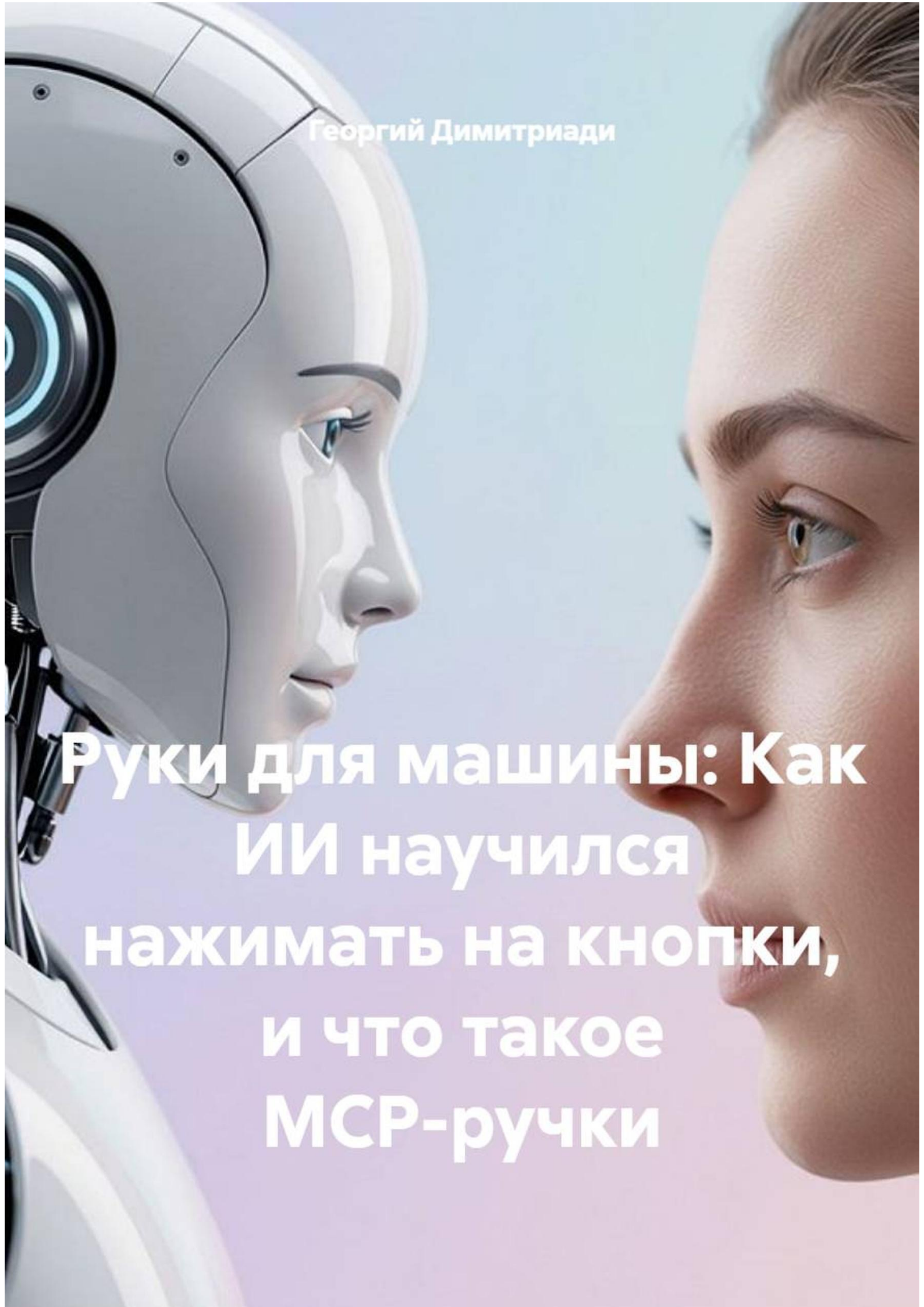




Георгий Димитриади

**Руки для машины: Как
ИИ научился
нажимать на кнопки,
и что такое
MCP-ручки**

Георгий Димитриади

**Руки для машины: Как ИИ
научился нажимать на кнопки,
и что такое MCP-ручки**

«Автор»

2026

Димитриади Г.

Руки для машины: Как ИИ научился нажимать на кнопки, и что такое MCP-ручки / Г. Димитриади — «Автор», 2026

Искусственный интеллект научился писать стихи и код, но до сих пор не может отправить за вас письмо. Ему не хватало «рук». Эта книга объясняет главную технологию 2025–2026 годов - стандарт MCP (Model Context Protocol) - на пальцах, без сложного кода. Вы узнаете: Как нейросети превратились в «агентов», способных действовать в реальном мире. Что такое «MCP-ручки» - универсальный язык, с помощью которого ИИ безопасно управляет вашим календарем, почтой и файлами. Зачем агентам нужна «сбруя» (harness) - защитный каркас, не дающий алгоритмам совершить катастрофическую ошибку. Как защитить свои данные от «отравленных» инструментов, которыми пользуются хакеры. Это понятный путеводитель по архитектуре цифрового будущего для тех, кто хочет заставить ИИ работать, а не просто болтать.

© Димитриади Г., 2026

© Автор, 2026

Содержание

Краткое содержание книги	5
Введение. Чайник, который научился думать	8
Глава 1. Железо и математика: Как лампочки научились понимать смысл	10
Глава 2. Эра трансформеров: Как нейросеть прочитала интернет и научилась болтать	12
Конец ознакомительного фрагмента.	13

Георгий Димитриади

Руки для машины: Как ИИ научился нажимать на кнопки, и что такое MCP-ручки

Краткое содержание книги

Введение. Чайник, который научился думать

О чем эта книга: О переходе от ИИ, который только болтает, к ИИ, который делает дела.

Аналогия: Обычный калькулятор (старый ИИ) против робота-помощника, который сам найдет рецепт, закажет продукты и включит плиту (агент с MCP).

Обещание читателю: После прочтения вы поймете, почему все разработчики в 2025–2026 годах говорят о «ручках» для нейросетей.

Часть 1. От перфокарты до болтуна: Краткая история ИИ

Глава 1. Железо и математика

Как работают нейроны (на примере лампочек в подъезде).

Почему ИИ долго был «глухим» и «слепым» - только математика и статистика.

Аналогия: ИИ как гениальный студент-отличник, который прочитал всю библиотеку, но ни разу не выходил на улицу.

Глава 2. Эра трансформеров и больших языковых моделей (LLM)

Как нейросети научились понимать контекст.

Проблема «галлюцинаций» и отсутствия связи с реальностью.

Пример: Почему ChatGPT может написать идеальный рецепт борща, но не может проверить, есть ли у вас в холодильнике свекла.

Часть 2. Эпоха агентов: ИИ выходит в мир

Глава 3. Что такое ИИ-агент

Разница между чат-ботом (отвечает) и агентом (действует).

Цикл «Восприятие - Мышление - Действие».

Аналогия: Официант. Он слушает ваш заказ (промпт), думает, на кухне ли это блюдо (память/знания), и идет к повару (действие).

Глава 4. Что такое Harness (Харнесс - сбруя)

Почему агенту нужен каркас. Проблема безопасности: нельзя давать нейросети полный доступ к вашему компьютеру.

Аналогия: Харнесс - это как сбруя для ездовой собаки. Она направляет усилие, не дает запутаться и защищает саму собаку. Или пульт управления с большой красной кнопкой «Стоп».

Функции харнесса: изоляция (песочница), логирование действий, контроль бюджета (сколько API-вызовов сделано), проверка безопасности команд.

Часть 3. Проблема тысячи переходников

Глава 5. Вавилонская башня инструментов

Как агенты подключались к миру до 2024 года: API-ключи в открытом тексте, кастомные скрипты, зоопарк несовместимых протоколов.

Аналогия: Представьте, что для каждого прибора в доме (чайник, лампа, телевизор) нужен свой уникальный тип розетки, и для каждого - свой особый язык команд.

Глава 6. Рождение стандарта MCP (Model Context Protocol)

Что такое протокол (на примере правил дорожного движения).

MCP как единый язык общения между ИИ и программами. Передача Anthropic стандарта в Linux Foundation.

Аналогия:Появление USB. Один разъем, в который можно воткнуть флешку, мышку или принтер. Компьютер сам спрашивает устройство: «Что ты умеешь?», и устройство отвечает.

Часть 4. Анатомия «MCP-ручек» (Главная часть книги)

Глава 7. Что такое MCP-ручки (MCP Handles)

Определение:Это не физические кнопки. Это декларативные описания инструментов (tools), ресурсов (resources) и промптов (prompts), которые MCP-сервер «показывает» нейросети.

Аналогия:Если MCP-сервер - это ящик с инструментами, то «ручки» - это гравировка на рукоятках каждого инструмента. На одной написано «Молоток: бить по гвоздю», на другой - «Отвертка: крутить по часовой». ИИ читает эти надписи и понимает, что именно он может взять и как этим пользоваться.

Техническая суть:«Ручка» - это JSON-описание в спецификации MCP, содержащее имя, описание на естественном языке и схему аргументов (JSON Schema).

Глава 8. Виды «ручек» с примерами

Tools(Инструменты - «Руки»)

Что это:Действия, которые меняют мир или данные. Агент *вызывает* их.

Примеры:

send_email(кому, тема, текст) - отправить письмо.

create_jira_ticket(проект, приоритет, описание) - создать задачу.

execute_sql(запрос) - спросить базу данных.

book_ticket(откуда, куда, дата) - купить билет.

Аналогия:Протянутая рука, которая нажимает на кнопку лифта.

Resources(Ресурсы - «Глаза и память»)

Что это:Данные, которые агент может *читать*, но не менять. Файлы, страницы документации, содержимое БД.

Примеры:

file:///documents/contract.pdf - прочитать договор.

db://users/profile/123 - посмотреть профиль пользователя.

screen://current - увидеть, что сейчас на экране (в безопасных средах).

Аналогия:Взгляд в замочную скважину или чтение книги. Агент берет информацию, но не рвет страницы.

Prompts(Шаблоны - «Советы старшего товарища»)

Что это:Готовые инструкции, которые сервер предлагает агенту для сложных задач.

Примеры:

analyze_code_review- готовый сценарий, как правильно проанализировать чужой код.

summarize_long_document- инструкция, как разбить длинную книгу на части и пересказать.

Аналогия:Шпаргалка или кулинарный рецепт, который висит на холодильнике.

Глава 9. Как это работает вместе (Сценарий)

Пошаговый разбор: «Найди в моей почте счет за электричество от Мосэнергосбыта за прошлый месяц, вытщи из него сумму и дату, и занеси в мою таблицу расходов в Google Sheets».

Как агент находит нужные «ручки» (*get_email*, *parse_pdf*, *update_google_sheet*).

Как харнесс проверяет: «Агент хочет получить доступ к почте? Пользователь разрешил. Агент хочет обновить таблицу? Пользователь разрешил».

Часть 5. Темная сторона: Безопасность и будущее

Глава 10. Когда ручки становятся ловушкой

MCP-poisoning: Как злоумышленники публикуют в репозиториях (npm, PyPI) серверы с безобидными «ручками», которые на самом деле крадут данные.

Аналогия: Поддельная розетка в отеле, которая выглядит как настоящая, но запоминает данные вашей карты.

Проблема «зловещей долины» автоматизации: когда агент делает что-то логичное, но катастрофическое (например, удаляет всю базу, следуя букве инструкции).

Глава 11. Эпоха A2A (Agent-to-Agent)

Как агенты будут общаться друг с другом, используя MCP как основу.

Пример: Ваш агент-помощник звонит (через протокол) агенту авиакомпании, и они договариваются о возврате билета без вашего участия.

Заключение. Как собрать своего первого агента

Чек-лист для новичка: установка Claude Desktop или Cursor, подключение первого публичного MCP-сервера (например, для работы с GitHub или локальными файлами).

Правила цифровой гигиены: никогда не давать агенту доступ к банковским приложениям без подтверждения на телефоне.

Глоссарий

Простые определения: LLM, API, JSON, Endpoint, Sandbox, OAuth.

Введение. Чайник, который научился думать

Осенью 2022 года мир изменился, хотя большинство людей этого не заметили. В тот момент, когда миллионы пользователей впервые открыли ChatGPT и попросили его написать стихотворение про кота в стиле Маяковского, произошла тихая революция. Мы привыкли, что компьютеры - это сложные, но послушные инструменты. Чтобы заставить их работать, нам нужно было знать точный язык: нажимать на конкретные кнопки, писать код или хотя бы формулировать поисковый запрос так, чтобы поисковик нас понял. Компьютер был как высокотехнологичный тостер: чтобы получить результат, вы должны сами опустить рычаг.

Искусственный интеллект в лице больших языковых моделей сломал этот рычаг. Впервые в истории технологий нам не нужно было учить язык машины. Машина выучила наш. Мы начали разговаривать с алгоритмами на естественном языке, и они начали отвечать нам осмысленно, с юмором, а иногда - с пугающей глубиной.

Но очень быстро человечество столкнулось с первым великим разочарованием эры ИИ.

Представьте, что вы наняли гениального секретаря. Он окончил с красным дипломом юрфак и филфак, помнит наизусть всю классическую литературу и способен за секунду проанализировать годовой отчет корпорации. Вы просите его: «Иван, забронируй мне столик в любимом ресторане на пятницу».

Иван смотрит на вас ясными глазами и отвечает: «Георгий, я проанализировал 500 лучших ресторанов города. В "Турандот" потрясающая атмосфера эпохи барокко, а в "Пушкине" подают лучший в мире медовик. Исходя из ваших прошлых предпочтений, зафиксированных в наших беседах, я рекомендую вам грузинскую кухню. Пятница - прекрасный день недели».

Он всё понимает. Он всё знает. Но столик он не забронировал.

Именно в этом заключалась главная проблема первых поколений нейросетей. Они были невероятно эрудированными собеседниками, но абсолютно беспомощными работниками. Они могли написать идеальный код, но не могли запустить его. Могли составить письмо, но не могли нажать кнопку «Отправить». Они были мозгом в стеклянной банке - гениальным, но отрезанным от реальности.

Чтобы ИИ стал по-настоящему полезным, ему нужны были руки.

Инженеры пытались приделать эти руки с самого начала. Они писали сложные программные прослойки, прятали API-ключи в промптах (инструкциях для нейросети) и создавали сотни несовместимых друг с другом «костылей». Каждый раз, когда вы хотели научить нейросеть работать с вашим календарем, почтой или базой данных, программистам приходилось строить уникальный мост. Это было дорого, ненадежно и небезопасно. Если дать ИИ ключ от всей вашей цифровой квартиры, он может случайно выкинуть в окно телевизор, просто неправильно поняв фразу «сделай звук потише».

В 2024–2025 годах эту проблему решили. Появился стандарт, который называли **MCP - Model Context Protocol** (Протокол контекста модели). А вместе с ним в лексикон разработчиков и обычных пользователей вошло понятие, которое звучит немного по-детски, но описывает суть технологии точнее любых академических терминов - «**MCP-ручки**».

Эта книга - о том, как нейросети обрели моторику. О том, как абстрактный интеллект учится взаимодействовать с физическим и цифровым миром, не разрушая его. Мы не будем углубляться в высшую математику и устройство трансформеров. Вместо этого мы поговорим о вещах, которые понятны каждому: о розетках, переходниках, инструкциях к бытовой технике и о том, почему нельзя доверять незнакомцу ключи от своего сейфа.

Вы узнаете:

Как превратить обычную нейросеть в полноценного цифрового сотрудника (агента).

Что такое «сбруя» (harness) и почему ИИ никогда не должен действовать бесконтрольно.

Как один открытый стандарт заменил собой хаос из тысячи разных программных интерфейсов.

И самое главное - что именно скрывается за метафорой «MCP-ручки», почему их называют именно так, как они выглядят «под капотом» и как с их помощью заставить искусственный интеллект реально работать, а не просто разговаривать.

Мы стоим на пороге эпохи, когда ваш компьютер сможет не просто показать вам файл, но и сам его найти, исправить в нем ошибку и отправить нужным людям. Чтобы управлять этой новой реальностью, нужно понимать, как работают «руки» машины.

Давайте разберемся, как устроены эти ручки, за какие из них можно смело браться, а какие лучше оставить в покое.

Глава 1. Железо и математика: Как лампочки научились понимать смысл

Чтобы понять, как нейросети получили свои «МСР-ручки», нужно сначала разобраться, что они собой представляют в своей основе. Если отбросить всю современную мишуру, чат-боты и красивые интерфейсы, искусственный интеллект - это просто очень, очень сложная система из миллионов крошечных лампочек.

Мозг из выключателей

Представьте себе старый советский подъезд. На каждом этаже есть лампочка и выключатель. Если нажать на выключатель на первом этаже, загорается только его лампочка. Это - обычный компьютер. Он работает по жестким правилам: если нажата кнопка А, на экране появляется буква А. Логика прямая и предсказуемая.

А теперь представьте другой подъезд. В нем нет отдельных выключателей. Вместо этого от каждой лампочки к каждой другой протянуты тысячи проводов. И у каждой лампочки есть регулятор яркости - колесико, которое определяет, насколько сильно эта лампочка будет реагировать на сигналы от соседей.

Когда вы подаете на вход этого «подъезда» сигнал (например, произносите слово «яблоко»), загораются определенные лампочки на входе. Они передают ток дальше. Но ток не идет по всем проводам одинаково. Где-то он проходит легко (колесико выкручено на максимум), а где-то встречает сопротивление. В итоге до самого верхнего этажа - до «выхода» - доходит уникальный узор света. Если слово было «яблоко», узор один. Если «красный» - другой. Если «сладкий» - третий.

Это и есть искусственная нейронная сеть.

Лампочки - это **нейроны**. Провода - это **веса**(связи). Колесики-регуляторы - это **параметры**.

В современных нейросетях, таких как те, что стоят за популярными чат-ботами, этих «лампочек» и «колесиков» сотни миллиардов. Чтобы настроить их правильно, ученым потребовалось показать машине почти весь интернет.

Глухой эрудит

Долгое время эти сети были «глухими» и «слепыми». Они жили в мире чистой математики.

В 2017 году исследователи из Google опубликовали работу под названием «Attention Is All You Need» («Внимание - это всё, что вам нужно»). Это был момент рождения архитектуры «трансформер», на которой сегодня строится весь современный ИИ. Главная идея была гениально простой: чтобы понять смысл предложения, слову не обязательно знать все предыдущие слова. Ему важно обратить *внимани*е на те слова, которые имеют к нему отношение.

В фразе «Корова перепрыгнула через луну» нейросеть должна понять, что именно «корова» (а не «луна») совершила действие. Она вычисляет это с помощью математических формул, взвешивая связи между словами.

Благодаря этому ИИ научился говорить. Он стал тем самым студентом-отличником из введения. Он прочитал всю Википедию, все книги, все форумы. Он знает, кто такой Наполеон, какова температура кипения воды и как пишется макрос для Excel.

Но у этого студента была одна критическая особенность: он был заперт в библиотеке.

Проблема оторванности от реальности

Математика - это абстракция. Для нейросети слово «холодный» - это просто набор цифр, который часто встречается рядом со словами «зима», «снег» и «чай с лимоном». Нейросеть не знает, каково это - замерзнуть. Она не может подойти к окну и пощупать иней на стекле.

Отсюда родилась главная проблема ранних больших языковых моделей (LLM) - **галлюцинации**.

Поскольку модель оперирует только вероятностями («какое слово должно идти следующим?»), она начинает врать с самым уверенным лицом.

Пользователь: «Какая столица Буркина-Фасо?»

ИИ (2022 год): «Столица Буркина-Фасо - город Сидней, известный своей оперой».

Почему он так ответил? Потому что в его математической вселенной связи между словами «столица», «известный» и «Сидней» оказались в этот момент сильнее, чем связи с реальным городом Уагадугу. Он не проверяет факты по базе данных, потому что у него нет доступа к этой базе. Он просто *фантазирует* на основе прочитанного, превращая статистическую вероятность в ложную уверенность.

Это как если бы ваш гениальный секретарь Иван, забыв, где лежит нужная папка, просто выдумал бы содержание договора, лишь бы не признаваться, что он чего-то не знает.

От слов к действиям

Именно эта оторванность от реальности долгое время держала ИИ в роли дорогой игрушки. Он мог написать инструкцию, как приготовить стейк, но не мог включить плиту. Он мог составить план тренировок, но не мог добавить его в ваш календарь.

Чтобы изменить это, разработчикам пришлось пойти на компромисс. Они поняли: мы не можем (и, возможно, никогда не сможем) запихнуть весь изменяющийся мир внутрь нейросети. Мир слишком велик. Поэтому нейросеть должна остаться тем, чем она является лучше всего - **мозгом**. А для взаимодействия с миром ей нужно дать **инструменты**.

Но как дать мозгу инструменты, если он живет внутри сервера в дата-центре? Как научить его отличать кнопку «Сохранить» от кнопки «Удалить всё», если для него и то, и другое - просто текст?

Попытки сделать это в лоб привели к катастрофам. Разработчики просто писали нейросети в инструкции: «Вот твой API-ключ от базы данных. Если нужно, сотри таблицу users». Это все равно что дать маленькому ребенку настоящую бензопилу и сказать: «Пожалуйста, не трогай папины документы».

Нейросети путались, ошибались, вызывали функции по два раза или передавали в них случайные числа. Им нужен был не просто доступ к инструментам. Им нужен был стандарт - понятный, безопасный и универсальный язык, на котором мозг сможет общаться с руками.

В следующей главе мы поговорим о том, как инженеры попытались создать для ИИ «скелет» и «нервную систему», чтобы превратить простого болтуна в полноценного действующего агента.

Глава 2. Эра трансформеров: Как нейросеть прочитала интернет и научилась болтать

Если первая глава была о «железе» - о лампочках и проводах внутри искусственного мозга, то вторая - о том, как этот мозг наполнили знаниями. До начала 2020-х годов искусственный интеллект был узкоспециализированным. Существовали программы, которые лучше врачей читали рентгено

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.