LLMOps

Управление большими языковыми моделями

Аби Ариан

УДК 004.056
ББК 32.973.26-018.2
А81

LLMOps: Managing Large Language Models in Production

Abi Aryan

Ариан, Аби.

А81 LLMOps: Управление большими языковыми моделями / Аби Ариан: [перевод с английского М. Стефанец]. — Алматы : Астана иностранная пресса, 2026. — 320 с. — (O’Reilly. Книги по программированию).

ISBN 978-601-12-9041-8

Книга «LLMOps: Управление большими языковыми моделями» посвящена практическим аспектам внедрения, эксплуатации и масштабирования систем на базе больших языковых моделей. Автор подробно рассматривает полный жизненный цикл LLM-проектов — от выбора архитектуры и подготовки данных до развертывания, мониторинга и оптимизации решений в промышленной среде.

Особое внимание уделено вопросам надежности, производительности, безопасности и стоимости эксплуатации моделей, а также адаптации LLM под конкретные бизнес-задачи. Рассматриваются современные подходы к организации LLMOps-процессов и интеграции моделей в существующую ИТ-инфраструктуру.

УДК 004.056
ББК 32.973.26-018.2

ISBN 978-601-12-9041-8

Посвящаю эту книгу своему брату Николасу,
которого я очень люблю и уважаю.

Оглавление

Предисловие

Я давно перестала считать, сколько раз меня спрашивали: «Чем LLM/AI-инженер отличается от LLMOps-инженера?» Этот вопрос всплывает снова и снова — коллеги задают мне его на рабочих встречах, конференциях и даже за кофе.

Я пробовала объяснять, какие именно задачи решает каждый из этих специалистов, но со временем пришла к выводу: люди не понимают, как сложно обеспечивать стабильную работу больших языковых моделей — LLM (Large Language Models) — без сбоев.

Информация о лучших моделях, технологиях и методах поддержки LLM обновляется каждые несколько дней. Единицы имеют представление о том, как работают эти системы. Многие до сих пор воспринимают *Ops* как простое развертывание, но в контексте LLM речь идет скорее о выстраивании взаимодействия между людьми, процессами и технологиями. Модели должны оставаться безопасными, надежными и устойчивыми бизнес-инструментами в любых условиях.

Компании и HR-отделы пытаются разобраться, что все это значит для команд и проектов. В своей книге я старалась дать подробный ответ на этот вопрос. Перед вами не просто руководство по описанию ролей или по разработке и развертыванию LLM — хотя эти аспекты затрагиваются. После релиза приложение на базе LLM нужно поддерживать и оптимизировать. В противном случае такое приложение усложнит решение простой задачи и, еще хуже, превратится в ненадежный инструмент, который не выдерживает серьезной нагрузки или становится легкой мишенью для атак с помощью промпт-инъекций.

В классическом сценарии (Software 2.0) все роли четко определены: инженеры разработки создают продукт, а инженеры сопровождения обеспечивают его стабильную работу. В случае с большими языковыми моделями функции также необходимо разграничивать. В Software 3.0 инженеры LLM/AI отвечают за создание, а инженеры LLMOps — за поддержку модели.

Хотя в основе LLMOps лежит набор методов и практик машинного обучения MLOps (Machine Learning Operations), навыки работы с табличными данными и дискриминативными моделями далеко не всегда применимы к генеративному искусственному интеллекту.

Эта книга поможет читателю понять уникальные особенности полного жизненного цикла LLM-приложений: от подготовки данных до развертывания

моделей, от проектирования API до мониторинга, обеспечения безопасности и оптимизации ресурсов. У вас в руках — инструмент для эффективной разработки, поддержки и оптимизации LLM-систем.

Условные обозначения, принятые в книге

В книге используются следующие способы выделения текста.

Курсив
: Курсивом выделены новые термины, URL, электронные адреса, имена и расширения файлов.

`Моноширинный шрифт`
: Моноширинным шрифтом выделены фрагменты кода, имена переменных или функций, базы и типы данных, переменные среды, инструкции и ключевые слова.

Благодарности

Выражаю благодарность Лукасу Мейеру за невероятную поддержку — его знания легли в основу нескольких глав этой книги. Хочу также поблагодарить редакторов Николь и Сару, без которых я не уложилась бы в срок. Спасибо рецензентам Лалиту Чури, Аммару Моханне и Нирмалу Будхатоки за ценные комментарии и рекомендации.

Особая благодарность моей семье — за моральную поддержку и нескончаемый запас чая. Отдельное спасибо всем читателям. Я надеюсь, что эта книга вдохновит на новые открытия как неофита, так и опытного специалиста.

ГЛАВА 1

Введение в большие языковые модели

Интерес к большим языковым моделям растет не просто так: они меняют наше взаимодействие с технологиями и расширяют возможности машинного обучения.

Однако есть важный нюанс: несмотря на впечатляющие результаты, масштабирование таких моделей и управление ими — задача непростая. Превратить прототип в надежный бизнес-инструмент сложно: этот путь сопряжен с множеством трудностей. Среди них и огромные вычислительные затраты, и сложная обработка данных, и необходимость обеспечить стабильную и безопасную работу как собственного решения, так и готового стороннего.

Прежде чем перейти к практическим вопросам эксплуатации больших языковых моделей, важно понять, как и зачем они появились. Знать основы — значит понимать, с какими проблемами мы сталкиваемся, когда пытаемся предсказать поведение LLM в реальных условиях.

Эволюция LLM — последовательность шагов, каждый из которых снимал ограничения предыдущих моделей. Первые системы имели урезанный набор функций и требовали активного участия человека даже для решения простых задач. Трансформеры вытеснили рекуррентные нейронные сети RNN (Recurrent Neural Networks), модели стали больше и мощнее, на фоне чего возникла необходимость обрабатывать огромные объемы данных и обеспечивать высокую эффективность обучения.

Итак, разберемся, как все устроено.

Ключевые термины

Прежде чем продолжить обсуждение, необходимо уточнить три ключевых понятия.

Базовые модели (Foundation Models)

Базовые модели — архитектуры машинного обучения, которые лежат в основе специализированных решений. Предобучаются на огромных датасетах, прежде всего текстах, но все чаще также на других типах данных: коде,

изображениях, звуке и видео. Цель такого обучения — сформировать общее понимание языка и способность распознавать закономерности. Эти модели кодируют статистические связи и языковые структуры из обучающего корпуса и создают прочную базу для тонкой настройки. Дообучение позволяет адаптировать системы под конкретные цели и задачи, например для работы LLM или других решений на базе искусственного интеллекта.

Большие языковые модели (Large Language Models)

Большие языковые модели — разновидность базовых моделей, дополнительно обученных и настроенных специально для работы с языком. Эти модели прогнозируют и генерируют текст, похожий на человеческий, за счет анализа и воспроизведения языковых закономерностей. LLM универсальны и применяются во множестве задач обработки естественного языка: генерации текста, анализе тональности, переводе, имитации диалога. Среди наиболее популярных сценариев — создание контента, мультиязычная коммуникация, анализ данных, написание кода. Самые востребованные агенты — чат-боты, рекомендательные системы и виртуальные ассистенты. Рассмотрим эти примеры более подробно в разделе «LLM в бизнесе: варианты использования больших языковых моделей».

Генеративные модели искусственного интеллекта (Generative AI Models)

Генеративный искусственный интеллект GenAI (*Generative AI*) — базовые модели, обученные генерировать контент (текст, изображения, звук и видео) на основе выявленных закономерностей и полученных знаний. Генеративные адверсариальные сети GAN (Generative Adversarial Networks) появились в 2018 году и стали одними из первых подобных моделей. За ними последовали диффузионные и большие языковые модели, а также мультимодальные решения типа Gemini. LLM генерируют контент, поэтому их справедливо относят к моделям генеративного искусственного интеллекта. LLM отвечают на вопросы, создают творческие тексты и описания товаров на основе входных данных и выученных паттернов.

Эти три понятия часто путают. Так, популярную нейросеть для генерации изображений DALL·E правильнее отнести к генеративным моделям искусственного интеллекта, а не к большим языковым моделям. Однако недавно ChatGPT, один из самых известных LLM-чат-ботов, научился создавать изображения с помощью DALL·E. Теперь пользователь может попросить LLM, например ChatGPT, сгенерировать картинку. Язык развивается в сторону упрощения, поэтому все описанные системы теперь называют общим термином «*AI-модели*».

Трансформеры

Трансформер — тип архитектуры, который впервые был описан в статье Attention Is All You Need («Все, что вам нужно, — внимание») [*https://oreil.ly/J8MBW*]. Именно трансформеры задали новые стандарты работы с языковыми данными и определили развитие современных языковых моделей.

До трансформеров в NLP широко использовались *рекуррентные нейронные сети*. Они обрабатывают данные последовательно, шаг за шагом. Такой подход можно применять в задачах, где имеет значение порядок входных данных, например при анализе текста. Однако у этого метода есть серьезный недостаток: сеть, которая обрабатывает длинные последовательности, забывает результаты с предыдущих временных шагов.

Чтобы понять, почему так происходит, рассмотрим процесс обучения нейросетей. Модель получает входные данные и выдает предсказание. Далее предсказание сравнивается с правильным ответом с помощью функции потерь, которая вычисляет ошибку — насколько предсказание отклоняется от эталона. Затем алгоритм, например *обратного распространения ошибки*, рассчитывает *градиенты* — значения, указывающие, как нужно скорректировать параметры модели (веса и смещения), чтобы минимизировать ошибку и повысить точность.

При работе с длинными последовательностями в RNN градиенты многократно перемножаются. Хранимое значение размывается во времени, машина начинает воспринимать градиенты как нули. Обучение фактически останавливается — модель перестает усваивать долговременные зависимости. Это явление называется *затуханием градиента*, и именно оно мешает RNN работать с длинными цепочками «воспоминаний».

Трансформеры решают эту задачу иначе. Они используют механизм внутреннего внимания (Self-attention) и параллельную обработку, что позволяет учитывать результаты со всех итераций. *Механизм внутреннего внимания* устроен так, что все токены в последовательности — слова в предложении — анализируют другие элементы входа независимо от их позиции. Для этого вычисляются веса внимания, которые показывают, насколько один токен важен по отношению к другому. Например, местоимение *это* замещает существительное, которое предшествует ему в предложении. Внутреннее внимание помогает модели правильно соотнести такие элементы, даже если они находятся далеко друг от друга. Таким образом, модель улавливает связи между всеми элементами входа, а не только соседними. Параллельная обработка дает трансформерам возможность значительно ускорять вычисления, а еще исключать затухание градиента и другие проблемы, связанные с пошаговыми методами.

Благодаря способности работать с долговременными зависимостями и обрабатывать огромные объемы данных, модели на основе трансформеров показывают высокую эффективность во множестве задач обработки естественного языка: от машинного перевода и саммаризации до ответов на вопросы. В сочетании с позиционным кодированием способность учитывать разные части последовательности вне зависимости от расстояния между элементами позволяет трансформерам обрабатывать большие фрагменты текста и не упускать важные зависимости.

Возникает справедливый вопрос: «Раз трансформеры масштабируются лучше прежних моделей, что будет, если дать им еще больше вычислительных ресурсов и данных?» Эксперименты с GPT-3, LLaMA[1] и более новыми системами показали: увеличение числа параметров принципиально улучшает результат трансформеров.

Сфера применения трансформеров не ограничивается обработкой естественного языка. Так, *визуальный трансформер ViT* (Vision Transformer) рассматривает патчи картинок как последовательность. Такой подход продемонстрировал отличные результаты в задачах классификации изображений и стал реальной альтернативой сверточным нейронным сетям CNN (Convolutional Neural Networks). В рекомендательных системах трансформеры также находят применение: их умение моделировать сложные зависимости повышает точность и уровень персонализации рекомендаций. В таблице 1.1 приведено сравнение различных архитектур нейросетей.

Таблица 1.1. *Эволюция архитектур нейронных сетей*

	Сверточные нейросети	**Рекуррентные нейросети**	**Трансформеры**
Применение	Работа с пространственными данными (например, изображениями)	Работа с последовательностями (например, текстом)	Универсальны: работа с изображениями, текстом, речью
Обработка данных	Высокая параллелизация	Последовательная обработка	Параллельная обработка входа
Работа с текстом	Для длинных цепочек требуется много слоев свертки	Справляются с длинными цепочками лучше, чем сверточные нейросети, но в пределах ограниченной длины	Справляются с длинными и сверхдлинными цепочками лучше, чем рекуррентные нейросети и сети долгой краткосрочной памяти
Масштабируемость	Высокая	Ограниченная	Очень высокая
Объем данных	Работают даже на малых выборках	Работают даже на малых выборках	Требуют больших датасетов

[1] Принадлежит компании Meta, которая признана экстремистской и запрещена на территории РФ. — *Здесь и далее — прим. автора, если не указано иное.*

	Сверточные нейросети	Рекуррентные нейросети	Трансформеры
Обучение	Простое	Более сложное, чем у сверточных нейросетей	Сложное
Интерпретируемость	Легко отлаживать	Сложно отлаживать	Сложно отлаживать
Развертывание	Простое	Простое	Сложное
Поддержка Edge AI	Есть	Есть	Поддержка ограничена
Объяснимость	Поддерживают разные методы	Ограниченная объяснимость	Крайне ограниченная объяснимость

Идея масштабировать трансформеры за счет увеличения данных и вычислительных мощностей привела к появлению больших языковых моделей. Параллельно трансформеры эволюционировали от архитектуры, ориентированной на одну модальность, к универсальной — способной работать сразу с несколькими типами данных. В различиях между архитектурами невозможно разобраться без понимания этого процесса.

Большие языковые модели

Большие языковые модели превосходно учитывают контекст и выявляют связи между словами, фразами и понятиями, что позволяет им давать осмысленные ответы на запросы. LLM автоматически извлекают знания из неструктурированных текстов — наполнять базы вручную не нужно. При обучении на разнообразных источниках такие модели охватывают колоссальные объемы информации без прямого участия человека. Однако здесь возникает проблема: вместе с полезными знаниями модели перенимают из данных предвзятые или ошибочные представления.

LLM анализирует текст и генерирует ответ как человек. Модель легко встраивается в интерактивные форматы от чат-ботов до голосовых помощников и понимает естественный язык. Этот инструмент упрощает поиск информации и превращает вопросы в структурированные решения.

Термин *большие* здесь связан не только с объемами данных, на которых обучаются модели, но и с числом параметров. *Параметры* можно представить как настройки модели, которые изменяются в процессе обучения, чтобы повысить точность ответов. К параметрам нейронных сетей относятся веса и смещения. Когда модель получает запрос в виде промпта, он преобразуется в числовое представление, которое проходит через слои сети. Каждый узел нейронной сети содержит смещение, которое прибавляется к входному значению или вычитается из него. Каждое соединение между узлами имеет вес, на который умножается