12+

АНДРЕЙ ЛАДЫГИН

Локальные LLM: от развертывания до создания автономных ИИ-агентов

ПРАКТИЧЕСКОЕ РУКОВОДСТВО
ПО ЗАПУСКУ, RAG, FINE-TUNING И АВТОМАТИЗАЦИИ
БЕЗ ОБЛАЧНЫХ API



Андрей Ладыгин

Локальные LLM: от развертывания до создания автономных ИИ-агентов. Практическое руководство по запуску, RAG, Fine-tuning и автоматизации без облачных API

«Издательские решения»

Ладыгин А.

Локальные LLM: от развертывания до создания автономных ИИ-агентов. Практическое руководство по запуску, RAG, Fine-tuning и автоматизации без облачных API / А. Ладыгин — «Издательские решения»,

Книга посвящена локальным большим языковым моделям (LLM) — технологиям, позволяющим запускать искусственный интеллект на собственном компьютере, или сервере без облачных API и платы за токены. Читатель пройдет путь от подбора оборудования и квантования моделей до построения систем RAG, тонкой настройки (LoRA/QLoRA) и создания автономных ИИ-агентов. Отдельное внимание уделяется информационной безопасности, соблюдению требований 152-ФЗ «О персональных данных» и лицензиям open-source моделей.

© Ладыгин А.
© Издательские решения

Содержание

Введение. Почему локальные модели — это новая реальность 6
Часть I. Фундамент: Железо, Теория и Выбор Модели 8
Глава 1. Железный вопрос: Что реально запустится на вашем ПК? 8
Глава 2. Магия квантования: Как впихнуть необъятное в вашу видеокарту 10
Глава 3. Как это работает под капотом: Токены, Эмбеддинги и Веса 12
Конец ознакомительного фрагмента. 14

Локальные LLM: от развертывания до создания автономных ИИ-агентов Практическое руководство по запуску, RAG, Fine-tuning и автоматизации без облачных API

Андрей Ладыгин

Дизайнер обложки ИИ GPT Image 2

© Андрей Ладыгин, 2026
© ИИ GPT Image 2, дизайн обложки, 2026

ISBN 978-5-0071-5358-4
Создано в интеллектуальной издательской системе Ridero

Введение. Почему локальные модели — это новая реальность

Еще вчера использование искусственного интеллекта ассоциировалось исключительно с облачными сервисами: вы отправляете запрос в ChatGPT, Claude или YandexGPT, а серверы корпораций где-то далеко в дата-центрах выполняют вычисления и возвращают вам ответ. Это было удобно. Но эта эпоха безоговорочного господства облаков заканчивается. Наступает эра **суверенного ИИ**.

Сегодня мы наблюдаем тектонический сдвиг. Российские гиганты (Яндекс, Сбер) устроили ценовую войну, снижая стоимость генерации токенов в разы. Казалось бы, зачем что-то усложнять, если можно просто заплатить? Ответ кроется в трех фундаментальных проблемах: **безопасность, экономика и контроль**. Именно они заставляют разработчиков, СТО и предпринимателей переводить свои ИИ-системы на локальные рельсы.

Экономика токенов: Сколько вы платите за воздух?

Давайте посмотрим на цифры. Согласно рыночным данным, генерация 1000 токенов в YandexGPT Pro 5.1 стоит около 0,80 руб, в GigaChat Pro — 0,50 руб, а у китайского DeepSeek (через API) — около 0,112 руб *(тарифы меняются, проверяйте актуальные цены у провайдеров)*. Если вы пишете пару писем в неделю, эти суммы кажутся смешными. Но когда речь заходит о корпоративном конвейере, обрабатывающем десятки тысяч документов, счет идет на миллионы рублей в год.

Локальная модель стоит 0 рублей за токен. Вы один раз инвестируете в видеокарту (или уже имеющийся ПК) и получаете безлимитный доступ к вычислениям. Окупаемость такой системы наступает уже через несколько месяцев активной работы.

Безопасность и суверенитет данных

Вторая, и часто главная причина — это данные. Отправляя запрос в облако, вы теряете над ним контроль. Для разработчиков ИИ-ассистентов в медицине, в юридической сфере или для государственных структур передача конфиденциальной информации на сторонние серверы — это прямое нарушение законодательства (включая 152-ФЗ «О персональных данных») и колоссальный риск утечки коммерческой тайны.

Локальное развертывание (On-Premise) означает, что **все вычисления происходят под вашим контролем**. Ваши данные никогда не покидают ваш сервер или ноутбук. Это не просто тренд, это элемент стратегии информационной безопасности.

Важно (дисклеймер): Для обработки персональных данных в соответствии с 152-ФЗ пригодны только варианты локального (изолированного) развертывания на территории РФ. Даже в этом случае мы рекомендуем маскировать ПДн перед подачей в модель, чтобы минимизировать риски.

Честный разговор: Кому открытая модель НЕ нужна?

Мы ценим ваше время и не хотим продавать вам инструмент, который вам не подходит. Если вы используете нейросеть пару раз в неделю, чтобы написать шаблонное письмо, а ваши данные не представляют коммерческой тайны — **локальная модель вам не нужна**. Подписка на облачный сервис решит ваши задачи быстрее и проще.

Но если вы относитесь к одной из шести категорий (разработчики, предприниматели, специалисты по ИБ, продакт-менеджеры, исследователи или госструктуры), то переход на собственное железо — вопрос времени.

Что вас ждет в этой книге?

Наша цель — провести вас по пути от «с чего начать» до создания полностью автономного ИИ-агента. Мы не будем изобретать велосипед или учить вас математике нейросетей. Вместо этого мы дадим вам инженерный подход.

— **В Части I** мы разберем ваше железо (AVX2, VRAM) и научимся выбирать модели. *(Сноска: Большинство моделей, таких как DeepSeek и Qwen, распространяются под лицензиями MIT и Apache 2.0, что разрешает коммерческое использование. Однако при скачивании с Hugging Face возможны ограничения доступа из РФ).*

— **В Части II** мы развернем Ollama и LM Studio, а также разберемся, почему локальные модели «глючат» и как это исправить.

— **В Части III** мы заставим модель читать ваши документы (RAG) и дообучим её под ваш стиль (Fine-tuning).

— **В Части IV** мы создадим автономных агентов, которые будут выполнять действия, а не просто генерировать текст.

— **В Части V** мы поговорим о профессиональной реализации в качестве промпт-инженеров.

Добро пожаловать в мир, где ИИ работает на вас, а не на облачного провайдера. Давайте начинать.

Часть I. Фундамент: Железо, Теория и Выбор Модели

Глава 1. Железный вопрос: Что реально запустится на вашем ПК?

Существует расхожее мнение, что для работы с искусственным интеллектом нужны серверные стойки, заполненные видеокартами NVIDIA стоимостью в автомобиль. Это правда лишь отчасти. Если вы хотите обучить модель уровня GPT-4 с нуля — да, вам понадобится суперкомпьютер. Но если ваша цель — запустить готовую open-source модель для анализа документов, написания кода или генерации текста, вам вполне хватит обычного домашнего или офисного компьютера.

Однако, прежде чем скачивать терабайты весов, нужно трезво оценить ресурсы вашего «железа». В этой главе мы разберем, из чего состоит модель, почему объем оперативной памяти важнее процессора, и как заставить тяжелую нейросеть летать на скромной видеокарте.

1. Анатомия модели: Что скрывается за цифрами 7B, 70B и 671B?

Когда вы видите название модели, например, DeepSeek-R1-Distill-Qwen-7B, цифра «7B» означает количество параметров — 7 миллиардов. Параметры (веса) — это, грубо говоря, числовые значения, которые определяют связи внутри нейросети и которые она подбирает во время обучения.

Чем больше параметров, тем «умнее» модель, но тем больше ресурсов ей требуется. Для понимания масштаба:

— Полная модель DeepSeek V3/R1 имеет **671 млрд** параметров (671B). Чтобы запустить её в оригинальном виде, требуются серверные кластеры с сотнями гигабайт видеопамяти. На домашнем ПК она не запустится.

— Урезанные (дистиллированные) версии (от **1.5B до 70B**) — это модели, которые «учились» у большой модели, но имеют меньший размер. Именно они являются целью нашего практикума, так как могут работать на потребительском железе.

Чтобы понять, как модель использует память, представьте расчет стоимости дома. У вас есть формула: *Расстояние от МКАД (вес 1) + Размер участка (вес 2) + Материал (вес 3)*. В нейросети происходит то же самое: параметры умножаются на веса, чтобы выдать результат. Чем больше параметров, тем больше вычислений нужно сделать в секунду.

2. Архитектура ПК: Требования к железу

Давайте разберем минимальные системные требования, которые предъявляются к современному ПК для работы с локальным ИИ. Что реально нужно вашему компьютеру?

1. **Процессор (CPU):** Обязательно должен поддерживать набор инструкций **AVX2** (это почти все современные процессоры Intel Core и AMD Ryzen). Без AVX2 современные инструменты вроде Ollama просто не запустятся или будут работать катастрофически медленно.

2. **Оперативная память (RAM):** Минимальный порог — **16 ГБ**. Если вы планируете работать с моделями побольше (14B-32B) или использовать RAG (поиск по документам), настоятельно рекомендуется **32 ГБ** и выше.

3. **Накопитель (SSD):** Модели весят много. Одна модель в 8B может занимать от 4 до 16 ГБ. Вам понадобится минимум **50 ГБ свободного места** на быстром SSD (желательно NVMe), так как при запуске модель загружается с диска в память.

3. Битва CPU и GPU: Почему видеокарта решает всё

Формально запустить LLM можно вообще без видеокарты, только на процессоре (CPU). Инструменты вроде llama. cpp (на котором базируется Ollama) отлично это умеют. Оба этих инструмента распространяются под свободной лицензией MIT, что разрешает их бесплатное коммерческое использование. Но есть нюанс: **скорость**.

Процессор обрабатывает запросы последовательно, поэтому генерация будет идти медленно (около 3—5 токенов в секунду). Видеокарта (GPU) обрабатывает матрицы параллельно, выдавая десятки токенов в секунду.

— **NVIDIA:** Идеальный выбор для локального ИИ благодаря технологии **CUDA** (проприетарная, но бесплатная для конечных пользователей). Если у вас карта от RTX 3060 (12 ГБ) и выше — вы в отличной форме.

— **AMD и Intel:** Поддерживаются, но требуют больше танцев с бубном при настройке ускорения (ROCm для AMD, DirectML для Windows).

— **Встроенная графика (Intel/AMD iGPU):** Обычно не подходит для ускорения. Если у вас ноутбук без дискретной видеокарты, система автоматически переключится на использование CPU и оперативной памяти.

4. Квантование: Магия, которая спасает ваш ПК

Что делать, если у вас видеокарта на 8 ГБ, а модель весит 16 ГБ? На помощь приходит **квантование**.

Квантование — это процесс сжатия весов модели с потерей незначительной точности. Если объяснять просто: мы переводим числа из формата с плавающей точкой (FP16) в целые числа (INT4 или INT8).

— Модель в 8B параметров весит около 16 ГБ (FP16).

— После квантования до 4-бит (Q4) она будет весить около 4—5 ГБ, но сохранит 90—95% своего «интеллекта».

Именно поэтому на домашнем ПК с 8—16 ГБ видеопамяти можно комфортно запускать модели 7B-14B в формате GGUF (4-bit). Мы подробно разберем виды квантования и выберем оптимальный вариант в следующей главе.

Итог: Так влезет или нет?

Запомните главное правило: **Модель должна помещаться в вашу видеопамять (VRAM)**. Если она не влезает, система начнет использовать оперативную память (RAM) и процессор, и скорость генерации упадет в 10—20 раз.

Если у вас 8 ГБ VRAM — ориентируйтесь на модели 7B-8B (Q4).

Если у вас 16 ГБ VRAM — вам доступны 14B (Q4) или 8B (Q8).

Если у вас 24 ГБ VRAM (как на RTX 3090/4090) — вы можете запускать 32B модели, и это будет очень мощный инструмент.

Не расстраивайтесь, если ваше железо кажется слабым. В следующей главе мы установим софт и проверим, на что реально способен ваш компьютер.

Юридическая сноска (для раздела про локальный запуск): Наличие локального оборудования (On-Premise) — это важнейший шаг к безопасности, но само по себе железо не гарантирует соблюдение 152-ФЗ. Для обработки персональных данных вам потребуется внедрить организационные меры (назначить ответственного за ИБ, настроить разграничение прав доступа, вести журналы аудита) и обеспечить физическую защиту сервера.

Глава 2. Магия квантования: Как впихнуть необъятное в вашу видеокарту

В предыдущей главе мы выяснили главное правило локального ИИ: модель должна помещаться в видеопамять (VRAM). Но что делать, если вы купили видеокарту на 8 или 12 ГБ, а вам хочется запустить модель, которая в оригинале весит 16 или 30 ГБ?

На выручку приходит **квантование** — самая важная технология для домашнего ИИ. Без неё локальные LLM остались бы уделом владельцев серверных станций. В этой главе мы разберем, как работает сжатие, что означает загадочная аббревиатура «GGUF» и как выбрать правильный файл модели, чтобы она не тормозила и не теряла логику.

1. Анатомия сжатия: Зачем терять точность?

Как мы уже говорили, нейросеть состоит из параметров (весов). В процессе обучения эти веса представляют собой числа с плавающей точкой высокого разрешения (обычно FP16 или FP32). Это позволяет модели быть очень точной.

Однако для генерации текста нам не нужна хирургическая точность до 10-го знака после запятой. Нам достаточно приблизительных значений.

Представьте, что вы смотрите на фотографию в формате RAW (весит 50 МБ) и в формате JPEG (весит 5 МБ). На экране смартфона вы вряд ли заметите разницу, но места JPEG займет в 10 раз меньше. **Квантование — это сжатие JPEG для нейросетей.**

Мы переводим веса из дробных чисел (16 бит) в целые числа (8 бит или даже 4 бита). Модель теряет микроскопическую долю своей «интеллектуальной остроты», но становится в 2—4 раза легче.

— Модель 8B в формате FP16 = ~16 ГБ.
— Модель 8B в формате Q8 (8-bit) = ~8.5 ГБ.
— Модель 8B в формате Q4 (4-bit) = ~4.5 ГБ.

2. Что такое GGUF и почему это стандарт для локального ИИ?

Гуляя по просторам Hugging Face, вы увидите модели с приписками в названиях файлов:.gguf,.safetensors,.bin. Для работы с Ollama и LM Studio нам нужен именно **GGUF (GPT-Generated Unified Format)**.

GGUF — это формат файлов, разработанный специально для llama. cpp (движка, на котором работает большинство локальных инструментов). Его главные преимущества:

1. **Всё в одном:** В одном файле лежат и веса модели, и токенизатор, и метаданные. Не нужно скачивать кучу вспомогательных файлов.

2. **Гибридный запуск:** GGUF позволяет легко разделить модель между видеокартой (VRAM) и процессором (RAM). Если видеопамяти не хватает на всю модель, GGUF загрузит в VRAM самые важные слои, а остальное отдаст в RAM (при этом скорость генерации заметно упадет из-за более медленной системной шины).

3. **Эффективность на CPU:** В отличие от форматов GPTQ или AWQ (которые жестко завязаны на видеокарты NVIDIA), GGUF отлично работает на процессорах.

Юридическая справка: Квантование (перевод модели в GGUF) — это создание производного произведения. Для большинства популярных моделей (DeepSeek, Qwen) это абсолютно легально благодаря либеральным лицензиям (MIT и Apache 2.0). При скачивании моделей с Hugging Face обращайте внимание на имя автора (Organization) — доверяйте только проверенным аккаунтам, чтобы не скачать вредоносное ПО под видом модели.

3. Расшифровываем названия: Q4_K_M и другие страшные буквы

Когда вы открываете список файлов модели на Hugging Face, вы видите пугающие названия: q3_k_s.gguf, q4_0.gguf, q5_k_m.gguf. Что это значит?

— Q (Quantization) + Цифра: Количество бит на параметр. Q8 — почти без потерь, но тяжелая. Q4 — оптимальный баланс. Q2 — сильно сжатая, но модель может начать «глючить» и нести чушь.

— Буквы после цифры:

— _0 — старая версия квантования (устарела).

— _K — современный метод K-quants (умное сжатие, при котором разные слои модели квантуются с разной степенью агрессивности для сохранения логики).

— _S, _M, _L — Small, Medium, Large. Показывают, насколько агрессивно сжаты разные части модели.

Золотой стандарт для домашнего ПК — это `Q4_K_M`. Это сбалансированный вариант, который экономит память, но сохраняет отличное качество ответов (около 95—97% от оригинала).

4. Практика: Как выбрать свой квант?

Ориентируйтесь на объем вашей видеопамяти (VRAM) и оперативной памяти (RAM). Вот шпаргалка:

— 8 ГБ VRAM: Ваш выбор — модели 7B-8B в квантовании **Q4_K_M** (займет около 4.5 ГБ) или **Q5_K_M** (около 5.5 ГБ). Модели побольше (14B) влезут только в Q2 или Q3, но их качество будет низким.

— 12—16 ГБ VRAM: Вы можете запускать 8B в **Q8_0** (максимальное качество) или 14B в **Q4_K_M**.

— 24 ГБ VRAM и выше (или 32+ ГБ RAM): Вам доступны модели 32B в **Q4_K_M** или 14B в **Q8_0**.

— Только CPU (без видеокарты): Скачивайте модели 7B в **Q4_K_M** и наберитесь терпения.

Совет: Не гонитесь за максимальным квантом (Q8), если вам не хватает видеопамяти. Лучше запустить 14B модель в Q4, чем 8B модель в Q8. Большее количество параметров (14B) даст больше «ума», чем высокая точность сжатия (Q8) у маленькой модели.

Блок Troubleshooting: Модель «сломалась»

Проблема: Вы скачали квант Q2 или Q3, а модель начала путать факты, забывать контекст или выдавать код с ошибками.

Решение: Это называется «деградация квантования». Слишком сильное сжатие разрушает логические связи в нейросети. Вернитесь на Hugging Face и скачайте версию **Q4_K_M** или выше.

Проблема: Модель загружается, но скорость падает до 1 токена в секунду, хотя видеокарта мощная.

Решение: Скорее всего, размер контекста (Context Window) слишком велик. Модель в Q4_K_M может занимать 4.5 ГБ, но при длинном промпте она резервирует дополнительные гигабайты под контекст (так называемый **KV-cache**), и в итоге вылезает за пределы VRAM, уходя в медленную RAM. Уменьшите размер контекста в настройках (например, с 8192 до 4096 токенов).

Важно для ИБ: Даже если сжатая модель занимает всего 4 ГБ, при обработке персональных данных (152-ФЗ) убедитесь, что ваш диск (SSD), на котором хранятся веса модели, логи и база документов (RAG), защищен шифрованием и правами доступа. Локальное железо не панацея от утечек, если к нему имеет физический доступ посторонний.

Глава 3. Как это работает под капотом: Токены, Эмбеддинги и Веса

Многие промпт-инженеры и разработчики относятся к локальным моделям как к «черному ящику»: написал запрос — получил ответ. Но когда модель начинает выдавать чушь, обрывать ответы на середине или выдавать ошибку контекста, понимание базовых принципов работы LLM становится критически важным.

В этой главе мы разберем три кита, на которых держится работа нейросети: как текст превращается в токены, как модель понимает смысл (эмбеддинги) и как она вычисляет следующий шаг (веса параметров).

1. Токенизация: Почему русский язык «дороже» английского?

Большие языковые модели не умеют читать буквы или слова так, как это делает человек. Они работают с **токенами** — это кусочки текста, которые могут включать целые слова, части слов или даже отдельные символы.

Процесс разбивки текста на токены называется **токенизацией**. У каждой модели есть свой собственный словарь токенов (BPE — Byte Pair Encoding).

В чем подвох для нас?

Английские слова часто целиком помещаются в один токен. С русскими словами ситуация сложнее: из-за богатства морфологии (приставки, суффиксы, окончания) одно длинное слово может разбиваться на 3—5 токенов.

Пример из практики: Фраза «Привет, как дела?» может разбиться на 7—9 токенов. А её английский аналог «Hello, how are you?» — на 4—6 токенов.

Почему это важно для локального ИИ?

1. **Контекстное окно:** У каждой модели есть лимит токенов (например, 4096, 8192 или 128 000). Чем больше токенов «съедает» русский текст, тем быстрее вы упретесь в этот лимит.

2. **Скорость и память:** Генерация каждого токена — это вычисление. Если ваша русская фраза разбивается на большее количество токенов, модель будет генерировать ответ дольше, а KV-cache (о котором мы говорили в прошлой главе) будет занимать больше видеопамяти.

Совет: Используйте онлайн-токенизаторы (например, от Hugging Face или Tiktokenizer), чтобы визуально видеть, как модель разбивает ваш текст.

2. Векторное представление слов (Эмбеддинги): Как ИИ понимает смысл?

Компьютеры не понимают значения слов, они работают с числами. Чтобы нейросеть могла «понимать» контекст, слова переводятся в математические векторы (наборы чисел с плавающей точкой). Этот процесс называется созданием **эмбеддингов (embeddings)**.

Представьте себе трехмерное пространство, где по осям отложены разные характеристики. В этом пространстве слова с похожим смыслом находятся рядом.

Классический пример, который демонстрирует магию эмбеддингов:

Вектор (Король) — Вектор (Мужчина) + Вектор (Женщина) ≈ Вектор (Королева).

Или:

Вектор (Москва) — Вектор (Россия) + Вектор (Япония) ≈ Вектор (Токио).

Где это используется в локальных LLM?

Эмбеддинги — это фундамент технологии **RAG (Retrieval-Augmented Generation)**, о которой мы подробно поговорим в Части III. Когда вы загружаете свои PDF-документы, они разбиваются на куски (чанки), переводятся в векторы и сохраняются в векторной базе данных. Отличная новость для бизнеса: популярные Embedding-модели (nomic-embed-text, bge-

m3) распространяются под свободными лицензиями (Apache 2.0 и MIT), что позволяет использовать их в коммерческих проектах бесплатно. Важно: для создания эмбеддингов используются именно Embedding-модели, а не сама генеративная LLM (как DeepSeek). Когда вы задаете вопрос, система ищет в базе не по ключевым словам, а по *смыслу* (ближайшие векторы).

Важно для ИБ: Векторизация — это **не анонимизация**. Если в вашу векторную базу (ChromaDB, FAISS) попадают эмбеддинги документов с персональными данными (ПДн), такая база данных подпадает под действие 152-ФЗ и требует соответствующей защиты (шифрование, разграничение прав доступа).

3. Веса параметров: Как модель предсказывает следующее слово?

Мы уже говорили, что модель состоит из миллиардов параметров (весов). Как происходит генерация текста?

Представьте расчет стоимости дома:

Стоимость = Расстояние от МКАД * (вес 1) + Размер участка * (вес 2) + Материал * (вес 3)

На этапе обучения мы даем модели данные (параметры) и правильный ответ (итоговую стоимость). Модель подбирает такие веса, чтобы формула работала идеально.

В LLM происходит то же самое, но в масштабе. Модель получает предыдущие токены (ваш промпт) и должна предсказать **вероятность появления следующего токена**.

Она перебирает миллионы вариантов и выдает тот, который имеет наибольший вес (вероятность) в данном контексте. Затем этот новый токен добавляется к промпту, и процесс повторяется заново. Это называется **авторегрессионной генерацией**. На то, насколько «смело» модель будет выбирать менее вероятные токены, влияет параметр temperature (температура). Чем она выше, тем более креативным (и непредсказуемым) будет ответ.

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, купив полную легальную версию на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.