СОДЕРЖАНИЕ

ВСТУПЛЕНИЕ

Глубокое обучение открывает дорогу инновациям и изменениям во всех сферах современной жизни. Большинство прорывов в области искусственного интеллекта, о которых вы узнаете из новостей, основаны на глубоком обучении. Очевидно, что понимание этого предмета необходимо и предпринимателю, заинтересованному в повышении эффективности своей организации, и политику, обеспокоенному этикой и приватностью в мире больших данных, и исследователю, работающему со сложной информацией, и просто любопытному обывателю, который хочет иметь представление о потенциале искусственного интеллекта и его влиянии на жизнь обычных людей.

Цель этой книги — дать широкому кругу читателей возможность разобраться в том, что такое глубокое обучение, откуда оно взялось, как работает, на что способно (а на что нет) и какова вероятная траектория его развития в ближайшие десять лет. Глубокое обучение представляет собой набор алгоритмов и моделей, а это значит, что для его понимания необходимо ориентироваться в том, как эти алгоритмы и модели обрабатывают данные. Поэтому данная книга не носит чисто описательный характер, в ней, помимо прочего, объясняется работа алгоритмов. Я попытался представить этот технический материал в доступной форме. Как показывает мой опыт преподавания, технические темы лучше всего усваиваются путем пошагового объяснения основополагающих концепций. Я старался как можно меньше обращаться к математике, но там, где без математических уравнений было не обойтись, стремился преподнести их настолько ясно и просто, насколько мог. Свои объяснения я дополнил примерами и иллюстрациями.

Замечательное свойство глубокого обучения состоит не в сложности его математических концепций, а в том, как с помощью таких простых вычислений ему удается выполнять настолько разнообразный спектр захватывающих и впечатляющих задач. Не стоит удивляться этой простоте. На самом деле модель глубокого обучения — это всего лишь большое (очень большое) количество умножений и сложений с добавлением нелинейных отношений (которые я тоже объясню). Несмотря на это, такие модели способны, к примеру, побеждать чемпионов мира по игре в го, держать пальму первенства в компьютерном зрении и машинном переводе, водить автомобили и т. д. Эта книга служит только введением в глубокое обучение, но я надеюсь, что она достаточно глубокая и для того, чтобы позже, лучше овладев данным материалом, вы могли еще не раз к ней вернуться.

БЛАГОДАРНОСТИ

Появление этой книги стало возможным благодаря жертвам, на которые пошли моя жена Афра и другие члены моей семьи, включая моих родителей, Джона и Бетти Келлехер. Я также получил огромную поддержку от своих друзей, особенно от Алана МакДоннела, Ионелы Лунгу, Саймона Добника, Лоррейн Бирн, Ноела Фитцпатрика и Джозефа ван Генабита.

Я бы также хотел упомянуть о помощи, оказанной работниками издательства MIT Press и целым рядом людей, которые прочитали отдельные разделы этой книги и поделились своими впечатлениями. MIT Press пригласило трех анонимных рецензентов, которые прочитали и прокомментировали мой черновик; я благодарен им за потраченное время и их полезные отзывы. С черновыми вариантами отдельных глав ознакомилось немало людей, и я хотел бы воспользоваться этой возможностью, чтобы публично отметить их вклад; поэтому выражаю признательность Майку Диллинджеру, Магдалене Кацмайор, Элизабет Келлехер, Джону Бернарду Келлехеру, Афре Керр, Филипу Клубичке и Абхиджиту Махалункару. Содержание этой книги опирается на множество бесед о глубоком обучении, которые я имел со своими коллегами и студентами, особенно с Робертом Россом и Джанкарло Салтоном.

Я посвящаю эту книгу моей сестре Элизабет (Лиз) Келлехер в знак признания ее любви и поддержки, а также за ее терпение по отношению к брату, который без конца что-то объясняет.

ГЛАВА 1

ВВЕДЕНИЕ В ГЛУБОКОЕ ОБУЧЕНИЕ

Глубокое обучение — это подраздел искусственного интеллекта, посвященный моделированию крупных нейронных сетей, которые способны принимать верные *решения на основе данных*. Глубокое обучение особенно хорошо себя проявляет в задачах со сложными данными и огромными массивами информации. На сегодняшний день оно применяется в большинстве интернет-компаний и высококачественных потребительских технологий. Например, Facebook использует его, помимо прочего, для анализа текста в онлайн-переписках. Google, Baidu и Microsoft выполняют с его помощью поиск по изображениям и машинный перевод. Системы глубокого обучения присутствуют во всех современных смартфонах; например, они стали стандартной технологией для распознавания речи, а также для обнаружения лиц в цифровых камерах. В сфере здравоохранения они применяются для обработки медицинских изображений (рентгеновских снимков, результатов компьютерной магнитно-резонансной томографии) и для диагностики состояния здоровья. Глубокое обучение также является ключевым элементом беспилотных автомобилей, отвечая за локализацию и сопоставление, планирование движения, рулевое управление и восприятие окружающей обстановки, а также за отслеживание состояния водителя.

Возможно, самым известным примером глубокого обучения является AlphaGo* от DeepMind. Го — это настольная игра, похожая на шахматы. AlphaGo стала первой компьютерной программой, сумевшей победить профессионального игрока. В марте 2016 года она обыграла ведущего корейского профессионала Ли Седоля в матче, за которым следило больше двухсот миллионов человек. А в 2017 году она выиграла у китайца Кэ Цзе, находившегося на вершине мирового рейтинга.

---

* https://deepmind.com/research/alphago/. — *Здесь и далее — прим. авт.*

В 2016 году успех AlphaGo стал большим сюрпризом. На тот момент большинство людей считали, что компьютеры смогут на равных соперничать с игроками высшего уровня лишь спустя много лет. Уже давно было известно, что запрограммировать компьютер для игры в го намного сложнее, чем для игры в шахматы. В го намного больше потенциальных комбинаций. Это обусловлено тем, что го имеет бóльшую доску и более простые правила. На самом деле возможных комбинаций в го больше, чем атомов во вселенной. Такое громадное пространство поиска и высокая степень ветвления (количество состояний доски, которых можно достичь за один ход) делает го невероятно сложной игрой — как для людей, так и для компьютеров.

Чтобы проиллюстрировать разницу в сложности между го и шахматами с точки зрения компьютерных программ, можно сравнить историю поединков между людьми и компьютерами в этих двух играх. В 1967 году шахматная программа MacHack-6, разработанная в MIT, успешно соперничала с живыми игроками и имела рейтинг Эло* куда выше начального уровня. В мае 1997 года компьютер DeepBlue сумел обыграть чемпиона мира Гари Каспарова. Для сравнения: первая полноценная программа для игры в го была написана лишь в 1968 году, а в 1997-м сильные игроки по-прежнему легко обыгрывали компьютеры.

Такое отставание отражает разницу в вычислительной сложности между этими двумя играми. Тем не менее второй пример демонстрирует, насколько революционным оказалось влияние глубокого обучения на конкурентоспособность компьютеров при игре в го. Шахматным программам потребовалось 30 лет, чтобы достичь уровня чемпиона мира. Но благодаря глубокому обучению программы для игры в го всего за семь лет преодолели разрыв между опытным любителем и лучшим профессиональным игроком. Повышение эффективности оказалось совершенно экстраординарным, и оно свидетельствует также о прогрессе, который глубокое обучение принесло во многие отрасли.

AlphaGo использует глубокое обучение для оценки состояний доски и для выбора следующего хода. Этот факт помогает понять, почему глубокое обучение приносит значительную пользу во множестве разных предметных областей. Принятие решений играет ключевую роль в нашей жизни. Решения можно принимать «интуитивно», доверившись своему «чутью». Но, наверное, большинство людей согласятся с тем, что решения лучше всего основывать

---

* Рейтинговая система Эло — это метод определения уровня игроков в играх с нулевой суммой, таких как шахматы. Она названа в честь ее создателя, Арпада Эло.

на соответствующей информации. Глубокое обучение позволяет делать это за счет определения и извлечения закономерностей из огромных наборов данных, устанавливая правильные связи между цепочкой сложных входных значений и удачными решениями.

Искусственный интеллект, машинное обучение и глубокое обучение

Глубокое обучение стало результатом исследований в области машинного обучения и искусственного интеллекта. Отношения между этими тремя направлениями проиллюстрированы на рис. 1.1.

Искусственный интеллект как раздел компьютерных наук возник в ходе семинара в Дартмутском колледже летом 1956 года. Там были представлены исследования в разных областях, включая доказательство математических теорем, анализ естественного языка, планирование в играх, компьютерные программы, способные обучаться на примерах, и нейронные сети. Современная область машинного обучения основана на последних двух направлениях.

Рис. 1.1. Отношения между искусственным интеллектом, машинным обучением и глубоким обучением

Машинное обучение подразумевает разработку и оценивание алгоритмов, которые позволяют компьютеру извлекать функции из набора данных (то есть учиться на ряде примеров). Чтобы понять, как это работает, необходимо сначала разобраться с тремя терминами: набор данных, алгоритм и функция.

Глубокое обучение позволяет принимать решения на основе данных за счет определения и извлечения закономерностей из огромных наборов информации, устанавливая правильные связи между цепочкой сложных входных значений и удачными решениями.

В простейшем виде набор данных представляет собой таблицу, каждая строка которой содержит описание одного примера из соответствующей предметной области*, признаки которой разделены по столбцам. Например, в табл. 1.1 показан демонстрационный набор данных из сферы займов. В нем содержатся подробности о заявках от четырех потенциальных заемщиков. Если не считать поля ID, которое нужно лишь для удобства, каждый случай характеризуется тремя значениями: годовым доходом претендента, его текущей задолженностью и его кредитоспособностью.

Таблица 1.1. Набор данных о потенциальных заемщиках и их кредитоспособности

ID	Годовой доход	Текущая задолженность	Кредитоспособность
1	$150	–$100	100
2	$250	–$300	–50
3	$450	–$250	400
4	$200	–$350	–300

Алгоритм — это процесс (рецепт, программа), которому может следовать компьютер. В контексте машинного обучения алгоритм определяет процедуру анализа набора данных и выделяет повторяющиеся в этом наборе закономерности. Например, он может найти закономерность между годовым доходом и текущей задолженностью, с одной стороны, и кредитоспособностью — с другой. В математике подобного рода отношения называют функциями.

Функция — это детерминистическое отношение между входными значениями и одним или несколькими выходными. Детерминированность этого отношения означает, что любой конкретный ввод всегда генерирует один и тот же вывод. Например, сложение является детерминистическим отношением: 2+2 всегда равно 4. Как вы позже увидите, функции можно создавать не только для элементарной арифметики, но и для более сложных предметных областей. Например, мы можем определить функцию, которая принимает в качестве ввода доход и задолженность человека и выдает на выходе его кредитоспособность. Понятие функции играет очень важную роль в глубоком

* Под предметной областью мы имеем в виду проблему или задачу, которую пытаемся решить с помощью машинного обучения. Это может быть, например, фильтрация спама, предсказание цен на недвижимость или автоматическая классификация рентгеновских снимков.

Функция — это детерминистическое отношение между входными значениями и одним или несколькими выходными.

обучении, поэтому стоит еще раз повторить ее определение: функция — это просто отношение между вводом и выводом. По существу, целью машинного обучения является получение функций из данных. Функция может быть представлена множеством разных способов: как простая арифметическая операция (например, сложение и вычитание принимают ввод и возвращают одно выходное значение), последовательность правил *if-then-else* или нечто куда более сложное.

Функцию можно представить с помощью нейронной сети. Глубокое обучение является подразделом машинного обучения, посвященным глубокому моделированию нейронных сетей. Закономерности, которые алгоритмы глубокого обучения извлекают из наборов данных, являются функциями, представленными в виде нейронных сетей. Структура нейронной сети проиллюстрирована на рис. 1.2. Прямоугольники слева обозначают ячейки памяти, в которых находится ввод нейронной сети. Каждый кружок на этой диаграмме обозначает нейрон, а каждый нейрон реализует функцию: он принимает на вход ряд значений и связывает их со своим выводом. Стрелки показывают, как вывод одного нейрона подается на вход другому. В этой сети информация передается слева направо. Например, слева ей на вход можно подавать доход и задолженность, а показатель кредитоспособности будет возвращаться из крайнего правого нейрона.

Для выведения функций нейронная сеть использует стратегию «разделяй и властвуй»: каждый ее нейрон формирует простую функцию, которая в совокупности с результатами других нейронов формирует общую (более сложную) функцию. То, как нейронная сеть обрабатывает информацию, будет описано в главе 3.

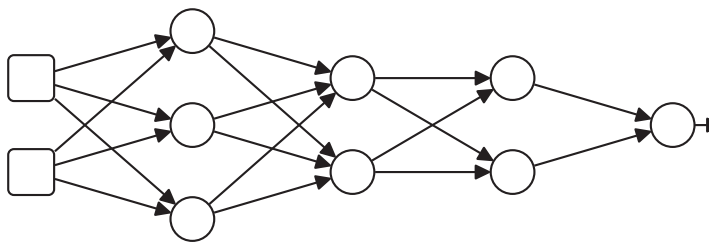

Рис. 1.2. Схематическое изображение нейронной сети

Что такое машинное обучение?

Алгоритм машинного обучения — это поисковый процесс, предназначенный для выбора из ряда потенциальных вариантов функции, которая лучше всего объясняет отношения между признаками набора данных. Чтобы получить

представление о том, как происходит извлечение функции из набора данных (то есть обучение), взгляните на следующий список входных и выходных значений для неизвестной функции. Определите на основе этих примеров, какая арифметическая операция (сложение, вычитание, умножение или деление) лучше всего объясняет отношение, которое устанавливается неизвестной функцией между вводом и выводом:

функция (ввод) = вывод
функция (5, 5) = 25
функция (2, 6) = 12
функция (4, 4) = 16
функция (2, 2) = 04

Большинство людей скажут, что лучше всего здесь подходит умножение, так как оно наиболее точно соответствует наблюдаемому отношению между вводом и выводом:

$5 \times 5 = 25$
$2 \times 6 = 12$
$4 \times 7 = 28$
$2 \times 2 = 04$

В этом конкретном случае выбор лучшей функции выглядит относительно просто, и человек может сделать его без помощи компьютера. Но с увеличением количества входных значений (иногда до сотен и тысяч) и разнообразия потенциальных функций, среди которых нужно выбирать, эта задача существенно усложняется. Именно в этом контексте мощь машинного обучения становится необходимой при поиске лучшей функции для сопоставления закономерностей в наборе данных.

Процесс машинного обучения состоит из двух этапов: собственно обучения и формирования логического вывода. На первом этапе алгоритм обрабатывает набор данных и выбирает функцию, которая лучше всего соответствует выявленным закономерностям. Затем эта функция кодируется в виде компьютерной программы (последовательности правил if-then-else или параметров определенного уравнения). Это так называемая модель, а анализ данных с целью извлечения функции часто называют обучением модели. В сущности, модель — это функция, закодированная в виде компьютерной программы. Однако в машинном обучении понятия модели и функции

настолько близки, что отличиями между ними зачастую пренебрегают и используют их как синонимы.

В глубоком обучении функции, извлекаемые из наборов данных в ходе их анализа, представлены в виде моделей нейронных сетей. Иными словами, модель нейронной сети кодирует функцию в виде компьютерной программы. Обычно в начале обучения нейронная сеть инициализируется с использованием случайных параметров (о параметрах нейронной сети мы поговорим позже; пока можете считать их значениями, которые влияют на поведение кодируемой функции). Такая сеть очень плохо сопоставляет различные входные значения и заданный ввод в предоставленном наборе данных. В процессе обучения она анализирует доступные примеры и в каждом случае сравнивает свой собственный вывод с результатом, указанным в наборе данных, корректируя свои параметры так, чтобы получить более близкое соответствие. Найдя функцию, которая выдает достаточно точное решение (с точки зрения того, насколько близок ее вывод к правильным результатам, указанным в наборе данных) для нашей проблемы, алгоритм машинного обучения завершает работу и возвращает итоговую модель. На этом этап обучения заканчивается.

С этого момента модель остается неизменной. Второй этап машинного обучения — формирование логических выводов. Это необходимо в том случае, когда модель применяется к новым примерам, правильные результаты для которых нам неизвестны, и мы хотим, чтобы она их для нас сгенерировала. Большая часть работы в машинном обучении заключается в формировании точной модели (то есть извлечении из данных подходящей функции). Это вызвано тем, что широкомасштабное применение обученной модели в промышленной среде для дальнейшего получения логических выводов из новых примеров требует навыков и методик, которыми большинство специалистов по анализу данных попросту не владеют. В компьютерной индустрии все шире распространяется понимание того, что для широкомасштабного применения систем искусственного интеллекта требуется особая квалификация, и это выражается в растущем интересе к области под названием DevOps. Данный термин выражает необходимость в сотрудничестве между разработчиками и командами эксплуатации/администрирования (последние отвечают за применение масштабирования готовых систем в промышленных условиях, а также за обеспечение их стабильной работы). Для описания обязанностей по применению обученных моделей также используют термины MLOps (machine learning operations — обслуживание машинного обучения) и AIOps (artificial intelligence operations — обслуживание искусственного интеллекта). Вопросы, относящиеся к процессу использования систем, выходят за рамки

этой книги, поэтому мы сосредоточимся на том, что такое глубокое обучение, для чего его применяют, как оно развивалось на протяжении своей истории и каким образом можно генерировать точные модели глубокого обучения.

Насущный вопрос в этом контексте звучит так: какую пользу приносит извлечение функций из данных? Дело в том, что извлеченную функцию можно применить к незнакомому набору данных, и значения, которые она при этом выдаст, помогут выработать правильные решения для этих новых задач (то есть ее можно использовать для получения логических выводов). Как вы помните, функция — это всего лишь детерминистическое отношение между вводом и выводом. Однако за простотой этого определения кроется разнообразие, присущее функциям. Рассмотрим следующие примеры.

- Фильтрация спама — это функция, которая принимает на вход электронное письмо и выдает значение, определяющее, является ли это письмо спамом.
- Распознавание лиц — это функция, которая принимает на вход изображение и помечает пиксели, относящиеся к лицу.
- Предсказание генов — это функция, которая принимает на вход последовательность ДНК в геноме и выделяет участки ДНК, которые являются кодирующими.
- Распознавание речи — это функция, которая принимает на вход аудиозапись и выдает текстовую транскрипцию того, что на ней слышно.
- Машинный перевод — это функция, которая принимает на вход предложение на одном языке и выдает его эквивалент на другом.

Машинное обучение стало настолько важным в последние годы именно благодаря тому, что огромное количество решений для множества задач в различных предметных областях можно представить в виде функций.

С чем связана сложность машинного обучения?

Существует ряд факторов, которые делают задачу машинного обучения сложной, даже несмотря на использование компьютеров. Во-первых, в большинстве наборов данных есть так называемый шум*, поэтому поиск функции,

* Понятие «информационный шум» означает поврежденные или неправильные данные. Он может быть вызван неисправными датчиками, ошибками при вводе данных и т. д.

которая в точности соответствует данным, не всегда оказывается лучшей стратегией, поскольку подразумевает обучение этому шуму. Во-вторых, часто бывает так, что количество возможных функций превышает количество примеров в наборе данных. Это означает, что задача, стоящая перед машинным обучением, сформулирована неправильно: предоставленной информации недостаточно для нахождения *единого* лучшего решения; вместо этого данным будут соответствовать несколько потенциальных решений. Чтобы это проиллюстрировать, рассмотрим такую задачу, как выбор арифметической операции (сложения, вычитания, умножения или деления): она лучше всего соответствует отношениям между вводом и выводом неизвестной функции. Вот пример отношений для этой задачи выбора функции:

функция (ввод) = вывод
функция (1,1) = 1
функция (2,1) = 2
функция (3,1) = 3

Умножение и деление подходят для этих примеров лучше, чем сложение и вычитание. Тем не менее на основе этого набора данных нельзя точно определить, какая именно операция играет роль неизвестной функции — умножение или деление, так как обе они соответствуют всем примерам. Следовательно, задача поставлена неправильно: ее условие не позволяет выбрать один лучший ответ.

Для решения таких задач можно попробовать собрать больше данных (больше примеров) в надежде на то, что это поможет отличить подходящую функцию от всех остальных альтернатив. Однако зачастую такую стратегию невозможно применить, поскольку дополнительные данные либо недоступны, либо требуют слишком больших усилий для сбора. Чтобы преодолеть этот недостаток, присущий задачам машинного обучения, алгоритмы пытаются дополнить имеющиеся данные рядом предположений о том, какими характеристиками должна обладать лучшая функция (или модель), и таким образом корректируют свой выбор. Эти предположения называются индуктивным сдвигом алгоритма, поскольку в логике процесс формирования общего правила на основе конкретных примеров известен как индуктивное умозаключение. Например, если все лебеди, которые вам когда-либо встречались, были белыми, вы можете сделать из этого логический вывод о том, что *все лебеди белые*. Эта концепция важна в данном контексте, потому что алгоритм машинного обучения формирует (или извлекает) общее правило (функцию) из набора конкретных примеров (набора данных). Следовательно, предположения, влияющие на этот алгоритм,

фактически корректируют процесс индуктивного умозаключения, и именно поэтому их называют индуктивным сдвигом алгоритма.

Итак, алгоритм машинного обучения использует для выбора лучшей функции два источника информации: набор данных и предположения (индуктивный сдвиг), вследствие которых он склонен отдавать предпочтение одним функциям перед другими, независимо от закономерностей, обнаруженных в наборе. Индуктивный сдвиг наделяет алгоритм машинного обучения собственным «взглядом» на предоставленные ему данные. Но, как и в реальном мире, не существует таких взглядов (сдвигов), которые были бы правильными во всех ситуациях (наборах данных). Вот почему алгоритмов машинного обучения так много: каждый из них кодирует свой собственный индуктивный сдвиг. Их допущения могут различаться по силе. Чем они сильнее, тем меньше свободы получает алгоритм при выборе функции, которая лучше всего объясняет выявленные закономерности. Набор данных и индуктивный сдвиг в каком-то смысле уравновешивают друг друга: алгоритмы с большим индуктивным сдвигом обращают меньше внимания на предоставленную информацию. Например, если алгоритм отдает предпочтение очень простым функциям, не имеет значения, насколько сложны закономерности в наборе данных, поскольку его индуктивный сдвиг очень большой.

В главе 2 мы объясним, как уравнение прямой можно использовать в качестве шаблона для определения функции. Это очень простое математическое уравнение. Алгоритмы машинного обучения, которые используют его в качестве шаблона для функций, соответствующих определенному набору данных, делают предположение о том, что генерируемая ими модель должна кодировать простое линейное отношение между вводом и выводом. Это предположение является примером индуктивного сдвига. На самом деле этот сдвиг довольно большой, так как несмотря на то, насколько сложные (или нелинейные) закономерности найдены в наборе данных, алгоритм попытается втиснуть в них линейную модель.

Алгоритмы с большим сдвигом могут столкнуться с одной из двух проблем. С одной стороны, если сдвиг слишком сильный, алгоритм проигнорирует важную информацию, и полученная функция не будет учитывать нюансы реальных закономерностей набора данных. Иными словами, функция окажется слишком сложной для этой предметной области, и вывод, который она генерирует, не будет точным. Такой исход называют недообучением. С другой стороны, если сдвиг слишком маленький (или щадящий), алгоритм получает излишнюю свободу в поиске функции, которая должна максимально соответствовать данным. В этом случае функция, скорее всего, получится слишком сложной для