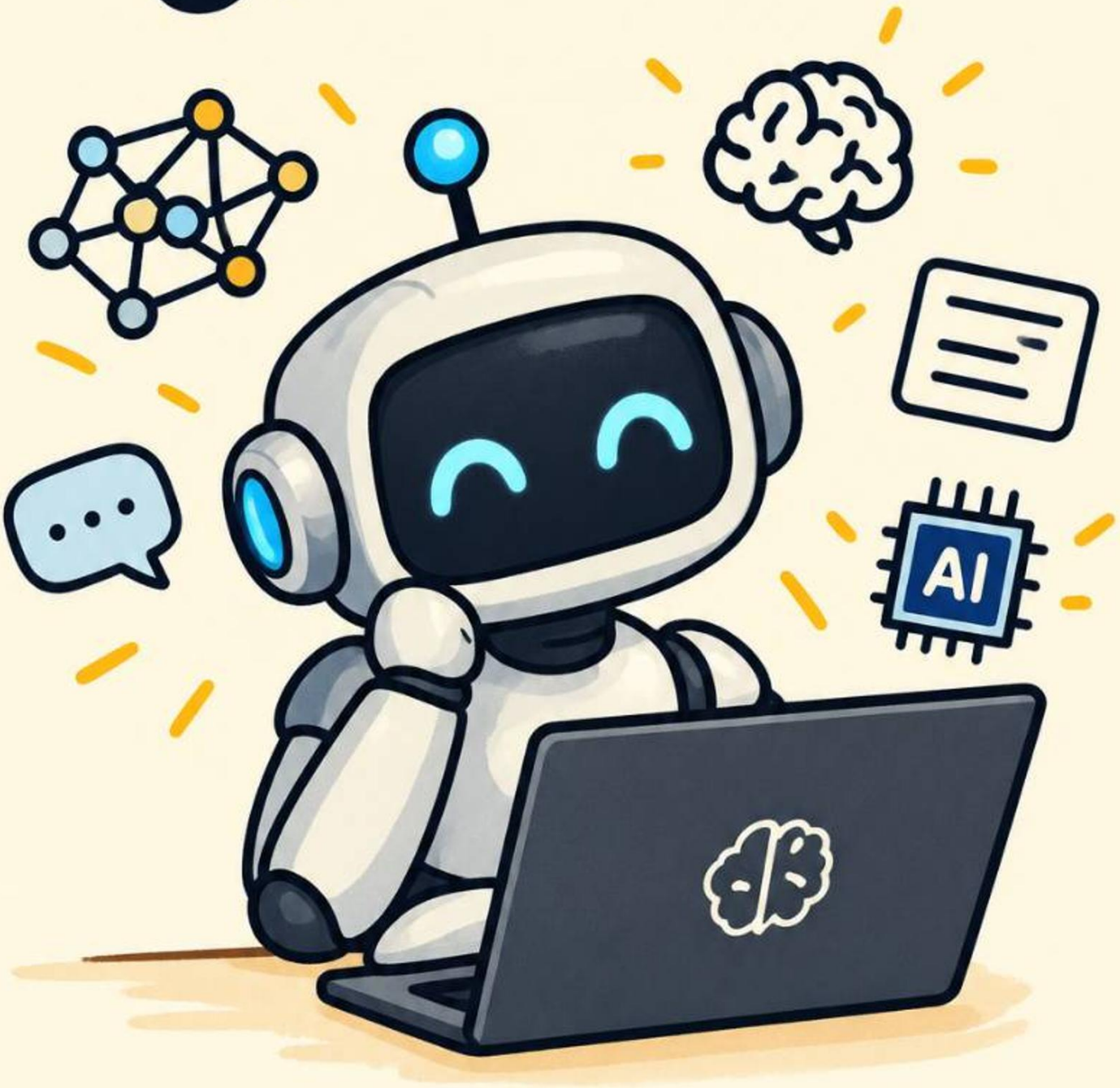
ИИ ЗА ВЕЧЕР



Сергей Лопонос

ИИ за вечер

«Автор»
2026

Лопонос С.

ИИ за вечер / С. Лопонос — «Автор», 2026

Что, если разобраться в искусственном интеллекте можно всего за один вечер?ИИ уже пишет тексты, отвечает на вопросы, создаёт изображения и помогает работать с информацией. Но что на самом деле происходит внутри? Как компьютер учится? Что такое нейронная сеть? Как машина понимает текст и угадывает следующее слово? И почему иногда искусственный интеллект уверенно говорит полную чушь?В этой книге автор простым языком разбирает, как устроен современный искусственный интеллект — от искусственного нейрона и обучения на данных до внимания, трансформеров и ChatGPT.Без сложной математики и необходимости быть программистом. Только понятные объяснения, наглядные примеры и последовательное погружение в тему.«ИИ за вечер» — вторая книга серии «За вечер». После электричества — самое время разобраться в технологии, которая стремительно меняет наш мир.

Содержание

Сергей Лопонос
ИИ за вечер

Глава 1. ИИ — это что вообще такое?

Если спросить человека, что такое искусственный интеллект, можно получить самые разные ответы. Кто-то скажет:

— Это ChatGPT.

Другой:

— Это нейросети.

Третий уверенно заявит:

— Это когда компьютер начинает думать.

А кто-нибудь обязательно добавит:

— Скоро роботы захватят мир.

Последнее, конечно, звучит особенно убедительно. Особенно если человек только что посмотрел фантастический фильм. Но давайте пока оставим восстание машин на потом. Начнём с гораздо более простого вопроса: что вообще такое искусственный интеллект?

Компьютер и раньше был довольно умным

Представьте обычный калькулятор. Вы вводите:

237×481

Он через долю секунды выдаёт ответ. Выглядит впечатляюще. Особенно если попросить посчитать это в уме у человека, который сегодня не выспался. Но калькулятор не является искусственным интеллектом. Почему?

Потому что он не учился умножать. Он не посмотрел на миллион примеров:

$2 \times 3 = 6$

$7 \times 8 = 56$

$12 \times 15 = 180$

и не сделал вывод: «Кажется, я начинаю понимать эту штуку».

Нет.

В него заранее заложили правила, по которым он должен выполнять вычисления. Вы дали числа. Он применил известный алгоритм. Получил ответ. В этом смысле обычная программа очень похожа на хорошего, но чрезвычайно буквального сотрудника.

Вы говорите ей: Если произошло А — сделай Б.

Если произошло В — сделай Г.

Если температура выше 100 градусов — включи вентилятор. И программа послушно выполняет инструкции. Проблема начинается тогда, когда мы хотим дать компьютеру задачу, для которой нам самим трудно сформулировать правила.

Попробуем объяснить компьютеру, что такое кошка

Представим, что мы хотим написать программу, которая будет смотреть на фотографии и определять: кошка перед нами или собака? На первый взгляд задача кажется простой. Человек обычно справляется с ней без особых проблем. Покажите ему:

кошка Он скажет: Кошка. Покажите: собака Собака. Ну и прекрасно. Теперь попробуем объяснить компьютеру, по каким правилам мы это определили. У кошки есть четыре лапы?

У собаки тоже. Есть хвост? Тоже. Есть шерсть? В большинстве случаев — да.

Два глаза? Как правило. Уши? У обеих имеются. Тогда можно сказать:

Если четыре лапы + хвост + шерсть + два глаза = кошка.

Получается кошка. Правда, где-то рядом уже стоит собака и возмущённо смотрит на эту классификацию. Попробуем добавить размер. Кошка обычно меньше собаки. Но есть маленькие собаки.

И большие кошки. Добавим форму ушей. У кошки обычно треугольные уши. Отлично. Только вот есть собаки с очень похожими ушами.

Тогда добавим форму морды. Цвет шерсти. Длину хвоста. Положение глаз. Положение тела.

И ещё несколько сотен признаков. Через некоторое время наша программа превратится в огромную инструкцию примерно такого вида: Если у объекта четыре лапы, шерсть определённого типа, нос такой формы, уши такие, глаза находятся на таком расстоянии, хвост изгибается вот так, а голова имеет вот такую геометрию — скорее всего, это кошка. Звучит разумно.

Но тут мы показываем программе сфотографированного сфинкса. И она впадает в экзистенциальный кризис. Где шерсть?!

А что, если не объяснять правила? Вот здесь начинается самое интересное. Можно попробовать поступить совершенно иначе. Не говорить компьютеру: «Вот тебе правила, по которым нужно узнавать кошку». А показать ему много примеров.

Вот кошка. Вот ещё кошка. Вот эта кошка. Вот эта тоже. А вот собака.

И ещё собака. И ещё. И ещё. Компьютер делает попытку. Ошибается.

Мы говорим: Нет. Это была кошка. Он немного меняет свои внутренние настройки. Получает следующую картинку. Снова ошибается.

Опять корректирует настройки. И так снова, снова и снова. Через огромное количество примеров система начинает находить закономерности. Причём мы не обязаны заранее говорить ей: «Обрати внимание на форму ушей». Она сама может обнаружить, что определённые особенности изображения помогают отличать один объект от другого.

Вот это уже совсем другой подход. Именно здесь появляется машинное обучение.

Искусственный интеллект, машинное обучение и нейросети

Теперь нам понадобятся три термина. Искусственный интеллект (ИИ) — самое широкое понятие. Это название для методов, которые позволяют компьютерам выполнять задачи, обычно связанные с человеческими интеллектуальными способностями: распознавать объекты, понимать текст, принимать решения, находить закономерности, создавать содержимое и так далее.

Машинное обучение — один из способов создать такой интеллект. Вместо того чтобы вручную прописывать все правила, мы даём системе данные и способ учиться на них. Нейронная сеть — один из наиболее мощных инструментов машинного обучения.

Именно нейронные сети лежат в основе огромного количества современных систем ИИ. Можно представить это как матрёшку:

Искусственный интеллект — Машинное обучение — Нейронные сети

Не каждый ИИ — нейронная сеть. Не каждое машинное обучение — нейронная сеть. Но современный бум ИИ во многом связан именно с развитием нейросетей.

Почему именно сейчас все заговорили об ИИ? Ведь искусственный интеллект придумали не вчера. Первые серьёзные исследования в этой области начались ещё в XX веке. Учёные десятилетиями пытались научить машины решать задачи, распознавать изображения, играть в игры и работать с языком.

И периодически случались впечатляющие успехи. А потом наступал очередной период разочарования. Компьютер оказывался не таким уж умным. Почему? Потому что для хорошего обучения нужны как минимум три вещи: данные, вычислительные мощности и хорошие алгоритмы.

Долгое время чего-то из этого не хватало. Представьте, что вы хотите научить ребёнка распознавать собак. Можно показать ему десять фотографий. Он, возможно, что-нибудь поймёт. Покажите сто тысяч — уже лучше.

А теперь представьте, что вместо ста тысяч фотографий у вас огромное количество разнообразных данных, а вместо одного компьютера — вычислительная инфраструктура, способная обрабатывать всё это с колоссальной скоростью. Современный ИИ получил доступ сразу ко всем этим преимуществам. И тут началось веселье.

Нейросети научились гораздо лучше распознавать изображения, работать с речью, переводить языки, писать тексты, создавать изображения и выполнять множество других задач. А потом появились большие языковые модели. И однажды мы обнаружили, что с компьютером можно просто поговорить.

Но разве ChatGPT — это и есть ИИ?

Нет.

Точнее, ChatGPT — это один из примеров применения искусственного интеллекта. Так же как автомобиль — не вся механика, а самолёт — не вся аэродинамика. Когда вы разговариваете с ChatGPT, вы взаимодействуете с системой, построенной на больших нейронных сетях, обученных работать с языком.

Но ИИ вокруг нас гораздо больше. Например: Ваш телефон узнаёт ваше лицо. Камера автоматически улучшает фотографию. Почта определяет, что письмо похоже на спам. Музыкальный сервис предлагает песню, которую вы, возможно, захотите послушать.

Навигатор выбирает маршрут. Сайт пытается угадать, какой товар вас заинтересует. Банк анализирует подозрительные операции. Переводчик переводит текст. Автомобиль помогает удерживаться в полосе.

И всё это может использовать методы искусственного интеллекта. То есть ИИ — это не обязательно человекоподобный робот, который ходит по квартире и спрашивает:

«Сергей, где мой кофе?»

ИИ может выглядеть гораздо скучнее. Например, как строчка кода в огромной серверной системе.

А теперь самое важное

До сих пор мы говорили об ИИ почти как о чём-то разумном. Мы говорили: компьютер учится; компьютер узнаёт кошку; компьютер понимает текст. Это удобные слова. Но здесь легко попасть в ловушку. Компьютер не учится так, как учится человек.

Когда вы изучаете новый предмет, вы можете понять идею, связать её с прошлым опытом, задать вопрос, почувствовать сомнение, изменить мнение. У нейросети всё происходит совершенно иначе. В основе лежит математика. Очень много математики.

Нейросеть получает числа. Выполняет огромное количество операций. Сравнивает результат с тем, который должен был получиться. Меняет свои параметры. И повторяет процесс.

Снова. И снова. И снова. И когда мы говорим: «Нейросеть научилась распознавать кошек», внутри не появляется маленькая цифровая кошка, которая внезапно осознала свою кошачью природу. Меняются числовые параметры системы.

И этих параметров может быть очень много. Очень. Очень. Настолько много, что если попытаться представить их все в виде ручек на пульте управления, у вас закончится место раньше, чем вы доберётесь до первого миллиарда.

И тут возникает странный вопрос

Если внутри всего лишь числа и математические операции... как из этого вообще получается что-то похожее на интеллект? Как система, которая работает с числами, может: узнать кошку; перевести текст; написать рассказ; решить задачу; описать фотографию; создать изображение; написать программу; поддерживать разговор?

И особенно интересно: как она может работать с языком? Ведь для нас слово — это не просто набор символов. Когда вы читаете слово «яблоко», у вас в голове возникает целый набор ассоциаций: красное, зелёное, круглое, сладкое, дерево, сад, яблочный пирог, Ньютон, и, возможно, воспоминание о том, как вы в детстве пытались сорвать яблоко с соседского дерева и совершенно случайно познакомились с владельцем сада.

Для компьютера никакого яблока в голове нет. Есть числа. И каким-то образом эти числа должны превратиться в структуру, с которой можно работать. А потом возникает ещё более удивительная вещь. Нужно научить машину не просто распознавать отдельные слова, а выстраивать между ними связи.

Почему в предложении:

«Маша положила яблоко на стол, потому что оно было тяжёлым»

слово «оно» относится к яблоку? Почему:

«Я зашёл в банк»

может означать совершенно разные вещи в зависимости от контекста? Почему:

«Он забил гол»

и

«Он забил гвоздь»

— это разные действия, хотя используется одно и то же слово?

И наконец: как из всего этого получается ChatGPT? Вот теперь мы подошли к настоящему началу нашей истории.

От калькулятора к искусственному интеллекту

Если очень сильно упростить весь путь, получится примерно такая цепочка: числа — математические операции — искусственные нейроны — нейронные сети — обучение на данных — огромное количество параметров — работа с текстом, изображениями, звуком и другими данными — современные модели ИИ На следующем этапе мы разберём первый настоящий кирпичик этой конструкции. Искусственный нейрон. И не пугайтесь названия.

На самом деле он гораздо проще, чем кажется. Можно даже представить его как очень примитивного охранника, который получает несколько сигналов, взвешивает их и решает:

«Пропускать или нет?»

Правда, охранник этот состоит из математики. А потом мы поставим рядом второго. Потом третьего. Потом тысячу. Потом миллион.

И посмотрим, как из толпы довольно тупых математических охранников постепенно получается система, способная делать вещи, которые снаружи выглядят почти разумными.

Глава 2. Компьютер, который учится на ошибках

В первой главе мы выяснили довольно странную вещь. Искусственный интеллект — это не маленький электронный человечек, который сидит внутри компьютера и размышляет о смысле жизни. В основе всего — математика, данные и огромное количество вычислений.

Но тогда возникает главный вопрос: как именно компьютер учится? Потому что человеку это понятно. Ребёнку показывают собаку:

— Это собака.

Показывают кошку:

— Это кошка.

Показывают снова собаку:

— Собака.

Через некоторое время ребёнок начинает узнавать их самостоятельно. Правда, иногда он может назвать овчарку «лошадью», но это уже отдельная история. А как объяснить компьютеру, что именно он сделал неправильно?

Представим очень глупую машину

Допустим, у нас есть компьютер, который должен отличать кошек от собак. Но пока он ничего не умеет. Показываем ему фотографию: кошка Компьютер смотрит на неё своими электронными глазами и выдаёт: Собака. Мы:

Нет.

Компьютер: Хорошо. Следующая фотография: собака Компьютер: Кошка. Мы: Опять нет. Компьютер: Понятно. Следующая: кошка Компьютер: Собака.

Мы: ТЫ СЕРЬЁЗНО? Компьютер: Я стараюсь. В этом месте человеку очень хочется сказать:

«Да ты просто посмотри внимательнее!»

Но компьютер не знает, как именно посмотреть внимательнее. Ему нужен механизм, который позволит исправлять свои ошибки. И вот здесь начинается обучение.

Ошибка — это не проблема. Это информация

Представьте, что вы стреляете в мишень. Первый выстрел: мимо. Второй: ещё дальше. Третий: почти попали. Что вы делаете?

Меняете прицел. Если выстрел ушёл вправо — немного двигаете прицел влево. Если ушёл вверх — корректируете вниз. И снова стреляете. Через некоторое количество попыток вы начинаете попадать.

Обучение нейросети устроено похожим образом. Она делает предположение. Получает правильный ответ. Сравнивает своё предположение с правильным ответом. Видит ошибку.

И немного меняет свои внутренние параметры. Потом пробует снова.

Но что именно она меняет? Вот здесь появляются веса. Помните нашего искусственного нейрона? Он получает несколько входных сигналов. Например: размер объекта; форма ушей; цвет; форма морды.

Но каждый сигнал для него важен по-разному. Форма ушей может быть очень полезной. Цвет — гораздо менее полезным. Поэтому каждому входу можно назначить определённый вес. Условно:

Уши × 0,8

Форма морды × 0,9

Цвет × 0,1

Размер × 0,4

Это не настоящие значения — просто пример. Вес показывает, насколько сильно данный сигнал влияет на результат. И вот обучение заключается в том, что нейросеть постепенно подбирает эти веса.

Представьте пульт с миллионами ручек

Это, пожалуй, одна из самых полезных картинок для понимания нейросетей. Представьте огромный пульт управления. На нём миллионы ручек. Одна отвечает за один параметр. Другая — за другой.

Третья — за что-то настолько мелкое, что мы даже не знаем, как это назвать. Вы показываете нейросети фотографию. Она выдаёт: 87% — собака. А правильный ответ: кошка. Значит, что-то настроено неправильно.

И система пытается понять, какие ручки и в какую сторону нужно немного повернуть, чтобы в следующий раз результат был лучше. Повернули. Показали следующую фотографию. Снова ошибка. Повернули ещё немного.

Снова проверили. И так огромное количество раз. Причём не один раз в секунду. А миллионы и миллиарды вычислительных операций. Вот это и называется обучением.

Но откуда компьютер знает, насколько он ошибся? Очень хороший вопрос. Представим, что нейросеть должна определить число от 0 до 1. Например:

0 = собака

1 = кошка

Показываем кошку. Правильный ответ:

1

А нейросеть сказала:

0,2

Она сильно промахнулась. Теперь показываем другую кошку. Нейросеть сказала:

0,8

Уже гораздо лучше. Ещё одна:

0,99

Прекрасно. То есть можно ввести специальную математическую величину, которая показывает: насколько результат отличается от правильного. Эту величину называют функцией потерь, или loss function. Название немного драматичное.

Будто нейросеть переживает личную трагедию. На самом деле это просто число, которое говорит:

«Насколько плохо ты сейчас справилась с задачей?»

Чем меньше это число — тем лучше.

Теперь начинается самое интересное

Допустим, нейросеть ошиблась. Мы знаем: результат должен быть другим. Но нужно понять: какие именно веса виноваты в ошибке? В большой нейросети их может быть миллионы, миллиарды и больше. Это всё равно что сказать: «Машина приехала не туда. Найдите, какая из нескольких миллиардов настроек маршрута привела к ошибке».

Вручную это сделать невозможно. Поэтому здесь в игру вступает математика. Система вычисляет, как изменение каждого параметра повлияет на ошибку. И затем немного корректирует параметры. Этот процесс связан с тем, что называют обратным распространением ошибки, или backpropagation.

Звучит так, будто сейчас будет очень сложно. Но идея на самом деле довольно простая. Ошибка появилась на выходе. Мы идём назад по сети и выясняем, какой вклад в неё внесли предыдущие элементы. После этого немного исправляем их.

Представим склон

Есть ещё одна замечательная аналогия. Вы стоите на огромной горе. Вокруг темно. Вам нужно попасть в самую низкую точку долины. Но карту вам никто не дал.

Что делать? Можно сделать маленький шаг. Проверить: Здесь стало ниже или выше? Если ниже — продолжить в этом направлении. Если выше — попробовать другое.

И постепенно спускаться вниз. Именно поэтому один из важнейших методов обучения нейросетей называется градиентным спуском. Мы ищем направление, в котором ошибка уменьшается. Не прыгаем сразу в идеальное решение. Делаем маленькие шаги.

Снова проверяем. И снова.

Почему нельзя просто исправить всё сразу? Потому что нейросеть — это огромная система взаимосвязанных параметров. Представьте оркестр. Если скрипка играет слишком громко, можно просто сделать её тише. Но если вы одновременно меняете громкость всех инструментов, задерживаете барабанщика на полсекунды и заставляете трубача играть в два раза быстрее, музыка может стать... интересной.

Но не обязательно хорошей. В нейросети похожая ситуация. Изменение одного параметра может повлиять на множество последующих вычислений. Поэтому обучение обычно идёт постепенно. Чуть-чуть изменили.

Проверили. Ещё чуть-чуть. Проверили. И снова.

А теперь представим настоящее обучение

Допустим, мы хотим обучить нейросеть распознавать кошек. У нас есть: 10 миллионов фотографий. На каждой фотографии заранее известно, что изображено. Кошка. Собака.

Лошадь. Машина. Человек. Нейросеть получает фотографию. Делает прогноз.

Сравнивает его с правильным ответом. Вычисляет ошибку. Корректирует параметры. Берёт следующую фотографию. И повторяет процесс.

Снова. И снова. И снова. После обработки огромного количества примеров параметры сети постепенно меняются. И результат становится лучше.

В какой-то момент мы показываем ей новую фотографию, которую она никогда раньше не видела. И она говорит: Кошка — 97%. Вот здесь и появляется настоящее ощущение магии. Но никакой магии нет. Она просто обнаружила закономерности, которые оказались полезными для решения задачи.

Самое интересное: мы не говорили ей, что искать

Вот это принципиальный момент. Мы не обязаны программировать: «Кошка имеет два глаза». «Кошка имеет уши». «У кошки есть нос». «Если уши треугольные — это кошка».

Мы просто даём ей примеры. А система сама подбирает внутренние параметры, которые позволяют хорошо разделять категории. И здесь есть один очень важный нюанс. Мы сами не всегда можем сказать, что именно она выучила. Может оказаться, что нейросеть нашла какую-нибудь закономерность, которую человек вообще не заметил.

Например, на фотографиях кошек почему-то чаще оказывается определённый фон. И нейросеть решает: «О! Фон очень полезен». А потом ей показывают кошку на белой стене. И она: «Не уверен. Это, наверное, табуретка».

Вот почему данные для обучения так важны. Но об этом мы поговорим чуть позже.

А можно ли обучить нейросеть чему угодно?

Нет.

И это важно понимать. Нейросеть не волшебный мозг, который можно посадить перед компьютером и сказать: «Ну-ка, разберись во всём». Она зависит от: данных, архитектуры, задачи, качества обучения и вычислительных ресурсов.

Если дать ей плохие данные, она может научиться плохим закономерностям. Если дать мало данных, ей может не хватить информации. Если задача слишком сложная, простой модели может оказаться недостаточно. И ещё одна проблема:

нейросеть может научиться запоминать вместо того, чтобы действительно обобщать.

Когда отличник просто выучил ответы

Представьте школьника. Вы дали ему тысячу задач и правильные ответы. Он заучил все тысячу. На экзамене ему попадаются ровно те же задачи. Он получает: 100 баллов.

Вы говорите: «Вот это умный человек». А потом даёте ему задачу, сформулированную немного иначе. Он смотрит. Молчит. Смотрит ещё минуту.

И наконец: «А этого вопроса в списке не было». С нейросетями может происходить нечто похожее. Если модель просто запомнила обучающие примеры, это ещё не означает, что она научилась хорошо решать задачу. Нам нужно, чтобы она обобщала.

То есть могла применить найденные закономерности к новым данным. Именно поэтому после обучения модель проверяют на примерах, которых она раньше не видела.

А теперь увеличим всё это до безумных масштабов

До сих пор мы говорили о небольшой нейросети. Но современные модели могут быть огромными. Вместо нескольких нейронов — огромное количество вычислительных элементов. Вместо тысячи примеров — гигантские объёмы данных. Вместо нескольких параметров — миллиарды параметров и больше.

И вместо одной задачи «кошка или собака» мы можем поставить гораздо более сложную цель. Например: предсказывать, какой текст должен идти дальше. Вот здесь наша история начинает постепенно приближаться к ChatGPT. Но прежде чем компьютер научится работать с человеческим языком, ему нужно каким-то образом этот язык увидеть.

Для нас текст — это слова и предложения. Для компьютера — числа. Каждое слово, каждый кусочек слова, каждая последовательность превращается в математическое представление. И тогда возникает совершенно удивительная картина.

Мы можем взять огромный объём человеческого текста и превратить его в данные, на которых будет учиться нейросеть. Она будет снова и снова пытаться предсказать правильное продолжение. Ошибка. Корректировка. Следующий пример.

Ошибка. Корректировка. Ещё пример. И так — невероятное количество раз. Постепенно система начинает улавливать закономерности языка.

Не потому, что ей объяснили школьную грамматику. Не потому, что внутри сидит преподаватель русского языка. А потому, что язык сам по себе содержит огромное количество закономерностей.

Но пока мы слишком рано забежали вперёд

Мы уже знаем несколько важных вещей. Первое. Нейросеть не получает готовый интеллект. Она начинает с параметрами, которые сами по себе ничего особенно полезного не делают. Второе.

Мы показываем ей данные и правильные ответы. Третье. Она делает прогноз. Четвёртое. Мы измеряем ошибку.

Пятое. Параметры немного корректируются. Шестое. Процесс повторяется огромное количество раз. Всё.

Вот из таких довольно простых шагов и складывается обучение. Конечно, настоящие системы намного сложнее. Но фундаментальная идея именно такая.

И вот тут появляется маленькая проблема

Мы сказали: «Нейросеть получает данные». Хорошо. Но что такое данные для нейросети? Человек видит фотографию кота. Компьютер не видит кота.

Он видит набор чисел. Человек читает: «Сегодня прекрасная погода». Компьютер не слышит в голове голос автора. Для него это последовательность числовых представлений. А значит, прежде чем мы разберёмся, как ChatGPT работает с текстом, нам придётся ответить на очень странный вопрос:

как превратить смысл, изображение или слово в числа? И вот здесь начинается следующая часть нашего путешествия. Но сначала нам нужно познакомиться с тем самым маленьким математическим кирпичиком, из которого строится вся эта огромная конструкция.

С искусственным нейроном. И обещаю: он окажется гораздо менее страшным, чем звучит его название.

Глава 3. Один искусственный нейрон

Слово «нейрон» звучит так, будто сейчас нам покажут фотографию мозга, достанут микроскоп и начнут рассказывать про какие-нибудь дендриты. Но не волнуйтесь. Сегодня обойдёмся без вскрытия черепа. Нам понадобится кое-что гораздо проще.

Один искусственный нейрон. Это маленький математический элемент, который умеет делать одну очень простую вещь: получать сигналы на входе, обрабатывать их и выдавать результат. Сам по себе он довольно глупый. Настолько глупый, что если оставить его одного, никакого искусственного интеллекта из него не получится.

Но именно из таких простых элементов собираются большие нейронные сети. И здесь начинается самое интересное.

Представим охранника

Допустим, вы управляете ночным клубом. В клуб хотят попасть люди. Ваша задача — решить, кого пускать. У входа стоит охранник. Он получает несколько признаков о человеке: есть ли у него билет; есть ли VIP-браслет; знает ли он пароль;

пытается ли он пройти без очереди. Охранник должен принять решение: пускать или не пускать. Самый простой вариант — придумать правила. Например: Есть билет — пускаем. Но что делать, если билета нет, зато есть VIP-браслет?

Тогда можно сказать: VIP-браслет важнее билета. А если человек говорит: «Меня сюда директор пригласил». Тут уже начинается самое интересное. Охранник может решить, что это очень важная информация. А может ответить: «Конечно. Директор каждый вечер лично встречает гостей у входа».

У каждого сигнала есть свой вес

Представим, что охранник оценивает четыре признака.

Билет: +5

VIP-браслет: +10

Знание пароля: +7

Попытка пролезть без очереди: −8 Почему разные числа? Потому что признаки имеют разную важность. VIP-браслет может быть очень убедительным аргументом. А вот то, что человек стоит в очереди, само по себе почти ничего не говорит.

Эти коэффициенты и напоминают то, что в нейронной сети называют весами. Вес показывает, насколько сильно определённый вход влияет на результат.

Теперь немного математики

Представим, что у нас есть три входа: x_1, x_2, x_3 И три соответствующих веса: w_1, w_2, w_3 Нейрон сначала перемножает каждый вход на его вес:

$x_1 \times w_1$

$x_2 \times w_2$

$x_3 \times w_3$

А потом складывает результаты:

$x_1w_1 + x_2w_2 + x_3w_3$

И к этому может добавиться ещё одно число — смещение, или bias. Получаем:

$z = x_1w_1 + x_2w_2 + x_3w_3 + b$

Вот и почти весь секрет нашего маленького нейрона. Да, серьёзно. Выглядит скромно. А потом из миллиардов подобных математических операций получается ChatGPT. Иногда полезно вспомнить об этом, когда кажется, что искусственный интеллект — это какая-то сверхъестественная технология.

В основе всё ещё сидит арифметика. Просто арифметики стало очень много.

А что такое bias? Смещение, или bias, можно представить как начальную установку. Например, наш охранник по умолчанию никого не пускает. Чтобы пройти, нужно набрать определённое количество баллов. Это и есть примерно такая идея.

Допустим, у нас есть:

билет = 1

VIP = 1

пароль = 1

А веса:

билет = 2

VIP = 5

пароль = 3

И пусть порог равен 5. Если человек имеет билет и знает пароль:

2 + 3 = 5

Проходите. Если только билет:

2

Извините. Клуб продолжает работать без вас. Конечно, настоящий нейрон устроен несколько сложнее. Но принцип очень похож: входы — веса — сумма — решение.

Почему нейрон называется «нейроном»? Потому что идея вдохновлена настоящими биологическими нейронами. В нашем мозге нервные клетки получают сигналы от других клеток, обрабатывают их и передают сигнал дальше. Искусственный нейрон — очень грубая математическая модель этой идеи.

Но важно не перепутать: искусственный нейрон — это не маленькая копия настоящего нейрона. Это скорее математическая карикатура. Как рисунок человека из палочек. Есть голова. Есть руки.

Есть ноги. Но если попытаться устроиться на работу по такому анатомическому портрету, кадровик может немного удивиться.

А теперь самое важное — решение

Мы получили число. Допустим:

z = 4,7

Что с ним делать? Нейрон должен каким-то образом превратить это число в сигнал на выходе. Для этого используется функция активации. Не пугайтесь. Название страшнее самой идеи.

Функция активации отвечает примерно на вопрос:

«Что делать с полученным числом дальше?»

В старых и простых моделях можно было сделать совсем примитивно: Если значение выше порога — 1.

Если ниже — 0. Получается что-то вроде выключателя. Щёлк.

Да.

Нет.

Да.

Нет.

Очень интеллектуально.

Но такой выключатель слишком грубый

Представим:

4,99 — нет

5,01 — да

Хотя значения практически одинаковые. Современные нейронные сети используют более гибкие функции активации. Например, очень известную ReLU. Она делает почти смешную вещь: если число отрицательное — превращаем его в ноль;

если положительное — оставляем как есть.

То есть: −7 — 0 −1 — 0 0 — 0 3 — 3 12 — 12 На первый взгляд кажется:

«И это всё?»

Да.

Но именно из таких простых операций строятся огромные вычислительные системы.

Один нейрон — почти бесполезен

Вот здесь начинается важный момент. Наш нейрон умеет выполнять очень простую математическую операцию. Но представьте, что мы хотим распознавать кошку. Можно дать одному нейрону: яркость изображения; количество красного;

размер; несколько других чисел. И попросить:

«Кошка или не кошка?»

Скорее всего, получится довольно грустный результат. Потому что задача слишком сложная. Нам нужно гораздо больше вычислений. И здесь появляется главный трюк: мы соединяем нейроны между собой.

Один охранник — хорошо. А если их тысяча? Представим огромный клуб. На входе стоит не один охранник, а целая система контроля. Первый смотрит на билет.

Второй — на лицо. Третий — на одежду. Четвёртый — на браслет. Пятый — на список гостей. Десятый вообще непонятно на что смотрит, но работает здесь двадцать лет, поэтому спорить с ним никто не хочет.

Каждый получает определённые сигналы. Каждый выдаёт свой результат. Эти результаты поступают дальше. Следующая группа охранников уже оценивает комбинации признаков. Например:

«Есть билет + знакомое лицо + VIP-браслет».

Ещё дальше другой уровень может сказать: «Похоже, это действительно тот человек, которого мы ожидали». Это уже похоже на слоистую нейронную сеть.

От простого к сложному

Именно здесь появляется одна из самых красивых идей нейронных сетей. Ранние слои могут обнаруживать простые закономерности. Например, на фотографии: границы; линии; контрасты; простые формы. Следующие слои могут объединять их:

круг; угол; изгиб; текстура. Ещё дальше: глаз; ухо; нос; лапа. А ещё дальше: Кошка. Получается своеобразная лестница. Простые признаки — более сложные признаки — объект.

И это очень мощная идея.

Представим фотографию

Возьмём обычную фотографию кота. Для человека: «Кот сидит на диване». Для компьютера изображение сначала можно представить как огромное количество чисел, связанных с пикселями. Нейросеть начинает их обрабатывать. Первый слой может обнаруживать простые особенности.

Следующий — более сложные. Следующий — ещё более сложные. И где-то глубоко внутри огромной сети появляются закономерности, которые помогают сделать вывод: Это кот. Причём мы не обязаны вручную говорить каждому нейрону: «Ты отвечаешь за левое ухо».

«Ты — за правый глаз». «Ты — за пушистость». В процессе обучения сеть сама подстраивает свои веса.

И вот мы возвращаемся к прошлой главе

Помните огромный пульт с миллионами ручек? Вот эти ручки и есть наши параметры. Веса. Смещения. Другие числовые настройки.

В начале обучения они не настроены так, чтобы давать хорошие ответы. Система делает прогноз. Ошибается. Вычисляет ошибку. А затем корректирует параметры.

И снова запускает вычисление. Получается цикл: вход — нейроны — ответ — ошибка — изменение весов — новый ответ. Так постепенно сеть настраивается.

Но как из этого получается что-то сложное? Представим, что у нас есть фотография. Первые нейроны работают с исходными данными. Их результаты передаются дальше. Следующий слой получает уже не исходную фотографию, а результаты работы предыдущего слоя.

Это очень важно. Каждый следующий слой получает всё более обработанное представление. Условно: пиксели — линии — формы — части объектов — объекты — смысловая информация Разумеется, реальная нейросеть не устроена настолько аккуратно и красиво.

Внутри нет маленького отдела: «Господа, сегодня распознаём левое ухо». Но такая картинка помогает понять общий принцип.

А что будет, если нейронов станет очень много? Тогда количество вычислений начинает расти невероятно быстро. Представьте: один нейрон выполняет несколько операций; тысяча нейронов — тысячи; миллион — миллионы. А если у нас много слоёв?

Тогда данные проходят через огромное количество вычислительных операций. И если добавить огромное количество параметров и огромные объёмы данных, получаются современные нейронные сети. Но возникает вопрос: кто будет всё это считать?

Человеческий мозг?

Нет.

Он и так занят тем, чтобы вспомнить, зачем вы только что открыли холодильник. Нужны компьютеры. Причём очень мощные.

Почему здесь нужны видеокарты? Вот тут происходит ещё одна интересная вещь. Для обучения нейросетей нужно выполнять огромное количество похожих математических операций. А графические процессоры — GPU — как раз отлично подходят для массовых параллельных вычислений.

Изначально они создавались прежде всего для того, чтобы быстро считать графику. То есть компьютер мог одновременно обрабатывать огромное количество точек изображения. А оказалось: «Погодите-ка. А ведь нейросетям тоже нужно одновременно считать огромное количество чисел».

И GPU прекрасно подошли для этой работы. Получилось примерно как с трактором: Его сделали для одной задачи, а потом обнаружили, что он ещё и прекрасно умеет таскать очень много всего сразу. И это стало одним из важных факторов стремительного развития современных нейросетей.

Самое удивительное — нейрон не знает, что он делает

Это стоит запомнить. Когда нейросеть распознаёт кошку, внутри нет отдельного нейрона с табличкой:

«ОТВЕТСТВЕННЫЙ ЗА КОШЕК»

И нет начальника отдела: «Коллеги, поступила новая фотография. Приступаем к распознаванию». Каждый отдельный элемент делает относительно простую математическую работу. Сложное поведение возникает из взаимодействия огромного количества простых операций.

Именно это одна из самых важных идей всей книги.

А теперь попробуем собрать всё вместе

У нас есть входные данные. Например, фотография. Она превращается в числа. Числа поступают в первый слой. Нейроны выполняют вычисления: умножают входы на веса, складывают, применяют функцию активации.

Получившиеся значения идут дальше. Следующий слой снова что-то считает. И следующий. И следующий. В конце сеть выдаёт результат.

Например:

Кошка — 0,97
Собака — 0,02
Автомобиль — 0,01

Получается, что нейросеть не «смотрит» на фотографию в человеческом смысле. Она пропускает через себя огромную цепочку математических преобразований. А на выходе получается число, которое можно интерпретировать как прогноз.

И вот здесь появляется очень важная мысль

Мы начали эту главу с одного простого нейрона. Он умел: получить несколько чисел — умножить их на веса — сложить — выдать результат. Всё. Ничего особо впечатляющего. Но если соединить огромное количество таких элементов в сложную архитектуру, обучить её на огромном количестве данных и дать ей достаточно вычислительных ресурсов, возникают системы, способные распознавать изображения, переводить языки, писать тексты и решать множество других задач.

Это примерно как если бы вы узнали, что самый маленький кирпичик небоскрёба — обычный кусок бетона. Сам по себе он ничего особенного. Но попробуйте собрать из них стоэтажное здание.

И всё-таки одного кирпичика мало

Мы уже знаем, из чего состоит отдельный нейрон. Знаем, что у него есть входы, веса, смещение и выход. Знаем, что нейроны можно объединять в слои. Знаем, что во время обучения их параметры постепенно изменяются. Но остаётся главный вопрос:

как из этой огромной системы получается способность решать сложные задачи? Почему одни слои могут обнаруживать простые закономерности, а другие — гораздо более сложные? Как сеть понимает, какие признаки важны? И главное:

как научить всю эту конструкцию не просто считать, а чему-нибудь действительно полезному? Для этого нам придётся посмотреть на то, как данные проходят через целую нейронную сеть. И вот там один нейрон перестанет быть одиноким.

Мы соберём из них настоящую команду. Правда, как это обычно бывает с большими командами, сначала всё пойдёт немного не по плану.

Глава 4. Нейронная сеть — когда один нейрон уже не справляется

В прошлой главе мы познакомились с искусственным нейроном. Он оказался не таким страшным, как обещало название. На вход ему приходят числа.

Каждому числу назначается свой вес.

Всё это складывается, пропускается через функцию активации — и получается результат.

Примерно так: входы — веса — сумма — активация — результат. Возникает закономерный вопрос: А если один нейрон умеет делать совсем простую вещь, как из них получается что-то, что распознаёт фотографии, переводит текст и пишет вполне убедительные ответы? Ответ простой: нейронов становится много. Очень много.

Но просто свалить миллион нейронов в одну кучу недостаточно. Нужна структура. И вот тут появляется нейронная сеть.

Представьте фабрику

Представим огромную фабрику. На вход приезжает машина с деталями. Первый цех смотрит на самые простые свойства:

— размер такой;

— форма такая;

— поверхность такая.

После обработки детали отправляются во второй цех. Там уже смотрят:

— эти две детали соединяются;

— здесь получается узел;

— тут есть характерная форма.

Потом третий цех:

— похоже на двигатель.

Четвёртый:

— нет, не просто двигатель, а двигатель определённого типа.

И в конце система выдаёт: «Это вертолётный двигатель». Ни один рабочий на фабрике не знает всей картины целиком. Каждый занимается своим маленьким участком. Но вместе они могут решить гораздо более сложную задачу. Нейронная сеть устроена примерно по такому принципу.

Слои: этажи нейронной сети

Нейроны объединяют в группы, которые называют слоями. Самая простая схема выглядит примерно так: вход — слой нейронов — слой нейронов — слой нейронов — выход Можно представить многоэтажное здание. На первом этаже информация только поступила.

На втором её немного обработали. На третьем обработали ещё сильнее. И так далее. В результате на последнем этаже принимается решение. Конечно, настоящая нейронная сеть устроена гораздо сложнее, чем офисное здание.

Но аналогия полезная. Потому что главный принцип именно такой: каждый следующий слой работает с результатом предыдущего.

А что именно происходит с информацией? Возьмём фотографию кота. Для человека всё очевидно. Мы смотрим: «Кот. Ну кот и кот».

Компьютеру так не повезло. Для него фотография — это набор чисел. Например, значения яркости и цвета отдельных пикселей. Условно:

12,18,24,240,197,103...

И таких чисел может быть миллионы. Первый слой получает эту кашу из чисел и начинает искать простейшие закономерности. Например:

— здесь есть граница между светлым и тёмным;

— здесь линия идёт примерно под таким углом;

— здесь меняется цвет.

Сам по себе такой признак ещё ничего не означает. Линия — это просто линия. Но дальше становится интереснее.

От линий к коту

Допустим, один слой научился замечать простые линии. Следующий может использовать эти линии, чтобы обнаруживать более сложные формы. Например: линии — углы — контуры — формы. Следующий слой уже может собрать из форм что-то похожее на:

глаз. Другой — на: ухо. Ещё один может обнаружить: морду. А следующий начинает видеть сочетание:

два глаза + уши + характерная форма головы + шерсть.

И где-то в конце сети появляется очень полезный результат: «Похоже, это кот». Причём никто не сидел внутри компьютера и не писал: Если есть два глаза, два уха, нос и шерсть — объявить кота. Система сама во время обучения настраивает огромное количество параметров так, чтобы постепенно научиться выделять полезные признаки.

И вот здесь мы возвращаемся к предыдущей главе.

Сеть не знает, что такое «ухо»

Это очень важный момент. Когда мы говорим:

«Первый слой распознаёт линии, второй — формы, третий — глаза...»

это удобное объяснение. Но не нужно представлять, что внутри сети действительно сидят аккуратные таблички: Нейрон №1847 — отвечает за левое ухо кота. Так обычно не работает. В реальной сети информация распределена гораздо сложнее.

Один и тот же нейрон может участвовать в обнаружении нескольких разных признаков. А один сложный признак может зависеть от работы огромного количества нейронов. Это больше похоже на оркестр. Скрипач №17 не является «ответственным за грусть».

Но когда сотни музыкантов играют вместе, возникает музыка. Так же и здесь: сложное поведение возникает из взаимодействия множества простых операций.

А где здесь обучение? Вот теперь можно соединить две предыдущие главы. У нас есть нейроны. Мы соединили их в сеть. Между нейронами есть связи.

А у каждой связи есть свой вес. И вот этих весов становится очень много. Очень. Если в маленькой сети несколько тысяч параметров, это уже немало. В современной большой модели их могут быть миллиарды и больше.

И каждый такой параметр — маленькая настройка. Можно снова вспомнить огромную панель управления с ручками. Только теперь ручек не десять. И не тысяча. А миллионы или миллиарды.

Конец ознакомительного фрагмента.