Содержание

Предисловие 8

Об авторах 11

Глава 1. Глубокое обучение и науки о жизни 13
Почему все говорят о глубоком обучении? 13
Современные науки о жизни – это науки о данных 14
О чем эта книга? 15

Глава 2. Введение в глубокое обучение 19
Линейные модели 20
Многослойные персептроны 21
Обучение модели 24
Проверка модели 26
Регуляризация 27
Оптимизация гиперпараметров 28
Другие типы моделей 29
Сверточные нейронные сети 30
Рекуррентные нейронные сети 31
Дополнительное чтение 32

Глава 3. Машинное обучение с DeepChem 33
Наборы данных DeepChem 34
Обучение модели для предсказания токсичности молекул 35
Пример: обучение модели MNIST 42
Набор данных распознавания цифр MNIST 42
Сверточная архитектура для набора MNIST 43
Заключение 47

Глава 4. Машинное обучение и молекулы 48
Что такое молекула? 48
Что такое внутримолекулярные связи? 50
Ковалентные связи 51
Нековалентные связи 51
Молекулярные графы 52
Конформации молекулы 53
Хиральность молекул 54
Фичеризация молекулы 55
Строки SMILES и пакет RDKit 55
Расширенные отпечатки связей 56
Молекулярные дескрипторы 57
Графовые свертки 57

Обучение модели для прогнозирования растворимости ... 58
MoleculeNet ... 60
Строки SMARTS ... 60
Заключение ... 63

Глава 5. Глубокое обучение и биофизика ... 64

Белковые структуры ... 65
Белковые последовательности ... 67
Общие принципы связывания с белками ... 70
Биофизическая фичеризация ... 71
Координатная фичеризация ... 71
Атомная фичеризация ... 76
Пример использования PDBBind ... 76
PDBBind Dataset ... 76
Представление набора данных PDBBind ... 79
Заключение ... 82

Глава 6. Глубокое обучение и геномика ... 85

ДНК, РНК и белки ... 85
Реальное положение дел ... 87
Сайты связывания и факторы транскрипции ... 90
Сверточная модель связывания TF ... 90
Доступность хроматина ... 93
РНК-интерференция ... 95
Заключение ... 98

Глава 7. Машинное обучение и микроскопия ... 99

Краткое введение в микроскопию ... 101
Современная оптическая микроскопия ... 102
Дифракционный предел ... 104
Электронная и атомно-силовая микроскопия ... 105
Микроскопия сверхвысокого разрешения ... 107
Глубокое обучение и дифракционный предел ... 109
Подготовка биологических препаратов для микроскопии ... 109
Окрашивание ... 109
Фиксация препаратов ... 110
Секционирование препаратов ... 111
Флуоресцентная микроскопия ... 111
Артефакты пробоподготовки ... 113
Применение глубокого обучения в микроскопии ... 114
Подсчет клеток ... 114
Клеточная сегментация ... 117
Вычислительные анализы ... 121
Заключение ... 121

Глава 8. Глубокое обучение в медицине ... 123

Компьютерная диагностика ... 123
Вероятностные диагнозы с байесовскими сетями ... 124

Данные электронных медицинских карт ... 126
В чем опасность больших баз данных ЭМК пациентов? ... 128
Глубокая радиология ... 129
Рентгенография и компьютерная томография ... 131
Гистология ... 133
Магниторезонансная томография ... 133
Модель глубокого обучения в качестве лечебного средства ... 134
Диабетическая ретинопатия ... 135
Перспективы глубокого обучения в медицине ... 139
Этические соображения ... 139
Потеря работы ... 140
Заключение ... 140

Глава 9. Генеративные модели ... 141

Вариационные автоэнкодеры ... 141
Генеративные состязательные сети ... 143
Применение генеративных моделей в науках о жизни ... 144
Генерация новых идей для соединений-прототипов ... 144
Разработка белков ... 145
Инструменты для научного поиска ... 145
Будущее генеративного моделирования ... 146
Работа с генеративными моделями ... 146
Анализ вывода генеративной модели ... 148
Заключение ... 151

Глава 10. Интерпретация глубоких моделей ... 154

Как объяснить предсказания? ... 154
Оптимизация входов ... 158
Прогнозирование неопределенности ... 161
Интерпретируемость, объяснимость и последствия для реального мира ... 165
Заключение ... 165

Глава 11. Практический пример виртуального скрининга ... 166

Подготовка набора данных для прогнозного моделирования ... 167
Обучение прогностической модели ... 172
Подготовка набора данных для прогнозирования ... 177
Применение прогностической модели ... 180
Заключение ... 186

Глава 12. Ожидания и перспективы ... 188

Медицинская диагностика ... 188
Персонализированная медицина ... 190
Фармацевтические исследования ... 191
Биологические исследования ... 193
Заключение ... 194

Колофон ... 195

Предметный указатель ... 196

Предисловие

Настало время, когда науки о жизни и данных соединились. Достижения в области робототехники и автоматики позволяют химикам и биологам получать огромное количество данных. Современный ученый за один день может сгенерировать больше данных, чем его предшественники два десятка лет назад могли бы собрать за всю карьеру. Эта способность быстро генерировать данные создала ряд новых научных проблем. Осталась позади эпоха, когда мы обрабатывали данные, загружая их в электронную таблицу и создавая пару графиков. Чтобы извлечь научные знания из новых огромных наборов данных, мы должны уметь выявлять и использовать неочевидные связи.

Одним из мощных инструментов для выявления закономерностей и взаимосвязей в данных является *глубокое обучение*, класс алгоритмов, которые произвели революцию в ряде областей, включая анализ изображений, языковой перевод и распознавание речи. Алгоритмы глубокого обучения превосходно зарекомендовали себя при выявлении и использовании шаблонов в больших наборах данных. По этим причинам глубокое обучение широко применяется во всех дисциплинах науки о жизни. В этой книге представлен обзор применения глубокого обучения в ряде областей, включая генетику, поиск лекарств и медицинскую диагностику. Обзор сопровождается примерами кода, которые помогают применить новые знания на практике и дают читателю отправную точку для будущих исследований и разработок.

Условные обозначения и соглашения, принятые в книге

В книге используются следующие типографские соглашения.

Курсив – используется для смыслового выделения важных положений, новых терминов, имен команд и утилит, а также имен и расширений файлов и каталогов.

`Моноширинный шрифт` – используется для листингов программ, а также в обычном тексте для обозначения имен переменных, функций, типов, объектов, баз данных, переменных среды, операторов, ключевых слов и других программных конструкций и элементов исходного кода.

`Моноширинный полужирный шрифт` – используется для обозначения команд или фрагментов текста, которые пользователь должен ввести дословно без изменений.

`Моноширинный курсив` – используется для обозначения в исходном коде или в командах шаблонных меток-заполнителей, которые должны быть заменены соответствующими контексту реальными значениями.

Такая пиктограмма обозначает совет или рекомендацию.

Такая пиктограмма обозначает указание или примечание общего характера.

Эта пиктограмма обозначает предупреждение или особое внимание к потенциально опасным объектам.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв прямо на нашем сайте **www.dmkpress.com**, зайдя на страницу книги, и оставить комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу **dmkpress@gmail.com**, при этом напишите название книги в теме письма.

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу **http://dmkpress.com/authors/publish_book/** или напишите в издательство по адресу **dmkpress@gmail.com**.

Скачивание исходного кода примеров

Скачать файлы с дополнительной информацией для книг издательства «ДМК Пресс» можно на сайте **www.dmkpress.com** или **www.дмк.рф** на странице с описанием соответствующей книги.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг — возможно, ошибку в тексте или в коде, — мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от расстройств и поможете нам улучшить последующие версии этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу **dmkpress@gmail.com**, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и O'Reilly очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или веб-сайта, чтобы мы могли применить санкции.

Пожалуйста, свяжитесь с нами по адресу электронной почты **dmkpress@gmail.com** со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую нам предоставлять вам качественные материалы.

Благодарности

Авторы хотели бы поблагодарить Николь Таш (Nicole Tache), нашего редактора в O'Reilly, а также технических рецензентов и бета-рецензентов за их ценный

вклад в книгу. Кроме того, мы хотели бы поблагодарить Карла Лесвинга (Karl Leswing) и Чжэньциня (Майкла) By (Zhenqin/Michael Wu) за их вклад в код, а также Джонни Израэли (Johnny Israeli) за ценные советы для главы по геномике.

Бхарат благодарит свою семью за поддержку и ободрение в течение многих долгих выходных и ночей, проведенных в работе над этой книгой.

Питер хотел бы поблагодарить свою жену за ее постоянную поддержку, а также многих коллег, от которых он так много узнал о машинном обучении.

Патрик благодарен своей жене Андреа и дочерям Эли и Мэдди за их любовь и поддержку. Он также благодарит прошлых и настоящих коллег из Vertex Pharmaceuticals и Relay Therapeutics, у которых он многому научился.

Наконец, мы хотим поблагодарить сообщество разработчиков программного обеспечения DeepChem за поддержку и консультации на протяжении работы над книгой.

Об авторах

Бхарат Рамсундар (Bharath Ramsundar) является соучредителем и техническим директором компании Datamined, занятой созданием больших наборов биологических данных. Эти наборы данных широко востребованы в связи с бумом ИИ в биотехнологиях. Бхарат также является ведущим разработчиком и создателем DeepChem.io, пакета с открытым исходным кодом, основанного на TensorFlow и нацеленного на повышение доступности глубокого обучения в области поиска лекарств, и соавтором пакета тестов MoleculeNet.

Бхарат получил степени бакалавра в Калифорнийском университете в Беркли по специальностям «Электроника и информатика» и «Математика» и был удостоен чести произнести прощальную речь перед выпускниками математического факультета. Недавно он защитил кандидатскую диссертацию по информатике в Стэнфордском университете (все, кроме формальной части) в группе Виджая Панде и при поддержке Hertz Fellowship, стипендиального фонда с самым строгим отбором аспирантов.

Питер Истман (Peter Eastman) разрабатывает программное обеспечение для биологов и химиков в отделе биотехнологий Стэнфордского университета. Он является ведущим автором OpenMM, инструментария для высокопроизводительного моделирования молекулярной динамики, и основным разработчиком DeepChem, пакета для глубокого машинного обучения в области химии, биологии и материаловедения. С 2000 года он является профессиональным инженером-программистом, в том числе вице-президентом по разработке программного обеспечения для Silicon Genetics – компании, занимающейся разработкой программного обеспечения для биоинформатики. Сейчас исследовательские интересы Питера сосредоточены на пересечении физики и глубокого обучения.

Пэт Уолтерс (Pat Walters) возглавляет группу по вычислительной информатике в Relay Therapeutics в Кембридже, штат Массачусетс. Его группа работает над новыми вычислительными методами, объединяющими компьютерное моделирование и экспериментальные данные в программах по разработке новых лекарственных препаратов. До прихода в Relay Therapeutics он более 20 лет проработал в Vertex Pharmaceuticals, где занимал должность генерального директора по моделированию и информатике.

Пэт является членом редакционно-консультативного совета «Журнала медицинской химии» и ранее занимал аналогичные должности в журналах «Молекулярная информатика» и «Бюллетень исследований и разработки лекарственных средств». Он продолжает играть активную роль в научном сообществе. Пэт был председателем конференции Гордона по компьютерной разработке лекарств 2017 года и сыграл важную роль в ряде профессиональных сообществ, включая базу данных разработчиков лекарственных средств (Drug Design Data Resource, D3R) и Ассоциацию американских химиков-разработчиков (American Chemical Society TDT initiative). Пэт получил докторскую степень по органической химии в университете Аризоны, где изучал применение искусственного интеллекта

в анализе молекулярных конформаций. До получения степени доктора он работал в Varian Instruments как химик и разработчик программного обеспечения. Пэт Уолтерс получил степень бакалавра по химии в Калифорнийском университете в Санта-Барбаре.

Виджай Панде (Vijay Pande), доктор философии, является генеральным партнером Andreessen Horowitz, где отвечает за инвестиции фирмы в области биологии и информатики, включая применение вычислений, машинного обучения и искусственного интеллекта в биологии и здравоохранении, а также инновационных научных технологий. Он работает адъюнкт-профессором факультета биотехнологий в Стэнфорде, где изучает применение компьютерных методов в медицине и биологии и консультирует студентов и аспирантов, что привело к появлению более двухсот публикаций, двух патентов и двух новых лекарственных препаратов.

Будучи предпринимателем, Виджай является основателем *Проекта распределенных вычислений Folding@Home по изучению заболеваний*, расширяющего границы применения компьютерных технологий, таких как распределенные системы, машинное обучение и экзотические компьютерные архитектуры, в биологии и медицине. Он занимается как фундаментальными исследованиями, так и разработкой новых методов лечения. Виджай также стал соучредителем Globavir Biosciences, где превратил свои научные достижения в Стэнфорде и Folding@Home в успешный стартап, открывая лекарства от лихорадки денге и лихорадки Эбола. В подростковом возрасте он был первым сотрудником стартапа видеоигр Naughty Dog Software, производителя популярной игровой франшизы Crash Bandicoot.

Глава 1

Глубокое обучение и науки о жизни

Существует много направлений, где могут проявить себя энтузиасты и эксперты по работе с данными, однако лишь немногие области могут сравниться с биомедицинскими исследованиями по фундаментальным последствиям применения больших данных. Появление современной медицины коренным образом изменило природу человеческого существования. За последние 20 лет мы увидели инновации, которые изменили жизнь множества людей. Впервые обнаруженный в 1981 году, ВИЧ/СПИД был смертельным заболеванием. Появление антиретровирусной терапии значительно увеличило продолжительность жизни больных ВИЧ в развитых странах. Другие болезни, такие как гепатит С, который десять лет назад считался в основном неизлечимым, теперь можно вылечить. Достижения в области генетики позволяют обнаруживать, и, надеемся, в скором времени, лечить, широкий спектр заболеваний. Инновации в диагностике и измерительной аппаратуре позволили врачам целенаправленно выявлять и контролировать заболевания в организме человека. Многие из этих прорывов случились и продолжают развиваться благодаря новым вычислительным методам.

Почему все говорят о глубоком обучении?

Алгоритмы машинного обучения теперь являются ключевым компонентом всех современных компьютерных технологий, от покупок в интернете до социальных сетей. Команды ученых-компьютерщиков разрабатывают алгоритмы, позволяющие цифровым помощникам, таким как Amazon Echo или Google Home, понимать речь. Достижения в области машинного обучения позволяют на лету выполнять перевод веб-страниц. Помимо влияния на повседневную жизнь, машинное обучение активно воздействует на многие области естественных наук и наук о жизни. Алгоритмы машинного обучения применяются ко всему, начиная с поиска новых галактик и заканчивая классификацией субатомных взаимодействий на Большом адронном коллайдере.

Одним из источников этих технологических достижений стало появление класса методов машинного обучения, известных как *глубокие нейронные сети*. Хотя технологические основы искусственных нейронных сетей были разработаны в 1950-х годах и усовершенствованы в 1980-х годах, истинная мощь этого метода не была полностью реализована до тех пор, пока за последнее десятилетие не

появились достаточно мощные компьютеры. Мы предоставим более полный обзор глубоких нейронных сетей в следующей главе, но хотим отдельно отметить некоторые из достижений, которые стали возможными благодаря применению глубокого обучения:

- *распознавание речи* в сотовых телефонах, компьютерах, телевизорах и других устройствах, подключенных к интернету, основано на глубоком обучении;
- *распознавание изображений* является ключевым компонентом беспилотных автомобилей, поиска в интернете и других приложений. Многие достижения в области глубокого обучения, которые привели к созданию потребительских приложений, в настоящее время используются в биомедицинских исследованиях. В одном из примеров исследователи используют глубокое обучение для классификации опухолевых клеток по типам и определения восприимчивости к лечению различными препаратами;
- *рекомендательные системы* стали ключевым компонентом сетевого взаимодействия. Такие компании, как Amazon, используют глубокое обучение, чтобы стимулировать дополнительные покупки при помощи подсказок «с этим товаром обычно покупают». Netflix использует аналогичный подход, чтобы рекомендовать фильмы, которые человек может захотеть посмотреть. Многие идеи, заложенные в основу этих рекомендательных систем, применяются для поиска молекул, которые могут стать отправной точкой в разработке новых лекарств;
- *языковой перевод* раньше выполняли очень сложные системы, основанные на правилах. За последние несколько лет системы перевода, основанные на глубоком обучении, далеко превзошли системы, основанные на ручном вводе правил. Аналогичные методы в настоящее время применяются для извлечения понятий из научной литературы и уведомления ученых о журнальных статьях, которые они могли пропустить.

Это лишь некоторые из инноваций, появившихся благодаря применению методов глубокого обучения. Мы живем в интересное время, когда происходит слияние массивов общедоступных научных данных и методов обработки этих данных. Те, кто способен комбинировать данные с новыми методами обучения по шаблонам в этих данных, могут добиться значительных научных достижений.

Современные науки о жизни – это науки о данных

Как упоминалось выше, фундаментальная природа наук о жизни изменилась. Доступность робототехники и микроэкспериментов привела к значительному увеличению объема извлекаемых экспериментальных данных. В 1980-х биолог проводил один эксперимент и получал один результат. Эти данные обычно можно обработать вручную с помощью карманного калькулятора. Если говорить о современной биологии, то у нас есть инструментарий, способный за день или два сгенерировать миллионы точек экспериментальных данных. Эксперименты, генерирующие огромные наборы данных, например секвенирование генов, стали недорогими и рутинными.

Успехи в секвенировании генов привели к созданию баз данных, которые связывают генетический код человека со множеством потенциальных заболеваний,

включая диабет, рак и генетические заболевания, такие как муковисцидоз. Используя вычислительные методы для извлечения и анализа данных, ученые совершенствуют понимание причин генетически обусловленных заболеваний и используют это понимание для разработки новых методов лечения.

Дисциплины, которые когда-то полагались главным образом на наблюдение за людьми, теперь используют наборы данных, просто не поддающиеся ручному анализу. Машинное обучение в настоящее время регулярно используется для классификации изображений клеток. Выходные данные этих моделей машинного обучения используются для обнаружения и классификации раковых опухолей и для оценки эффективности возможного лечения заболеваний.

Успехи в экспериментальных методах привели к созданию нескольких баз данных, которые каталогизируют структуры химических веществ и влияние этих веществ на широкий спектр биологических процессов или видов деятельности. Эти *структурно–функциональные взаимосвязи* (SAR, structure-activity relationship) составляют основу области, известной как *химическая информатика*, или *хемоинформатика*. В настоящее время ученые обрабатывают эти обширные наборы данных и используют их для построения *прогностических моделей*, на которых основаны новые методы разработки лекарств.

Новое поколение данных нуждается в новом поколении ученых, которые чувствуют себя комфортно как в лаборатории, так и за компьютером. Обладатели гибридных навыков способны увидеть закономерности и тенденции в больших наборах данных и сделать научные открытия завтрашнего дня.

О чем эта книга?

В первых нескольких главах мы даем обзор глубокого обучения и того, как его можно применять в науках о жизни. Мы начнем с *машинного обучения*, которое было определено как «наука и искусство программирования компьютеров, извлекающих знания из данных»[1].

Глава 2 дает краткое введение в разновидность машинного обучения, известную как *глубокое обучение*. Мы начнем с примера того, как глубокое обучение можно использовать для решения такой простой задачи, как линейная регрессия, и перейдем к более сложным моделям, которые обычно используются для решения реальных проблем в науках о жизни. Машинное обучение обычно сопровождается разделением начального набора данных на *обучающий набор* (training set), который используется для обучения модели, и *проверочный набор* (test set), который используется для проверки точности модели.

В главе 2 мы обсуждаем некоторые детали, связанные с обучением и проверкой прогностических моделей. После того как модель обучена, ее производительность обычно можно оптимизировать, изменяя ряд характеристик, известных как *гиперпараметры* (hyperparameters). В этой главе представлен обзор процесса оптимизации. Глубокое обучение – это не одиночный подход, а набор связанных методов. Глава 2 заканчивается знакомством с несколькими наиболее важными подходами к глубокому обучению.

---

[1] *Furbush J.* Machine Learning: A Quick and Simple Definition // **https://www.oreilly.com/ideas/machine-learning-a-quick-and-simple-definition**.

В главе 3 мы представляем *DeepChem*, библиотеку Python с открытым исходным кодом, которая была специально разработана для упрощения создания моделей глубокого обучения в области наук о жизни. Вслед за обзором DeepChem мы представляем наш первый пример программирования, демонстрирующий применение этой библиотеки. Мы создаем модель, предсказывающую токсичность молекул. Во втором примере программирования мы показываем, как DeepChem можно использовать для классификации изображений, что является обычной задачей в современной биологии. Как кратко упомянуто выше, глубокое обучение используется в различных приложениях *визуальной диагностики*, начиная от диагностики рака до выявления глаукомы. Из обсуждения конкретных приложений затем вытекает объяснение некоторых внутренних методов глубокого обучения.

В главе 4 представлен обзор того, как машинное обучение применяется к молекулам. Мы начинаем со знакомства с понятием молекул, составных частей окружающего нас мира. Хотя молекулы в какой-то мере можно считать аналогом строительных кирпичей, они являются гибкими и демонстрируют динамическое поведение. Чтобы охарактеризовать молекулы с помощью вычислительных методов, таких как глубокое обучение, нам нужно найти способ представления молекул в компьютере. Компьютерная кодировка молекул похожа на представление изображения в виде набора пикселей. Во второй половине главы 4 мы описываем ряд способов представления молекул и их применение для построения моделей глубокого обучения.

Глава 5 содержит введение в *биофизику* – науку, применяющую законы физики к биологическим явлениям. Мы начнем с обсуждения белков, своеобразных молекулярных машин, благодаря которым существует белковая форма жизни. Прогнозирование воздействия лекарств на организм невозможно без понимания эффектов их взаимодействия с белками. Чтобы понять эти эффекты, мы начнем с обзора того, как устроены белки и как различаются их структуры. Белки – это объекты, чья трехмерная структура определяет их биологическую функцию. Для того чтобы модель машинного обучения могла предсказать влияние молекулы лекарства на функцию белка, мы должны представить трехмерную структуру белка в форме, понятной для программы машинного обучения. Во второй половине главы 5 мы рассматриваем несколько способов представления структур белка. Опираясь на эти знания, мы представим еще один пример кода, где глубокое обучение применяется для предсказания эффекта взаимодействия молекулы лекарства с белком.

Генетика стала ключевым компонентом современной медицины. Генетическое секвенирование опухолей позволило персонализировать лечение рака и может произвести революцию в медицине. Секвенирование генов, которое раньше было сложным процессом, требующим огромных инвестиций, теперь стало обычным делом и может проводиться где угодно. Мы уже достигли этапа, когда владельцы собак могут провести недорогие генетические тесты для определения происхождения своего питомца. В главе 6 мы даем обзор генетики и геномики, начиная со знакомства с молекулами ДНК и РНК, важнейшими шаблонами, для производства белков. Недавние открытия показали, что взаимодействия ДНК и РНК протекают намного сложнее, чем считалось изначально. Во второй половине главы 6 мы представляем несколько примеров кода, предсказывающего влияние различных факторов на взаимодействие ДНК и РНК.

Ранее в этой главе мы упоминали о выдающихся успехах, которые были достигнуты благодаря применению глубокого обучения для анализа биологических и медицинских изображений. Многие из изучаемых в этих экспериментах явлений слишком малы, чтобы их можно было наблюдать человеческим глазом. Изображения для последующего анализа методами глубокого обучения получают при помощи микроскопа. В главе 7 представлен обзор микроскопии в ее многочисленных формах, от простого светового микроскопа, который мы все использовали в школе, до сложных приборов, способных получать изображения одиночных атомов. Эта глава также охватывает некоторые ограничения современных подходов и знакомит с экспериментальными источниками изображений, на которых строятся модели глубокого обучения.

Одной из областей, в которой раскрываются огромные перспективы, является применение глубокого обучения для медицинской диагностики. Медицина невероятно сложна, и ни один врач не способен вобрать в себя все имеющиеся медицинские знания. В идеале система машинного обучения могла бы изучить всю медицинскую литературу в мире и помогать врачам ставить диагнозы. Хотя мы еще далеки от этого уровня, но уже есть успехи. Глава 8 начинается с истории использования методов машинного обучения в медицинской диагностике и показывает переход от ручного кодирования правил к статистическому анализу медицинских данных. Как и во многих обсуждаемых нами темах, ключевой проблемой является представление медицинской информации в компьютерном формате, понятном программам машинного обучения. В этой главе мы представляем введение в электронные медицинские карты и некоторые проблемы, связанные с этими записями. Во многих случаях медицинские изображения бывают очень сложными, и анализ и интерпретация этих изображений вызывают затруднения даже у квалифицированных специалистов. В этих случаях глубокое обучение может улучшить навыки эксперта, классифицируя изображения и выявляя ключевые особенности. Глава 8 завершается рядом примеров того, как глубокое обучение используется для анализа изображений в различных областях медицины.

Как мы упоминали ранее, машинное обучение становится ключевым инструментом для поиска новых лекарств. Ученые используют модели глубокого обучения, чтобы оценить взаимодействия между молекулами лекарств и белками. Эти взаимодействия могут вызывать биологический отклик, оказывающий терапевтическое воздействие на пациента. Модели, которые мы упоминали до сих пор, являются *дискриминантными моделями*.

Исходя из характеристик молекулы, модель генерирует прогноз некоторого свойства вещества. Описание молекулы может быть получено из обширной базы данных существующих молекул или родиться в воображении ученого. Но что, если мы не будем полагаться на существующие описания или собственную фантазию и разработаем компьютерную программу, способную «изобретать» новые молекулы? Глава 9 представляет *генеративную модель*, которая сначала обучается на наборе существующих молекул, а затем применяется для генерации новых молекул. Модель глубокого обучения, которая генерирует эти молекулы, также может зависеть от других моделей, предсказывающих активность новых молекул.

До сих пор мы обсуждали модели глубокого обучения как «черные ящики». Мы запускаем модель с набором входных данных, и она генерирует прогноз без объяснения того, как или почему прогноз был сгенерирован. Во многих ситуациях это

неприемлемо. Если у нас есть модель глубокого обучения для медицинской диагностики, нам нужно понимать причины диагноза. Объяснение причин диагноза даст врачу больше уверенности в прогнозе, а также может повлиять на решение о лечении. У моделей глубокого обучения есть исторический недостаток – прогнозы даже самых надежных моделей трудно поддаются логическому обоснованию. В настоящее время разрабатывается ряд методов, позволяющих пользователям лучше понять факторы, влияющие на прогноз. В главе 10 представлен обзор некоторых методов, способствующих пониманию причин прогноза. Другим важным аспектом *предсказательного моделирования* является *точность предсказаний* модели. Знание точности модели помогает решить, насколько можно полагаться на эту модель. Учитывая, что машинное обучение может использоваться для постановки жизненно важных диагнозов, оценка точности модели имеет решающее значение. В последнем разделе главы 10 представлен обзор некоторых методов оценки точности предсказания модели.

В главе 11 мы представляем практический пример использования DeepChem. В этом примере мы используем метод, известный как *виртуальный скрининг*, чтобы определить потенциальные отправные точки для поиска новых лекарств. Создание новых лекарств – это сложный процесс, который часто начинается с техники, известной как скрининг. Он применяется для поиска молекул, которые могут стать основой для новых лекарств. Скрининг может проводиться экспериментально, когда миллионы молекул тестируются в миниатюрных биологических тестах (так называемых *пробах*), или на компьютере с использованием виртуального скрининга. В методе виртуального скрининга модель обучается на наборе известных лекарств или других биологически активных молекул. Затем эта модель используется для прогнозирования активности большого количества новых молекул. Благодаря высокому быстродействию машинного обучения за несколько дней вычислений удается обработать сотни миллионов молекул.

Книга заканчивается главой 12 «Ожидания и перспективы». В этой главе рассматривается сегодняшнее положение дел и перспективы применения глубокого обучения в науках о жизни. Обсуждается ряд текущих проблем, включая доступность и качество наборов данных. Мы также выделяем перспективы и подводные камни в ряде других областей, включая диагностику, персонализированную медицину, фармацевтику и исследования в области биологии.

Глава 2

Введение в глубокое обучение

В этой главе представлены основные принципы глубокого обучения. Если вы хорошо знакомы с глубоким обучением, то можете пропустить эту главу и перейти к следующей. Новичкам в области глубокого обучения следует внимательно изучить главу, поскольку материал, который она охватывает, будет важен для понимания остальной части книги.

Решение большинства проблем, которые мы обсудим, сводится к созданию математической функции общего вида

$$\mathbf{y} = f(\mathbf{x}).$$

Обратите внимание, что **x** и **y** выделены жирным шрифтом. Это означает, что они являются *векторами*. Функция может принимать тысячи или даже миллионы чисел в качестве входных данных, и она может выдавать столь же много чисел в качестве выходных данных. Вот несколько примеров функций, которые вы могли бы создать:

- **x** содержит цвета всех пикселей изображения. $f(\mathbf{x})$ равно 1, если изображение содержит кошку, и 0, если это не так;
- то же, что и выше, однако $f(\mathbf{x})$ должно быть *числовым вектором*. Первый элемент указывает, содержит ли изображение кошку, второй – содержит ли оно собаку, третий – содержит ли оно самолет, и так для тысяч разновидностей объектов;
- **x** содержит последовательность ДНК хромосомы, а **y** должен быть вектором, длина которого равна числу оснований в хромосоме. Каждый элемент должен равняться 1, если это основание является частью области, кодирующей белок, и 0, если это не так;
- **x** описывает структуру молекулы. Мы обсудим различные способы компьютерного представления молекул в последующих главах. **y** должен быть вектором, где каждый элемент описывает некоторое физическое свойство молекулы, например растворимость в воде, силу связи с другой молекулой, и т. д.

Как видите, функция $f(\mathbf{x})$ может быть очень и очень сложной! Обычно она принимает длинный вектор в качестве входных данных и пытается извлечь из него информацию, которая совсем не очевидна, если взглянуть на входные числа.

Традиционный подход к решению этой проблемы заключается в разработке функции вручную. При таком подходе вы бы начали с анализа проблемы. Какие сочетания пикселей с наибольшей вероятностью указывают на присутствие кошки? Какие паттерны ДНК указывают на отличие кодирующих участков от некодирующих? Вы должны написать компьютерный код для выделения определенных признаков, а затем попытаться определить комбинации признаков, которые надежно обеспечивают желаемый результат. Это медленный и трудоемкий процесс, который сильно зависит от опыта человека, выполняющего анализ данных.

Машинное обучение использует совершенно другой подход. Вместо того чтобы разрабатывать функцию вручную, вы позволяете компьютеру *обучить* свою собственную функцию на основе данных. Вы берете тысячи или миллионы изображений и помечаете те из них, на которых есть кошка. Затем вы предоставляете эти *обучающие данные* компьютеру и позволяете ему искать функцию, которая стремится к 1 для изображений с кошками и к 0 для изображений, на которых кошки нет.

Что значит «позволить компьютеру искать функцию»? В общем вы создаете *модель*, которая определяет некоторый большой класс функций. Модель включает в себя *параметры* – переменные, которые могут принимать любое значение. Выбирая значения параметров, вы тем самым выбираете конкретную функцию из всего множества функций в классе. Задача компьютера заключается в подборе значений параметров. Компьютер пытается найти такие значения, чтобы при подаче на вход модели обучающих данных ее выходные данные были как можно ближе к соответствующим целевым значениям.

Линейные модели

Одна из самых простых моделей, которые вы можете рассмотреть, – это *линейная модель*:

$$\mathbf{y} = M\mathbf{x} + \mathbf{b}.$$

В этом уравнении $\mathbf{M}$ представляет собой матрицу (иногда называемую «весами»), а $\mathbf{b}$ – это вектор (называемый «отклонениями»). Их размеры определяются количеством входных и выходных значений. Если $\mathbf{x}$ имеет длину T и вы хотите, чтобы $\mathbf{y}$ имел длину S, то $\mathbf{M}$ будет матрицей $S \times T$, а $\mathbf{b}$ будет вектором длины S. Вместе они составляют параметры модели. Это уравнение просто говорит о том, что каждый выходной компонент является линейной комбинацией входных компонентов. Задавая параметры $\mathbf{M}$ и $\mathbf{b}$, вы можете выбрать для каждого компонента любую линейную комбинацию, которую пожелаете.

Это была одна из самых ранних моделей машинного обучения. Она была предложена еще в 1957 году под названием «*персептрон*»[1] (perceptron). Название представляет собой яркий пример научного маркетинга. В нем звучит научная фантастика, и оно обещает удивительные вещи, хотя на самом деле это не что иное, как линейное преобразование. Во всяком случае, названию удалось продержаться более полувека.

[1] На русском языке иногда пишут «перцептрон». – *Прим. перев.*

Линейная модель очень легко записывается в общем виде. Она имеет одну и ту же форму независимо от того, к какой проблеме она относится. Единственные различия между линейными моделями – это длина входного и выходного векторов. В данном случае обучение – это лишь вопрос выбора значений параметров, что можно сделать простым способом с помощью общих алгоритмов. С точки зрения машинного обучения это идеальный подход, при котором модель и алгоритмы не зависят от решаемой проблемы. Для автоматического определения параметров достаточно лишь предоставить обучающие данные, и модель будет преобразована в функцию, которая решает вашу проблему.

К сожалению, линейные модели также очень ограничены. Как показано на рис. 2.1, линейная модель (уравнение первой степени, описывающее прямую линию) просто не может соответствовать большинству реальных наборов данных. Ситуация становится еще хуже, когда вы переходите к очень многомерным данным. Никакая линейная комбинация пикселей в изображении не будет надежно определять, содержит ли изображение кошку. Задача распознавания изображений требует гораздо более сложной нелинейной модели. Фактически любая модель, которая решает проблему поиска объектов, обязательно будет очень сложной и очень нелинейной. Но как мы можем определить нелинейную модель в общем виде? Пространство всех возможных нелинейных функций бесконечно велико. Каким образом определить модель так, чтобы, просто выбирая значения параметров, мы могли создать практически любую нелинейную функцию, которая нам когда-либо понадобится?

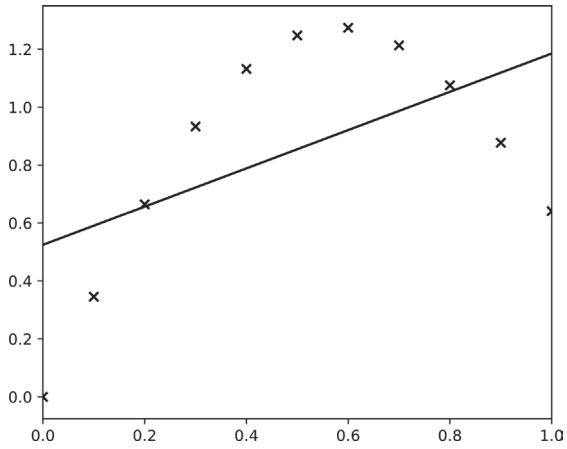

Рис. 2.1 ❖ Линейная модель не может представлять точки данных, лежащие на кривой. Для этого нужна нелинейная модель

Многослойные персептроны

Простой подход заключается в последовательном объединении нескольких линейных преобразований. Например, мы можем записать двойное преобразование в следующем виде:

$$\mathbf{y} = \mathbf{M}_2\varphi(\mathbf{M}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2.$$

Постарайтесь вникнуть в последовательность действий. Мы начинаем с обычного линейного преобразования $\mathbf{M}_1\mathbf{x} + \mathbf{b}_1$. Затем пропускаем результат преобразования через нелинейную функцию $\varphi(x)$ и к полученному выходу применяем второе линейное преобразование. Функция $\varphi(x)$, известная как *функция активации*, является важной частью модели. Без нее модель все равно осталась бы линейной и не более мощной, чем предыдущая. Ведь линейная комбинация линейных комбинаций все равно остается не более чем линейной комбинацией исходных входов! Внося нелинейность, мы даем возможность модели исследовать гораздо более широкий диапазон функций.

Мы можем не останавливаться на двух линейных преобразованиях и выстроить их в последовательность произвольной длины:

$$\mathbf{h}_1 = \varphi_1(\mathbf{M}_1\mathbf{x} + \mathbf{b}_1)$$
$$\mathbf{h}_2 = \varphi_2(\mathbf{M}_2\mathbf{h}_1 + \mathbf{b}_2)$$
$$\ldots$$
$$\mathbf{h}_{n-1} = \varphi_{n-1}(\mathbf{M}_{n-1}\mathbf{h}_{n-2} + \mathbf{b}_{n-1})$$
$$\mathbf{y} = \varphi_n(\mathbf{M}_n\mathbf{h}_{n-1} + \mathbf{b}_n).$$

Такая модель называется *многослойным персептроном* (multilayer perceptron, MLP). Промежуточные шаги h_i называются *скрытыми слоями*. Их название отражает тот факт, что они не являются ни входными данными, ни выходными данными, а являются лишь промежуточными значениями, используемыми в процессе вычисления результата. Также обратите внимание, что мы добавили индекс к каждой функции $\varphi(x)$, указывая на то, что разные слои могут использовать разные нелинейности.

Эту последовательность вычислений можно визуально представить в виде *стека слоев*, как показано на рис. 2.2. Каждый слой соответствует линейному преобразованию с последующей нелинейностью. Информация перемещается с одного уровня на другой, а выход одного уровня становится входом для следующего. Каждый слой имеет свой собственный набор параметров, которые определяют, как его выход рассчитывается на основе его входных данных.

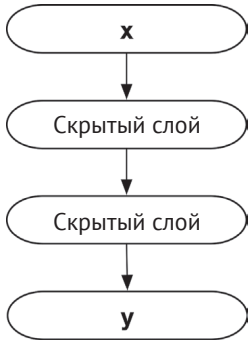

Рис. 2.2 ❖ Многослойный персептрон представлен как стек слоев с информацией, перетекающей из одного слоя в другой

Многослойные персептроны и их вариации также называют *нейронными сетями*. Название отражает сходство между машинным обучением и нейробиологией. Биологический нейрон соединяется со многими другими нейронами. Он получает от них сигналы, складывает их вместе, а затем отправляет свои собственные сигналы другим нейронам в зависимости от результата. В очень грубом приближении можно считать, что MLP работают так же, как нейроны в вашем мозгу.

Какой должна быть функция активации $\varphi(x)$? Неожиданный ответ заключается в том, что это в целом не имеет значения! Конечно, это не совсем так, но значение не столь велико, как вы могли ожидать. Подойдет почти любая монотонная и достаточно плавная непрерывная функция. За долгие годы было опробовано множество различных функций, и хотя некоторые из них работают лучше, чем другие, почти все они дают приемлемые результаты.

На сегодняшний день наиболее популярной функцией активации считается *выпрямленная линейная функция* (Rectified Linear Unit, ReLU) $\varphi(x) = \max(0, x)$. Если вы не знаете, какую функцию выбрать, она будет хорошим вариантом по умолчанию. К другим распространенным вариантам относятся *гиперболический тангенс* $\tanh(x)$ и *логистическая сигмоида* $\varphi(x) = 1/(1 + e^{-x})$. Все эти функции показаны на рис. 2.3.

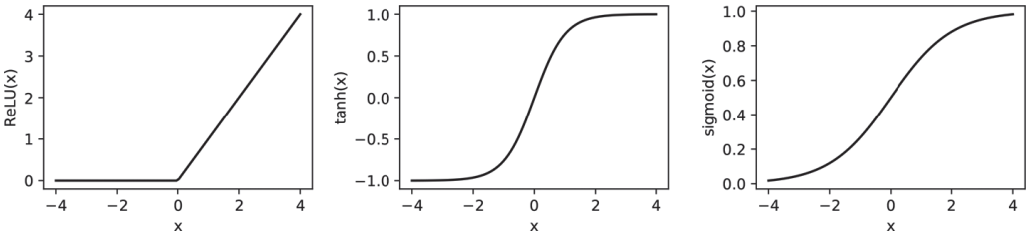

Рис. 2.3 ❖ Три обобщенные функции активации: выпрямленная линейная (ReLU), гиперболический тангенс и логистическая сигмоида

Мы также должны выбрать два других свойства MLP: его *ширину* и *глубину*. Ширина означает размер скрытых слоев. С простой линейной моделью у нас не было выбора. Размеры **M** и **b** были полностью определены исходя из длины **x** и **y**. Другое дело – скрытые слои. Для каждого $\mathbf{h}_i$ мы можем выбрать любую длину, какую пожелаем. В зависимости от задачи $\mathbf{h}_i$ могут быть намного больше или намного меньше, чем входные и выходные векторы.

Глубина относится к числу слоев в модели. Модель только с одним скрытым слоем называется *неглубокой* (shallow model). Модель со многими скрытыми слоями описывается как *глубокая* (deep model). Отсюда, по сути, происходит термин «глубокое обучение», означающий машинное обучение с использованием многослойных моделей.

Выбор количества и ширины слоев в модели – это в равной мере искусство и наука. Или, говоря более формально: «Это все еще активная область исследований». Часто все сводится к тому, чтобы попробовать множество комбинаций и посмотреть, какая из них лучше сработает. Однако есть несколько общих прин-

ципов, которые послужат руководством или, по крайней мере, помогут вам понять результаты задним числом.

1. MLP с одним скрытым слоем является *универсальным аппроксиматором*. Это означает, что он может аппроксимировать вообще любую функцию (в определенных достаточно разумных пределах). В каком-то смысле вам никогда не понадобится более одного скрытого слоя. Теоретически этого уже достаточно, чтобы воспроизвести любую функцию. К сожалению, этот результат сопровождается серьезным побочным эффектом: точность аппроксимации зависит от *ширины* скрытого слоя, и вам может потребоваться очень широкий слой, чтобы получить достаточную точность для текущей задачи. Это рассуждение подводит нас ко второму принципу.
2. Глубокие модели, как правило, требуют меньше параметров, чем неглубокие. Это утверждение намеренно сформулировано несколько расплывчато. Для конкретных случаев можно доказать и более строгие утверждения, но в общем можно сказать так: каждая проблема требует модели с определенной глубиной для эффективного достижения приемлемой точности. И как только глубина уменьшается ниже этого предела, необходимая ширина слоев (и следовательно, общее количество параметров) резко возрастает. Можно предположить, что глубокие модели всегда лучше. К сожалению, это частично противоречит третьему принципу.
3. Глубокие модели, как правило, труднее обучать, чем неглубокие. Примерно до 2007 года большинство моделей машинного обучения оставались неглубокими. Теоретические преимущества глубоких моделей были известны, но исследователям обычно не удавалось их обучать. С тех пор появились улучшенные алгоритмы обучения, новые типы моделей, которые легче обучать, и, конечно, более быстрые компьютеры в сочетании с большими наборами обучающих данных. Эти достижения повысили полезность глубоких моделей и дали начало «глубокому обучению» как самостоятельной отрасли знаний. Тем не менее, несмотря на эти достижения, общий принцип остается прежним: чем глубже модель, тем труднее ее обучать.

Обучение модели

Наверное, у вас возник закономерный вопрос: как же нам обучать модель? MLP предоставляют нам общую модель, которая может использоваться для любой проблемы. Мы обсудим другие, более специализированные типы моделей чуть позже. Теперь мы хотим, чтобы аналогичный общий алгоритм нашел оптимальные значения параметров модели для данной задачи. Как мы это сделаем?

Конечно, первое, что вам нужно, – это *обучающий набор*. Он должен состоять из большого числа пар (**x**, **y**), также известных как *выборки*[1] (samples). Каждый набор определяет входные данные для модели и указывает, что ожидается на выходе модели при заданных входных данных. Например, обучающая выборка может

[1] В публикациях на русском языке иногда встречается взаимная перестановка терминов: обучающий набор называют обучающей выборкой, а пару (**x**, **y**) называют выборкой или прецедентом. Нужно просто уяснить, что независимо от терминов модель учится на большом количестве соответствий «вход-выход». – *Прим. перев.*

представлять собой набор изображений вместе с метками, указывающими, содержит ли данное изображение кошку или нет.

Далее необходимо определить *функцию потерь* $L(\mathbf{y}, \hat{\mathbf{y}})$, где $\mathbf{y}$ – это фактический результат модели, а $\hat{\mathbf{y}}$ – целевое значение, указанное в обучающем наборе. Таким образом вы измеряете, насколько хорошо модель воспроизводит данные обучения. Затем производится усреднение по всем выборкам в обучающем наборе:

$$\text{Средние потери} = \frac{1}{N}\sum_{i=1}^{N} L(\mathbf{y}_i, \hat{\mathbf{y}}_i).$$

Значение $L(\mathbf{y}, \hat{\mathbf{y}})$ должно быть маленьким, когда аргументы функции близки друг к другу, и большим, когда они далеко друг от друга. Другими словами, мы берем каждую выборку в обучающем наборе, используем ее в качестве входных данных для модели и смотрим, насколько выходной сигнал отличается от целевого значения. Затем усредняем разницу по всему обучающему набору.

Для каждой конкретной проблемы должна быть выбрана соответствующая функция потерь. Распространенным вариантом является *евклидово расстояние* (также известное как расстояние L_2):

$$L(\mathbf{y}, \hat{\mathbf{y}}) = \sqrt{\sum_i (y_i + \hat{y}_i)^2},$$

где y_i означает i-й компонент вектора $\mathbf{y}$.

Когда $\mathbf{y}$ представляет распределение вероятности, обычно применяется *перекрестная энтропия* (cross entropy) $L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_i y_i \log \hat{y}_i$. Возможны и другие варианты, поскольку универсального «лучшего» выбора не существует. Это зависит от особенностей вашей проблемы.

Теперь, когда у нас есть способ измерить, насколько хорошо работает модель, нам нужен способ ее улучшить. Мы хотим найти значения параметров, которые минимизируют среднюю потерю по обучающему набору. Есть много способов минимизации средней потери, но в большинстве случаев используют один из вариантов алгоритма *градиентного спуска*. Пусть θ представляет собой набор всех параметров в модели. Градиентный спуск представляет собой ряд небольших шагов

$$\theta \leftarrow \theta - \epsilon \frac{\partial}{\partial \theta}(L),$$

где L – средняя потеря за тренировочный набор. Каждый шаг перемещает модель на небольшое расстояние в направлении «вниз» и немного меняет каждый из параметров модели для уменьшения средней потери. Если звезды вам благоволят, а Луна вошла в правильную фазу, то в конечном итоге вы получите параметры, которые помогут решить вашу проблему. Коэффициент ϵ называется *скоростью обучения* (learning rate) и определяет степень изменения параметров на каждом шаге. Его нужно выбирать очень осторожно: слишком маленькое значение приведет к тому, что обучение будет очень медленным, а слишком большое значение вообще не позволит модели обучиться.

Этот алгоритм действительно работает, но у него есть серьезная проблема. Для каждого шага градиентного спуска вам нужно пройтись по каждой выборке в обучающем наборе. Это означает, что время, необходимое для обучения модели, пропорционально размеру обучающего набора!