токенов * 2500 примеров * (25\$*10⁻⁶) = 31\$ для gpt-4o
500 токенов * 2500 примеров * (3\$*10⁻⁶) = 4\$ для gpt-4o-mini
Но надо еще умножить на количество эпох (3-5).

MODEL:

Llama 3.1 Instruct (70B) Reference

TRAINING DATASET (TOKENS)

1,250,000

VALIDATION DATASET (TOKENS)

178,600

EPOCHS (# OF ITERATIONS)

4

NUMBER OF EVALUATIONS

4

ESIMATED COST

\$21.00

LoRA Fine-tuning						
Organization	Model Name	Model String for API	Context Length	Max Batch Size	Min Batch Size	Training Precision Type*
Meta	Llama 3.1 (8B) Reference	meta-llama/Meta-Llama-3.1-8B-Reference	8192	32	8	AMP
Meta	Llama 3.1 Instruct (8B) Reference	meta-llama/Meta-Llama-3.1-8B-Instruct-Reference	8192	32	8	AMP
Meta	Llama 3.1 (70B) Reference	meta-llama/Meta-Llama-3.1-70B-Reference	8192	16	8	AMP
Meta	Llama 3.1 Instruct (70B) Reference	meta-llama/Meta-Llama-3.1-70B-Instruct-Reference	8192	16	8	AMP
Meta	Llama 3 (8B)	meta-llama/Meta-Llama-3-8B	8192	32	8	AMP
Meta	Llama 3 Instruct (8B)	meta-llama/Meta-Llama-3-8B-Instruct	8192	32	8	AMP
Meta	Llama 3 (70B)	meta-llama/Meta-Llama-3-70B	8192	16	8	AMP
Meta	Llama 3 Instruct (70B)	meta-llama/Meta-Llama-3-70B-Instruct	8192	16	8	AMP
Meta	Llama-2 (7B)	togethercomputer/llama-2-7b	4096	128	8	AMP

Возможный план действий

- Доуточнить правила проверки (абстрактные заявления, клише и т.д.).
- Создать синтетические формально безошибочные примеры.
- Попробовать выполнить LoRa fine-tuning модели Llama-3.1-70B-Instruct-Reference на ~ 2500 синтетических примерах на платформе Together.ai.
- Оценить результаты с точки зрения разницы в количестве срабатываемых правил проверок.
- Обсудить с Олисеенко В.Д. и SberDevices возможность и методы fine-tuning для нашей задачи.
- Выполнить fine-tuning GigaChat на данных разметчиков и исторических данных.
- Оценить результаты.

Спасибо за внимание!