УДК 004.85
ББК 32.971.3
Ф43

Ферлитш Э.
Ф43 Шаблоны и практика глубокого обучения / пер. с англ. А. В. Логунова. – М.: ДМК Пресс, 2022. – 538 с.: ил.

ISBN 978-5-93700-113-9

В книге рассматриваются актуальные примеры создания приложений глубокого обучения с учетом десятилетнего опыта работы автора в этой области. Вы сэкономите часы проб и ошибок, воспользовавшись представленными здесь шаблонами и приемами. Проверенные методики, образцы исходного кода и блестящий стиль повествования позволят с увлечением освоить даже непростые навыки. По мере чтения вы получите советы по развертыванию, тестированию и техническому сопровождению ваших проектов.

Издание предназначено для инженеров машинного обучения, знакомых с Python и глубоким обучением.

УДК 004.85
ББК 32.971.3

ISBN 978-1-6172-9826-4 (англ.)
ISBN 978-5-93700-113-9 (рус.)

Оглавление

Содержание

Предисловие

Одна из моих обязанностей в качестве сотрудника Google состоит в том, чтобы обучать инженеров-программистов приемам применения машинного обучения. У меня уже был опыт создания онлайновых учебных занятий, встреч, презентаций на конференциях, рабочих семинаров и курсовых работ для частных школ программирования и аспирантур университетов, но я всегда ищу новые способы эффективного преподавания.

До Google я в течение 20 лет проработал в японской информационно-технологической индустрии в качестве главного научного сотрудника – и все время без глубокого обучения. Почти все, что я вижу сегодня, мы делали в инновационных лабораториях 15 лет назад; разница лишь в том, что нам нужен был коллектив, полный ученых, и огромный бюджет. Невероятно, как все так быстро поменялось в результате повсеместного внедрения технологии глубокого обучения.

Еще в конце 2000-х годов я работал с небольшими структурированными наборами геопространственных данных из национальных и международных источников, разбросанных по всему миру. Коллеги называли меня исследователем данных, но никто не знал, что это такое на самом деле. Затем появились большие данные, которые проявили мою неосведомленность об инструментах и каркасах больших данных, и я неожиданно перестал быть исследователем данных. Вот незадача. Мне пришлось поднапрячься и изучить инструменты и концепции, лежащие в основе больших данных, и я снова стал исследователем данных.

И вот появилось машинное обучение на больших наборах данных, такое как линейная/логистическая регрессия и анализ CART, а я не использовал статистику со времен аспирантуры десятилетней давности, и я снова перестал быть исследователем данных. Вот дела! Мне пришлось поднапрячься и выучить статистику заново, и я снова стал исследователем данных. Затем пришло глубокое обучение, а я не знал теории и основ нейронных сетей, и я внезапно перестал быть исследователем данных. Что опять? Но я снова поднапрягся и изучил теорию и другие основы глубокого обучения. И опять-таки, теперь я – снова исследователь данных.

Об этой книге

Кому следует прочитать эту книгу

Добро пожаловать в мое последнее начинание – книгу «Шаблоны и практика глубокого обучения». Эта книга предназначена для инженеров-программистов, инженеров машинного обучения, а также младших, средних и старших специалистов-исследователей данных. Хотя вы, возможно, посчитаете, что начальные главы будут полезны для последней группы, мой уникальный подход, скорее всего, даст вам дополнительную информацию и поможет освежить знания. Книга построена так, чтобы каждый читатель достиг точки «зажигания» и смог продвигаться вперед к глубокому обучению самостоятельно.

Я преподаю шаблоны конструирования и образцы практики главным образом в контексте компьютерного зрения, так как именно здесь шаблоны конструирования появились для глубокого обучения впервые. Разработки в области понимания естественного языка и моделей структурированных данных отставали и по-прежнему были сосредоточены на классических подходах. Но по мере того, как они догоняли, эти области вырабатывали свои собственные шаблоны конструирования для решения задач глубокого обучения, и я излагаю эти шаблоны и образцы практики на протяжении всей книги.

Несмотря на то что я демонстрирую исходный код моделей компьютерного зрения, мое внимание сосредоточено на концепциях, лежажих в основе подходов и инноваций: тому, как они устроены и почему они устроены таким образом. Указанные опорные концепции применимы к обработке естественного языка, структурированным данным, обработке сигналов и другим областям, и, резюмируя, вы должны быть в состоянии адаптировать эти концепции, методы и образцы практики к задачам в вашей предметной области. Многие модели и методы, которые я обсуждаю, не зависят от предметной области, и на протяжении всей книги, где это уместно, я также обсуждаю ключевые инновации в областях обработки естественного языка, понимания естественного языка и структурированных данных.

Если говорить об общей подготовке, то вы должны знать, по крайней мере, основы Python. Все в порядке, если вы все еще пытаетесь разобраться в том, что такое включение в список или генератор, или если у вас все еще есть некоторая путаница в отношении странного среза многомерного массива и того, какие объекты являются мутируемыми и немутируемыми в куче. Для этой книги данного уровня будет достаточно.

Какой должна быть подготовка тех инженеров-программистов, которые хотят стать инженерами машинного обучения? Инженер машинного обучения (MLE) – это инженер-прикладник. Вам не требуется знать статистику (реально не нужно!), и вам не требуется знать теорию вычислений. Если вы заснули в колледже на уроке математики на теме производной, то это нормально, и если кто-то попросит вас выполнить матричное умножение, не стесняйтесь спрашивать, зачем оно нужно.

Ваша задача состоит в том, чтобы изучить «кнопки и рычаги» вычислительного каркаса и применять свои навыки и опыт для выработки решений реально существующих задач. Вот в чем я собираюсь вам помочь, и вот в чем суть шаблонов конструирования с использованием модуля TF.Keras.

Эта книга предназначена для инженеров машинного обучения и исследователей данных на сопоставимых уровнях. Тем же, кто следует по пути анализа данных, я рекомендую изучить дополнительные материалы, связанные со статистикой.

Прежде чем мы начнем, я хочу объяснить, как вы будете учиться, поэтому в данном первом разделе больше рассказывается о моей философии и подходе к преподаванию. Затем мы рассмотрим некоторые основополагающие материалы, включая терминологию, переход от классического или семантического ИИ к узкому или статистическому ИИ, а также проведем обзор основных шагов машинного обучения. Наконец, мы подробно остановимся на том, чему посвящена книга: на современном подходе к машинному обучению, основанному на *консолидации моделей*.

Я не использую традиционный западный подход: заучивание наизусть, повторение, повторение, проверка правильности ответов и затем продвижение вертикально вверх. Помимо моего мнения о том, что такой подход к преподаванию менее эффективен, я считаю, что он непреднамеренно дискриминирует учащихся.

Вместо этого я имел возможность преподавать инженерное дело и естественные науки в различных культурах и методиках преподавания и разработал уникальный стиль преподавания, с привлечением того, что я называю *боковым подходом*: я начинаю с ключевых понятий, а затем продвигаюсь по спирали, используя то, что я называю *абстракцией*. Когда начнут задаваться вопросы, я постепенно перехожу к указыванию другим студентам на их мысли по поводу ответов на эти вопросы, а потом размышляю над их мыслями. Я не провожу контрольные работы, в которых студенты пытаются получить 100 %. Вместо этого я даю задания, которые каждый студент провалит. Я по-

зволяю студентам биться над задачей изо всех сил, и при этом они начинают открывать для себя опорные принципы того, что им нужно усвоить. Например, я могу дать задание натренировать стандартную модель ResNet50 с использованием набора данных CIFAR-10, отметив, что авторы соответствующих статей по ResNet достигли на CIFAR-10 точности 97 %. Каждый студент провалит решение, модель не сойдется, они не наберут более 70 % и так далее.

Затем я собираю студентов в группы, чтобы решать задачи вместе. Совещаясь друг с другом, они учатся делать совместные обобщения. И прежде чем они достаточно созреют, я совершаю прыжок, в котором ставлю перед студентами еще одну трудноразрешимую задачу, – и процесс начинается снова. Я никогда не даю студентам возможности заучивать наизусть.

Используя свой пример, я могу разместить на доске четыре возможных решения, например: 1) обогащение изображений, 2) еще больше регуляризации, 3) еще больше гиперпараметрического поиска, 4) отложить снижение размера изображения глубоко внутрь нейронной сети (это правильный ответ). Далее в середине я останавливаю студентов и прошу каждую группу указать опробованное ими решение и то, что они усвоили к тому моменту. Затем я объясняю причины правильности/неправильности каждого решения, а потом снова меняю задачу.

По мере того как студенты переходят на более продвинутые уровни, я переключаюсь с роли учителя на то, что я называю ролью магистранта, и участвую в обучении. Студенты учат меня и друг друга так же, как я учу их. Я наблюдаю за каждым студентом и ищу то, что я называю *зажиганием*, – этап, когда студент начнет саморазвиваться как ученик, то есть когда он учится постоянно. В своем методе преподавания я замечаю, что весь класс собирается вместе и ни один ученик не остается позади.

Время от времени на одно из моих занятий приходил администратор школы программирования. Он слышал доносящуюся от студентов болтовню и хотел понаблюдать за тем, как это работает. Разумеется, администраторы испытывают потребность в том, чтобы всему давать название. В одной частной школе программирования администратор описал мою методику как «каждый становится учеником». Студенты учатся у учителя, учитель учится у студентов, и студенты учатся у студентов. Администратор назвал ее «Давайте учиться вместе».

У меня же для моей методики преподавания есть собственное название, а именно «Я верю в себя». Я часто говорю своим ученикам: как можно верить в меня («учителя»), если вы прежде всего не верите в себя?

Как эта книга организована: дорожная карта

Эта книга состоит из трех частей: основы, общие шаблоны конструирования и шаблоны конструирования для решения задач тренировки и развертывания в производстве.

Часть I «Основы глубокого обучения» предоставляет читателям обновленную информацию о глубоком обучении, которая включает введение в сверточные нейронные сети, а также обсуждение концепций и терминологии, которые являются сегодня магистральными для всех областей – компьютерного зрения, обработки естественного языка и структурированных данных.

Шаблоны конструирования моделей представлены в части II «Базовые шаблоны конструирования». В главах 5–7 я ввожу современные шаблоны конструирования и способы их применения ко многим современным и некогда передовым моделям глубокого обучения. Я расскажу о шаблоне процедурного реиспользования, который был преобладающим подходом для моделей, которые конструировались вручную. Я излагаю подходы к конструированию, усовершенствования и плюсы/минусы крупных моделей, обнаруженных исследователями для движения послойно вглубь (глава 5), движения послойно вширь (глава 6), и применению альтернативных или готовых («прямо из коробки») шаблонов связности (глава 7).

В главе 5 рассматривается шаблон процедурного конструирования для сверточных нейронных сетей, а также разработка остаточных блоков с отождествляющими связями с вниманием в трансформерах для понимания естественного языка.

В главе 6 подробно рассказывается о шаблоне процедурного конструирования для сверточных нейронных сетей и о том, как исследователи разведали движение послойно в ширину в качестве альтернативы движению в глубину. Я показываю, каким образом такой подход, как ResNeXt, привел к достижению сравнимой точности в сопоставлении с глубокими слоями с меньшей подверженностью запоминанию и исчезающим градиентам. Я также проведу разведывательный анализ степени релевантности широких сверточных нейронных сетей для разработок в широких и глубоких моделях TabNet для структурированных данных.

В главе 7 рассматриваются шаблоны конструирования моделей. Указанные шаблоны разведывают другие альтернативные соединения между слоями, чтобы двигаться послойно вглубь либо вширь с целью повышения точности, сокращения числа параметров и увеличения прироста информации в промежуточном латентном пространстве внутри модели.

В главе 8 инспектируются уникальные конструктивные соображения и особые ограничения для мобильных сверточных нейронных сетей. Из-за ограничений этих устройств по памяти необходимо учитывать компромиссы между размером и точностью. Я расскажу о прогрессе в этих компромиссах, плюсах/минусах и о том, как конструкции мобильных сетей отличаются от их крупномодельных визави, чтобы учесть эти компромиссы.

В главе 9 представлены автокодировщики для неконтролируемого обучения. Практическая применимость автокодировщиков в качестве автономных моделей очень узка. Но сделанные автокодировщи-

ками открытия способствовали прогрессу в предтренировке моделей. Такие модели лучше обобщают на обслуживание запросов вне распределения, то есть на предсказательные запросы к модели, развернутой в производственной среде, которые имеют иное распределение, чем данные, на которых модель была натренирована. Я также разведаю сопоставимость автокодировщиков с векторными вложениями в отрасли понимания естественного языка.

Все модели во второй части этой книги внесли эпохальный вклад в исследования и разработки в области глубокого обучения и продолжают использоваться сегодня, либо их вклад был включен в современные модели.

В части III «Работа с конвейерами» рассматриваются шаблоны конструирования и образцы практики для производственных конвейеров. В главе 10 мы рассмотрим гиперпараметрическую настройку, как ручную, так и автоматическую. Я изложу конструктивные решения, плюсы/минусы и лучшие практические приемы определения пространства поиска и шаблонов поиска в нем.

В главе 11 обсуждается тема переноса обучения (трансферного обучения) и вводятся концепции и методы манипулирования переносом весов и настройки под аналогичные и отдаленные задачи. Я также рассматриваю приложение по переносу предметных знаний с целью реиспользования весов во время предтренировки; данное приложение предназначено для моделей, которые тренируются полностью с нуля.

В главах с 12 по 14 выполняется высокоуровневый обзор производственных конвейеров. В главах 12 и 13 мы погрузимся в эту тему со стороны данных. Глава 12, в которой рассказывается о распределении данных, является единственной, в которой статистика излагается подробно. С 2017 года, когда можно было ожидать, что специалист будет обладать знаниями в области статистики на уровне доктора философии, многое изменилось. Сегодня многое из того скрыто или же автоматизировано в каркасах глубокого обучения, таких как TensorFlow. Понимание распределения данных и пространства поиска остается одной из преобладающих областей ожидаемых знаний из области статистики, и оно может существенно влиять на стоимость тренировки и способность модели обобщать после ее развертывания в производстве.

Наконец, в главах 13 и 14 мы переходим со стороны данных на сторону развертывания. Я рассказываю о концепциях и лучших образцах практики конструирования со стороны данных, а затем со стороны тренировки производственного конвейера.

Об исходном коде

Эта книга содержит массу примеров исходного кода как в пронумерованных листингах, так и во вставках, встроенных в обычный текст. В обоих случаях исходный код отформатирован шрифтом фиксированной

ширины, чтобы отделять его от обычного текста. Иногда исходный код также выделяется **жирным шрифтом**, чтобы выделять исходный код, который изменился по сравнению с предыдущими шагами в главе, например когда в существующую строку исходного кода добавляется новая функция.

Во многих случаях исходный код был переформатирован; мы добавили разрывы строк и переработали отступы, чтобы уложиться в доступное пространство книжной страницы. Кроме того, комментарии в исходном коде часто из листингов удалялись, когда исходный код описывался в тексте. Многочисленные листинги сопровождаются аннотациями исходного кода, выделяя важные концепции.

Все приводимые в книге примеры исходного кода написаны на Python и являются рабочими; правда, в них могут отсутствовать инструкции импорта. Во многих случаях образцы исходного кода являются частью более крупного компонента, такого как модель. В подобных случаях весь исходный код доступен в моем публичном репозитории для связей с разработчиками Google Cloud AI на GitHub (https://github.com/GoogleCloudPlatform/keras-idiomatic-programmer/tree/master/zoo).

Другие онлайновые ресурсы

Я использую вычислительный каркас TensorFlow 2.x, в который включен API моделей Keras. Я думаю, что сочетание этих двух факторов является фантастическим средством для образования, выходящим за рамки их производственной ценности.

Материал книги является мультимодальным. В дополнение к книге и полным образцам исходного кода в репозитории на моем YouTube-канале для связей с разработчиками Google Cloud AI (www.youtube.com/канал/uc8ov0vkzhtp8_puwedzlbjg) есть слайды презентаций, семинары, лабораторные работы и предварительно записанные лекции по каждой главе.

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 на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и 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.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Manning Publications очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Об авторе

Я твердо верю, что мой жизненный опыт делает меня одним из самых идеальных людей для преподавания концепций глубокого обучения. Когда эта книга выйдет из печати в первый раз, мне будет почти 60 лет. Я обладаю богатыми знаниями и опытом, которые сегодня соответствуют ожиданиям сотрудников. В 1987 году я получил ученую степень в области искусственного интеллекта. Я специализировался на обработке естественного языка. Когда я закончил колледж, то думал, что буду писать говорящие книги. Как оказалось, это было время зимы искусственного интеллекта.

В начале своей карьеры я выбрал другие направления. Прежде всего я стал экспертом в области государственной безопасности для мейнфреймов. По мере того как я набирался все больше опыта в конструировании и программировании ядер операционных систем, я стал разработчиком ядра для UNIX, будучи одним из авторов современного тяжеловесного ядра UNIX. В те же годы я участвовал в популяризации свободно распространяемого ПО «shareware» (еще до раскрытия исходного кода) и был основателем WINNIX, условно-бесплатной программы, которая конкурировала с коммерческим инструментарием MKS для исполнения оболочки UNIX и команд в среде DOS.

Впоследствии я разработал низкоуровневый инструментарий объектного кода. В начале 1990-х я стал экспертом как в области вычислений на защищенном уровне, так и в области компиляторов/ассемблеров для массово-параллельных вычислений. Я разработал инструмент MetaC, который обеспечивал инструментальную поддержку ядер операционных систем как традиционных операционных систем, так и высокозащищенных и массово-параллельных компьютеров.

В конце 1990-х годов я сменил карьеру и стал научным сотрудником японской корпорации Sharp. Через пару лет я стал главным научным сотрудником этой компании в Северной Америке. За 20-летний период Sharp подала более 200 заявок на патенты в США на мои исследования, из которых 115 были удовлетворены. Мои патенты охватывали области солнечной энергетики, телеконференций, ви-

зуализации, цифровых интерактивных вывесок и автономных транспортных средств. Кроме того, в 2014–2015 годах я был признан ведущим мировым экспертом по открытым данным и онтологиям данных и основал организацию opengeocode.

В марте 2017 года, по настоянию моего друга, я решил посмотреть, «что это за диковинка такая, которую называют глубоким обучением». Для меня это было естественно. У меня был большой опыт работы с данными, я работал специалистом и исследователем по обработке изображений, имел степень магистра искусственного интеллекта, работал над автономными транспортными средствами – все это, казалось, укладывалось в одну линию. И стало быть, я совершил прыжок.

Летом 2018 года Google обратилась ко мне с просьбой стать сотрудником Google Cloud AI. Я принял должность в октябре того же года. Это был и остается великолепный опыт работы в Google. Сегодня я работаю с огромным числом экспертов в области искусственного интеллекта как в Google, так и с корпоративными клиентами Google, обучая, наставляя, консультируя и решая задачи по внедрению глубокого обучения в больших производственных масштабах.

Часть I

Основы машинного обучения

В этой части вы изучите основы, необходимые для того, чтобы приступить к строительству моделей глубокого обучения. Мы начинаем с базовых принципов и шагов глубоких нейронных сетей (deep neural networks, аббр. DNN), с многочисленных диаграмм, иллюстрирующих эти шаги, а также с фрагментов исходного кода, которые имплементируют эти шаги. Я опишу каждый шаг, а затем расскажу об исходном коде. Далее мы рассмотрим принципы и шаги работы сверточных нейронных сетей (convolutional neural networks, аббр. CNN). Я проведу вас по эпохальным шаблонам конструирования. Указанные шаблоны лежали в основе ранних передовых сетей VGG и ResNet. Вы научитесь кодировать каждую из этих модельных архитектур, имея в распоряжении полный исходный код, который находится в общедоступном репозитории книги на GitHub.

Что будет дальше, после того как вы начнете кодировать сверточные нейросети? Вы будете их тренировать. Мы завершаем эту часть усвоением принципиальных знаний, касающихся тренировки сверточных нейросетевых моделей.

1 Конструирование современного машинного обучения

Эта глава охватывает следующие ниже темы:

- эволюция от классического ИИ к новейшим подходам;
- применение шаблонов конструирования для глубокого обучения;
- знакомство с шаблоном конструирования, именуемым «процедурным реиспользованием», для моделирования нейронных сетей.

Последняя революция в области глубокого обучения, с внедрением подхода, который я назвал, работая в Google Cloud AI, *консолидацией моделей* (model amalgamation), происходит не на микро-, а на макроуровне. При таком подходе модели разбиваются на составные единицы, коллективно использующие и адаптирующие компоненты с целью достижения разных целевых задач с одинаковыми первоначальными данными. Указанные компоненты взаимосвязаны в различных шаблонах связности, в которых каждый компонент *усваивает* коммуникационные интерфейсы между моделями на основе конструкции, не нуждаясь в том, чтобы иметь приложение серверного типа (бэкенд).

В дополнение к этому консолидация моделей может использоваться для тренировки устройств интернета вещей (IoT) с целью обогащения данных, преобразуя датчики интернета вещей из статических в динамически обучающиеся устройства – данный технический прием получил название *сращивание моделей*. Консолидация предоставляет средства для внедрения ИИ в производство в масштабах и слож-

ности эксплуатации, немыслимых в 2017 году, когда только-только начиналось движение в сторону производства.

Подумайте, например, об операционной сложности визуальных данных об объектах недвижимости, связанной с различными аспектами рынка аренды, таких как цены, состояние объекта недвижимости и удобства. Используя подход в форме консолидаций моделей, можно создать конвейер анализа общей картины, который соединяет компоненты отдельных моделей, каждый из которых работает над одним из этих аспектов. В конце у вас будет система, которая автоматически учится определять состояние, удобства и общую привлекательность рынка с соответствующими и надлежащими расценками арендной платы.

Подход в форме консолидаций моделей побуждает инженеров рассматривать модели как шаблоны или заготовки конструктивных решений, которые можно приспосабливать для создания индивидуальных компонентов. Поэтому если вы надеетесь использовать этот подход, то вам нужно будет понять конструкции ключевых моделей и систем, разработанных другими инженерами для решения задач, подобных тем, с которыми столкнетесь вы.

Цель этой книги состоит в том, чтобы помочь вам в этом глубоком понимании, познакомив вас с шаблонами конструирования эпохальных моделей глубокого обучения, а также с конструкцией или системной архитектурой, которая сводит эти компоненты вместе для разработки, тренировки, развертывания и обслуживания более крупных систем глубокого обучения. Даже если вы никогда не работали с огромными консолидациями, применяемыми на производственных предприятиях, свободное владение опорными конструкциями этих моделей и архитектур улучшит разработку любой создаваемой вами системы глубокого обучения.

1.1 Курс на адаптируемость

Поскольку эта книга предназначена для не совсем опытных инженеров глубокого обучения и исследователей данных, часть I начинается с конструкций базовых глубоких нейронных сетей (DNN), сверточных нейронных сетей (CNN) и остаточных нейронных сетей (ResNet). В части I также рассматривается архитектура простых конвейеров тренировки. Об этих сетях и архитектурах и только о них в свое время были написаны целые книги, поэтому здесь вы получите больше напоминаний о том, как они работают, с акцентом на шаблонах и конструктивных принципах. Суть тут в том, чтобы изложить конструкцию базовых компонентов глубокого обучения, куда будут вписываться все модели, которые вы увидите в части II.

Тем не менее если вы разбираетесь в основах хорошо, то вы можете перейти непосредственно к части II, в которой рассматриваются модели, сыгравшие эпохальную роль в развитии глубокого обучения.

Мой подход заключается в том, чтобы давать информацию о конструкции каждой модели в объеме, достаточном для того, чтобы у вас была возможность с ними поиграть и обдумать решения задач искусственного интеллекта, с которыми вы, возможно, столкнетесь. Модели представлены более или менее хронологически, поэтому часть II также служит своего рода историей глубокого обучения с акцентом на эволюцию от одной модели к другой.

Далее, если производство на предприятиях переходит к автоматическому усвоению и развитию моделей, то вы можете задаться вопросом о ценности проведения ревизии этих сконструированных вручную, некогда передовых (state-of-the-art, аббр. SOTA) моделей. И тем не менее многие из этих моделей по-прежнему используются в качестве стоковых моделей, в особенности для переноса обучения (трансферного обучения). Некоторые же вообще не попали в производство, но сыграли неоценимую роль в открытиях, которые продолжают использоваться сегодня.

Разработка моделей для производства по-прежнему представляет собой комбинацию автоматического усвоения и ручного обучения, что часто имеет решающее значение для собственных нужд или достижений. Но конструирование вручную не означает, что нужно начинать с нуля; как правило, вы начинаете со стандартной модели и вносите изменения и корректировки. Для того чтобы делать это эффективно, вам нужно разбираться в том, как модель работает и почему она работает таким образом, понимать концепции, лежащие в основе ее конструкции, а также плюсы и минусы альтернативных строительных блоков, которые вы будете узнавать из других передовых моделей.

Заключительная часть книги посвящена глубокому погружению в шаблоны конструирования для тренировки и развертывания в производстве. Хотя не все читатели будут развертывать интересующие меня системы уровня производственного предприятия, я чувствую, что эта информация является актуальной для всех. Знакомство со многими типами и размерами систем, решающих самые разные производственные задачи, обязательно вам поможет, когда решение задачи потребует от вас нестандартного мышления. Чем больше вы знаете об опорных концепциях и конструкциях, тем умелее и способнее к адаптации вы становитесь.

Эта способность к адаптации, пожалуй, является самым ценным выводом из данной книги. Производство предусматривает огромное число движущихся частей и бесконечный поток «разводных гаечных ключей», бросаемых в смесь. Если инженеры или исследователи данных просто заучивают наизусть наборы воспроизводимых шагов в вычислительном каркасе, то как они будут справляться с разнообразием производственных задач, с которыми они будут сталкиваться, и увиливать от бросаемых в них разводных ключей? Работодатели ищут большего, чем просто навыки и опыт; они хотят знать, насколько вы технически адаптивны.

Вообразите себя на собеседовании: вы набираете высокие баллы по навыкам и опыту работы и справляетесь с производственной задачей кодирования машинного обучения. Затем рекрутеры бросают вам разводной ключ, неожиданную или необычную задачу. Они делают это, чтобы понаблюдать за тем, как вы обдумываете задачу, какие концепции вы применяете и обосновываете их, как вы оцениваете плюсы и минусы различных решений и каково ваше умение выполнять отладку. Это и есть способность к адаптации. И это то, что, я надеюсь, разработчики глубокого обучения и исследователи данных извлекут из данной книги.

1.1.1 Компьютерное зрение задает тон

Все эти концепции я преподаю в первую очередь в контексте компьютерного зрения, потому что шаблоны конструирования впервые появились именно в компьютерном зрении. Но они применимы и к обработке естественного языка (NLP), структурированным данным, обработке сигналов и другим областям. Если откатить время назад в период до 2012 года, то во всех областях машинного обучения использовались главным образом классические методы, основанные на статистике.

Различные академические исследователи, такие как Фей-Фей Лю (Fei-Fei Liu) из Стэнфордского университета и Джеффри Хинтон (Geoffrey Hinton) из Университета Торонто, стали пионерами в применении нейронных сетей к компьютерному зрению. Лю вместе со своими учениками собрала набор данных по компьютерному зрению, теперь известный как ImageNet, с целью продвижения исследований в области компьютерного зрения. В 2010 году ImageNet наряду с набором данных PASCAL стал основой для ежегодного конкурса ImageNet Large Scale Vision Recognition Challenge (ILSVRC). На ранних стадиях там использовались традиционные методы распознавания изображений/обработки сигналов.

Затем, в 2012 году, Алекс Крижевский (Alex Krizhevsky), также из Университета Торонто, продемонстрировал модель глубокого обучения AlexNet, используя слои свертки. Эта модель выиграла конкурс ILSVRC, и к тому же со значительным отрывом. Модель AlexNet, разработанная совместно с Хинтоном и Ильей Суцкевером (Ilya Sutskever), положила начало глубокому обучению. В своей статье «Классифицирование ImageNet с помощью глубоких сверточных нейронных сетей» (ImageNet Classification with Deep Convolutional Neural Networks, http://mng.bz/1ApV) они показали подход к конструированию нейронных сетей.

В 2013 году Мэтью Зейлер (Matthew Zeiler) и Роб Фергус (Rob Fergus) из Нью-Йоркского университета победили в конкурсе, доработав AlexNet до версии, которую они назвали ZFNet. И эта регулярность развития успеха друг друга продолжилась. Группа по визуальной геометрии в Оксфорде расширила конструктивные принципы AlexNet

и выиграла конкурс 2014 года. В 2015 году Каймин Хе (Kaiming He) и другие сотрудники Microsoft Research расширили конструктивные принципы AlexNet/VGG и выиграли конкурс, представив новые шаблоны конструирования. Их модель, ResNet, и их статья «Глубокое остаточное обучение для распознавания изображений» (Deep Residual Learning for Image Recognition, https://arxiv.org/abs/1512.03385) вызвали всплеск интереса к нащупыванию и разведыванию конструктивного пространства сверточных нейросетей.

1.1.2 За пределами компьютерного зрения: обработка ЕЯ, понимание ЕЯ, структурированные данные

В эти первые годы разработки принципов и шаблонов конструирования с использованием глубокого обучения для компьютерного зрения, разработки моделей понимания естественного языка (NLU) и структурированных данных отставали и продолжали концентрироваться на классических подходах. В них для ввода текста использовались классические каркасы машинного обучения, такие как Естественно-языковой инструментарий (NLTK), и для ввода структурированных данных – классические алгоритмы, основанные на деревьях решений, такие как случайный лес.

В сфере понимания естественного языка (NLU) с внедрением рекуррентных нейросетей (RNN) и слоев с долгой кратковременной памятью (LSTM) и вентильного рекуррентного блока (GRU) был достигнут прогресс. В 2017 году этот прогресс привел к скачку с введением трансформерного шаблона для естественного языка и публикацией соответствующей статьи «Внимание – это все, что вам нужно» Ашиша Васвани и соавт. (Ashish Vaswani, Attention Is All You Need, https://arxiv.org/abs/1706.03762). Исследовательская организация Google Brain по глубокому обучению в рамках Google AI осуществила раннее внедрение аналогичного механизма внимания в ResNet. Прогресс в шаблонах для структурированных данных эволюционировал аналогичным образом с внедрением шаблона широкой и глубокой модели, описанного в 2016 году в статье Хэн-Цзе Ченг и соавт. «Широкое и глубокое обучение для рекомендательных систем» (Tze Cheng, Wide & Deep Learning for Recommender Systems, https://arxiv.org/abs/1606.07792), опубликованной в агностичной к технологиям исследовательской группе Google Research.

Хотя, рассказывая об эволюции и современном состоянии дел в шаблонах конструирования, я сосредоточиваюсь на компьютерном зрении, где это уместно, я ссылаюсь на соответствующий прогресс в понимании естественного языка (NLU) и структурированных данных. Многие концепции в этой книге перекрестно применимы ко всем сферам и типам данных. Например, главы 2–4 охватывают универсальные основы, а все главы, кроме одной главы в части III, охватывают концепции, которые являются агностичными к модели и типу данных.

В главах, где это имеет смысл, в основном в части II, я привожу пример из области, выходящей за рамки компьютерного зрения. Например, в главе 5 я сравниваю разработку остаточных блоков с отождествляющими связями с вниманием в трансформерах для понимания естественного языка (NLU). В главе 6 мы проведем разведывательный анализ релевантности широких сверточных нейросетей для разработок в широких и глубоких и TabNet-моделях для структурированных данных. В главе 9 объясняется сопоставимость автокодировщиков с вложениями в понимании ЕЯ, а в главе 11 рассматривается сопоставимость шагов переноса обучения с пониманием ЕЯ и со структурированными данными.

Отталкиваясь от примеров, которыми я с вами делюсь из компьютерного зрения, обработки естественного языка (NLP) и структурных данных, вы должны быть в состоянии адаптировать концепции, методы и технические приемы к задачам в вашей области.

1.2 Эволюция подходов, основанных на машинном обучении

В целях понимания современного подхода сначала необходимо понять, где мы находимся с ИИ и машинным обучением и как мы сюда попали. В этом разделе представлено несколько подходов и шаблонов конструирования верхнего уровня для работы в современной производственной среде, включая интеллектуальную автоматизацию, машинное конструирование, сращивание моделей и консолидацию моделей.

1.2.1 Классический ИИ против узкого ИИ

Давайте вкратце рассмотрим разницу между классическим ИИ и современным узким ИИ. В классическом ИИ (также именуемом семантическим ИИ) модели конструировались как системы, основанные на правилах. Эти системы использовались для решения задач, которые невозможно было решать с помощью математического уравнения. Вместо этого система организовывалась так, чтобы имитировать профильного эксперта, или эксперта в предметной области. На рис. 1.1 представлен наглядный пример такого подхода.

Классический ИИ хорошо работал в низкоразмерных входных пространствах (например, имел малое число отдельных входных значений); имел входное пространство, которое можно было разбить на дискретные сегменты, такие как категории или корзины; и поддерживал сильную линейную взаимосвязь между дискретным пространством и выходом. Профильный эксперт конструировал набор правил, основанных на входных данных и преобразованиях состояний, которые имитировали его опыт. Затем программа конвертировала эти

правила в систему, основанную на правилах, обычно в форме «Если *A* и *B* истинны, то *C* истинно».

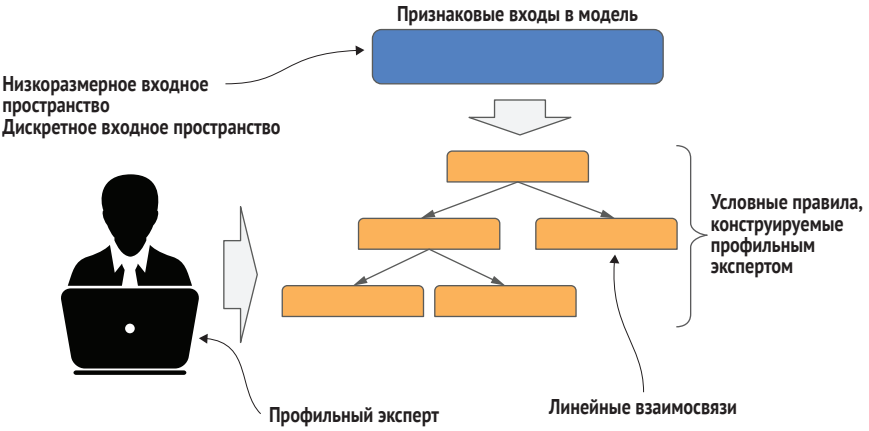

Рис. 1.1 В классическом подходе к ИИ профильный эксперт конструирует правила, имитирующие его знания

Подобного рода системы хорошо подходили для таких задач, как предсказывание качества и пригодности вина, что требовало лишь небольшого набора правил. Например, для селектора вин входными данными могли быть: обед или ужин, основное блюдо, повод и наличие десерта. Но классический ИИ не был способен масштабироваться до более крупных задач; точность резко падала, и правила требовали постоянного совершенствования, чтобы попытаться оттянуть падение. Неувязки между профильными экспертами, которые конструировали эти правила, были еще одной проблемой, приводящей к неточностям.

В *узком ИИ* (также именуемом *статистическим* ИИ) модель тренируют на крупном объеме данных, что уменьшает потребность в профильных экспертах. Вместо них в модели используются принципы статистики, позволяющие ей усваивать закономерности в распределении входных данных, т. н. *выборочном распределении*. Затем эти закономерности могут применяться высокоточно к образцам, которые не были замечены во время тренировки. После тренировки с использованием выборочного распределения, состоящего из крупных объемов данных, являющихся весомыми представителями более крупной популяции, или *популяционного распределения*, можно безгранично моделировать задачи, которые возникают в классическом ИИ. Другими словами, узкий ИИ очень хорошо может работать с существенно более высокой размерностью во входном пространстве (имея в виду крупное число несхожих входных данных) и с входными данными, которые могут быть как дискретными, так и непрерывными.

Давайте сопоставим ИИ, основанный на правилах, с узким ИИ, применив оба к предсказыванию продажной цены дома. Система,

основанная на правилах, обычно может учитывать лишь малый объем входных данных, например размер участка, площадь, число спален, число ванных и налог на имущество. Подобная система могла бы предсказывать среднюю цену для сопоставимых домов, но не для какого-либо отдельного дома из-за нелинейности во взаимосвязях объекта недвижимости с ценой.

Давайте отступим на один шаг назад и обсудим разницу между линейной и нелинейной взаимосвязями. В *линейной взаимосвязи* значение одной переменной предсказывает значение другой. Например, предположим, что у нас есть функция $y = f(x)$, которую мы определяем как $2 \times x$. Значение y может быть предсказано со 100%-ной уверенностью для любого значения x. В *нелинейной взаимосвязи* значение y может быть предсказано только с распределением вероятностей для любого значения x.

Используя наш пример с жильем, мы могли бы попытаться задать $y = f(x)$ как *продажная цена = кв_метры × цена_в_расчете_на_кв_метр*. Реальность такова, что на *цену_в_расчете_на_кв_метр* влияет множество других переменных, причем существует некоторая неопределенность в том, как они влияют на цену. Другими словами, квадратный метр дома имеет нелинейную взаимосвязь с продажной ценой, которая сама по себе может предсказывать только распределение вероятностей продажной цены.

В узком ИИ мы существенно увеличиваем число входов, чтобы усваивать нелинейности, например добавляем год постройки дома, дату выдачи разрешений на модернизацию, тип архитектуры, материалы, используемые для кровли и фасадного покрытия, информацию о школьном округе, возможности трудоустройства, средний доход, район, а также преступность и близость к паркам, общественному транспорту и автомагистралям. Эти дополнительные переменные помогают модели усваивать распределение вероятностей с высокой степенью уверенности. Входы, значение которых проистекает из фиксированного множества, такого как архитектура здания, являются *дискретными*, а входы, поступающие из неограниченного диапазона, такого как средний доход, являются *непрерывными*.

Модели узкого ИИ хорошо работают с входами, которые имеют высокий уровень нелинейности по отношению к выходам (предсказаниям), за счет усвоения границ, которые сегментируют входы, – опять же, если эти сегменты имеют строго линейную взаимосвязь с выходами. Такие типы моделей основываются на статистике, требуя крупных объемов данных, и называются *узким ИИ*, потому что они хороши в решении узких задач, состоящих из лимитированного круга задач внутри одной области. Узкие модели не так хороши в обобщении на задачи широкого масштаба. Рисунок 1.2 иллюстрирует подход на основе узкого ИИ.

Увидеть разницу между классическим ИИ и узким ИИ можно иначе, а именно глядя на снижение частоты ошибки в обоих типах моделей, поскольку глубокое обучение постоянно приближается к байесовому

теоретическому пределу ошибки. Байес описал этот теоретический предел ошибки в виде прогрессии, как показано на рис. 1.3.

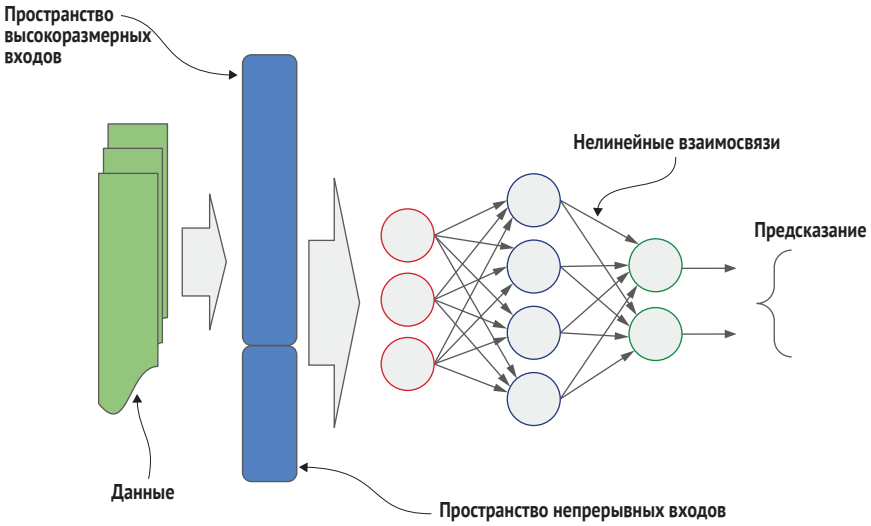

Рис. 1.2 В узком ИИ модель учится быть профильным экспертом, тренируясь на крупном наборе данных, являющемся весомым представителем более крупной популяции

Рис. 1.3 Машинное обучение продвинулось к байесовому теоретическому пределу ошибки

Прежде всего какова будет частота ошибок при решении задачи одним средним неэкспертом? Затем какова будет частота ошибок при решении задачи одним экспертом (это аналогично семантиче-

скому ИИ)? Какова будет частота ошибок при решении задачи группой экспертов? И наконец, теоретический предел: какова будет частота ошибок при решении задачи бесконечным числом экспертов?

Глубокое обучение в огромном числе задач компьютерного зрения и обработки естественного языка (NLP) позволило достичь частоты ошибок группой экспертов, значительно превысив как традиционные программные приложения, так и экспертные системы. В 2020 году исследователи и инженеры машинного обучения на производственных предприятиях начали разрабатывать производственные системы, которые соответствуют байесовому теоретическому пределу ошибки.

1.2.2 Следующие шаги в компьютерном обучении

Теперь, когда мы понимаем, как мы сюда попали, осталось выяснить, где, собственно говоря, мы находимся. По мере того как компьютерное обучение менялось, мы сначала переходили от искусственного интеллекта к интеллектуальной автоматизации. А затем к машинному конструированию, сращиванию моделей и консолидации моделей. Давайте дадим определения этим современным достижениям.

Интеллектуальная автоматизация

Как мы только что увидели, ранний искусственный интеллект означал классический ИИ, который основывался главным образом на правилах и требовал наличия профильных экспертов. Это позволило нам, по сути, написать программно-информационное обеспечение, чтобы начать автоматизировать задачи, которые обычно выполнялись вручную. Затем, в узком ИИ, мы применили статистику к самообучению, или усвоению, устранив необходимость в профильном эксперте.

Следующим крупным достижением стала *интеллектуальная автоматизация* (ИА, intelligent automation, аббр. IA). В этом подходе модели усваивают (почти) оптимальный способ автоматизирования процесса, превосходящий производительность и точность ручного или компьютерно-автоматизированного визави.

В типичной ситуации система ИА работает как конвейерный процесс. Кумулятивная информация, преобразования и переходы из состояния в состояние являются данными, которые поступают на вход модели в различных точках конвейера. Данные на выходе, или предсказание, из каждой модели используются для выполнения следующего преобразования информации и/или принятия решения о следующем переходе из состояния в состояние. В типичной ситуации каждая модель тренируется и развертывается независимо, обычно в виде микрослужбы, при этом всем конвейерным процессом руководит приложение серверного типа (внутреннее или бэкендовое).

Примером ИА является автоматизированное выделение информации о пациентах из медицинских записей из различных источников и форматов, включая источники, на которых модель никогда не тренировалась. В 2018-м я работал над архитектурным дизайном таких систем в области здравоохранения. Сегодня целый ряд поставщиков предлагают такие системы «под ключ»; Google Cloud Healthcare API (https://cloud.google.com/healthcare) является одной из них.

В 2019 году в огромном числе компаний размера промышленного предприятия ИИ двигался в сторону полноценного производства. В течение того года я проводил все большее число встреч с крупнейшими клиентами Google. И сейчас давайте поговорим об ИИ с точки зрения бизнеса. Упомянутые выше технологические концепции превратились в деловые концепции.

На этих встречах мы отошли от использования ИИ и заменили его на ИА, чтобы демистифицировать процесс. Мы просим клиента описать каждый шаг (ручной и компьютеризированный) в процессе, к которому они хотят применить ИИ. Предположим, что один шаг стоит 100 000 долларов. В прошлом нашей тенденцией было бы прыгнуть к этому шагу и применить ИИ – «большую награду». Но предположим, что еще один шаг стоит всего копейки, но случается миллион раз в день – а это 10 000 долларов в день, или 3.65 млн долларов в год. И давайте предположим, что мы могли бы заменить этот спелый и сочный плод моделью, которая усваивает оптимальный способ автоматизирования данного шага и которая в оперативном плане стоит 40 000 долларов в год. Ведь никто не оставит на столе 3.61 млн долларов.

Это и есть интеллектуальная автоматизация. Вместо того чтобы программисты кодировали ранее разработанный алгоритм автоматизации, программисты ориентируют модель, чтобы та разумно усваивала оптимальный алгоритм. На рис. 1.4 показано применение ИА к отдельному шагу в конвейере обработки заявок.

Давайте проведем высокоуровневый обзор происходящего в этом конвейере. На шаге 1 относящиеся к заявке документы сканируются и подаются в конвейер ИА. На шаге 2 предшествующая практика оператора документов, который поочередно просматривает каждый отсканированный документ и его размечает, заменяется естественно-языковой классификационной моделью, которая была натренирована выполнять эту задачу обработки заявок.

Это замещение имеет ряд преимуществ. Во-первых, исключается стоимость ручного труда. В дополнение к повышению скорости работы за счет компьютера по сравнению с человеком указанный процесс можно распределить таким образом, что можно будет обрабатывать массовое число документов в параллельном режиме.

Во-вторых, частота ошибок на правильной разметке классов документа существенно снижается по сравнению с частотой ошибок людей. Давайте подумаем, почему. Каждый человек-оператор имеет разный уровень подготовки и опыта, а также большое непостоянство

(дисперсию) в точности. Кроме того, увеличению частоты ошибок способствует усталость человека. Но давайте предположим, что мы располагаем одной тысячей обученных операторов-людей, которые просматривают один и тот же документ(ы), и для маркировки документа мы используем метод мажоритарного голосования. Мы ожидаем, что частота ошибок будет существенно снижена, приблизившись к нулю.

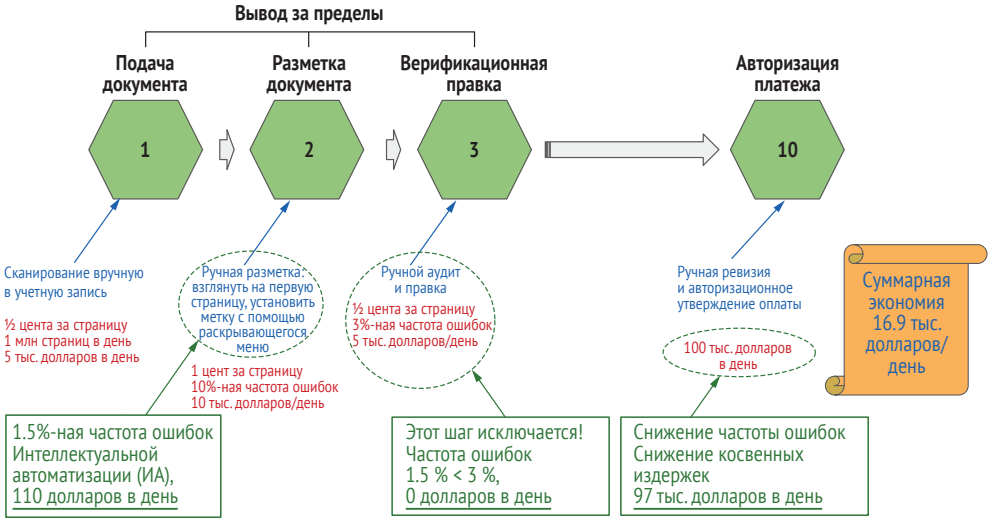

Рис. 1.4 Интеллектуальная автоматизация, применяемая для обработки заявок

Именно это и делает модель: она была натренирована на огромном числе документов, которые были помечены огромным числом обученных людей-операторов. Таким образом, после тренировки результативность модели равна результативности коллектива обученных людей-операторов.

На шаге 3 в ручной версии опытный оператор-человек инспектирует разметку, чтобы еще больше уменьшить число ошибок. Этот этап в процессе ИА не исключается, но благодаря существенному сокращению ошибок по сравнению с этапом 2 рабочая нагрузка на человека-оператора значительно снижается.

Нижестоящие процессы ИА продолжают снижать/устранять издержки на человека-оператора и еще больше снижать частоту ошибок. Как только мы перейдем к заключительному шагу, обученный профильный эксперт (subject-matter expert, аббр. SME) проводит заключительную ревизию на предмет авторизации (или неавторизации) платежа. Теперь, когда рассматриваемая профильным экспертом информация имеет более высокую точность, субъективное решение человека получается более надежным, что еще больше снижает издержки на принятие неправильного субъективного решения.

Мы в отрасли перестали использовать термин *машинное обучение* и заменили его *машинным конструированием*, чтобы провести аналогию с *компьютерным конструированием* (computer-aided design, аббр. CAD). Компьютерное конструирование применялось к задачам, которые были слишком сложными, чтобы разрабатывать даже субоптимальное решение. Эти системы имели строительные компоненты, математические знания и правила экспертной системы, а профильные эксперты ориентировали системы компьютерного конструирования, чтобы те отыскивали хорошее субоптимальное решение.

В отличие от них, в машинном конструировании система усваивает строительные компоненты, математические знания и правила сама, а инженер машинного обучения ориентирует машинное конструирование, чтобы то отыскивало оптимальное решение. Переходя к машинному конструированию, мы высвобождаем ценные человеческие ресурсы для решения сложных задач следующего уровня, ускоряя техническое развитие людей и обеспечивая бизнесу более высокую возвратность инвестиций (ROI) в расчете на одного сотрудника.

Машинное конструирование

До глубокого обучения профильные эксперты разрабатывали программно-информационные продукты для проведения поиска хороших решений в тех частях программно-информационного и аппаратного обеспечения, которые отличались высокой сложностью. Как правило, эти программы представляли собой комбинацию поисковой оптимизации и методов, основанных на правилах.

В следующем продвижении вперед, *машинном конструировании* (machine design), модели усваивают (почти) оптимальный способ конструирования и интегрирования программно-информационных и аппаратных компонентов. Эти системы превосходят по производительности, точности и сложности, даже если их сравнивать с моделями, разработанными профильными экспертами с помощью программы CAD. Конструктор-человек использует свой опыт работы в реальном мире, чтобы ориентировать поиск решений, проводимый моделью в пространстве поиска.

Рассмотрим больницу с двумя рентгеновскими отделениями; в одном отделении установлен дорогой рентгеновский аппарат, а в другом – недорогой. Осматривающий врач выбирает отделение, в которое направлять пациента для подтверждения диагноза пневмонии, *в зависимости от правдоподобной возможности, что у пациента есть пневмония*. Если такая возможность пневмонии невелика или маловероятна, то врач направляет пациента на недорогой рентгеновский аппарат, руководствуясь страховым полисом и желанием снизить расходы для страховой компании. Если определение является высоким или возможным, то пациент заслуживает дорогостоящего рентгеновского снимка. Это пример машинного конструирования, наполняющего пространство гиперпараметрического и архитектурного

поиска информацией, ориентирующий конвейер в системе автоматического усвоения медицинских снимков из разных распределений (медицинских устройств, в данном случае рентгеновских аппаратов).

Имейте в виду, что если для тренировки модели используются накопительные рентгеновские лучи и диагностические определения от двух рентгеновских устройств, то у нас будет смещение в данных. Вместо того чтобы учиться на данных, модель может непреднамеренно усвоить уникальные характеристики этих двух медицинских устройств – то есть смещение вследствие *точки зрения*. Классическим примером проблемы смещения в модели вследствие точки зрения является случай определения разницы между собаками и волками, в котором модель непреднамеренно усвоила снег как атрибут волков, поскольку все тренировочные снимки волков были сделаны зимой.

В машинном конструировании, помимо тренировки модели, система усваивает оптимальный тренирующий конвейер для антагонистической модели, именуемой *суррогатом*. Если вы хотите углубиться в эту тему подробнее, то статья «Антагонистический подход для устойчивого классифицирования пневмонии по рентгенограммам грудной клетки» (https://arxiv.org/pdf/2001.04051.pdf) является основополагающей работой по машинному конструированию, которая имеет прямую связь с упомянутой проблематикой.

Сращивание моделей

Сращивание моделей (model fusion) представляет собой следующий шаг в разработке более точных и недорогих систем для предсказательного технического сопровождения и обнаружения неисправностей наподобие тех, которые используются в сенсорных системах интернета вещей. Традиционно датчики интернета вещей встраивались в очень дорогое оборудование и инфраструктуру, такие как заводские станки, самолеты и региональные энергетические инфраструктуры. Непрерывные данные от этих датчиков передавались в разработанные экспертами алгоритмы, которые опирались на правила.

Проблема с этими традиционными системами заключалась в том, что они подвержены высокому непостоянству (дисперсии) окружающей среды, что влияет на их надежность. Например, в электроэнергетике на каждой вышке линии электропередачи установлены датчики, которые отслеживают наличие аномалий в линейном импедансе между вышками. Импеданс может колебаться в результате давления на линейное соединение из-за ветра, изменений температуры, влияющих на проводимость, и вторичных факторов, таких как накопление влаги.

Сращивание моделей повышает надежность систем интернета вещей посредством задействования машинно-усвоенной модели с более высокой операционной стоимостью с целью генерирования помеченныех данных, чтобы конвертировать систему, сконструированную экспертом. Продолжая наш пример с энергетикой, сегодня там используются беспилотные летательные аппараты и модели глу-

бокого обучения, натренированные в компьютерном зрении, чтобы периодически проводить инспекцию линий электропередач. Этот процесс является высокоточным и в операционном плане более дорогостоящим.

Вследствие этого он используется для того, чтобы генерировать помеченные данные для сенсорных импедансных данных, создаваемых более дешевой сенсорной системой. Помеченные данные, сгенерированные с высокой операционной стоимостью, затем используются для тренировки еще одной модели, которую импедансные датчики (недорогая система) затем применяют для достижения сопоставимой надежности.

Консолидация моделей

До глубокого обучения приложения создавались либо как монолитное, работающее на внутреннем (бэкендовом) сервере, либо как стержневая магистраль на сервере, использующем распределенные микрослужбы. В *консолидации моделей* модель (модели) по существу становится целым приложением, которое напрямую обменивается модельными компонентами и выходными данными и усваивает коммуникационный интерфейс между моделями. Все это происходит без необходимости в громоздком приложении серверного типа или микрослужбах.

Вообразите модельный конвейер для индустрии недвижимости, в котором используется анализ изображений на фотографиях домов и многоквартирных жилых зданий, чтобы определять расценки арендной платы. Модели внутри набора выстроены в цепочку, причем каждая модель натренирована на отдельной функциональности; вместе они автоматически определяют арендные условия, арендные удобства и привлекательность рынка и предлагают соответствующие расценки.

Давайте сравним это с более традиционным ИА, где каждый шаг в конвейере процесса является отдельным развернутым экземпляром модели, принимающим на входе изначальное изображение либо преобразование и состояние, все из которых контролируются основанным на правилах приложением серверного типа, разработанным экспертом(ами). Напротив, в консолидации экземпляры моделей обмениваются друг с другом напрямую. Каждая модель усвоила оптимальный метод выполнения своей специализированной работы (например, определение состояния); усвоила оптимальный путь обмена информацией и представление между моделями (моделями для дома, комнаты и удобств), а также усвоила оптимальный метод определения изменений состояния (условий объекта недвижимости).

Я ожидаю, что на уровне предприятия в 2021 году производство перейдет к консолидации моделей. Мы все еще пытаемся понять, как сделать так, чтобы все это работало как надо. До консолидации развертывалось большое число моделей, которые выполняли разные за-

дачи, и разработчики конструировали приложение серверного типа, которое осуществляло передачу состояния представления данных (REST) или вызывало микрослужбы. Мы по-прежнему кодировали логику приложения, а также интерфейс и обмен данными между приложением и моделями. На рис. 1.5 приведен пример консолидации моделей, которую я разработал в конце 2019 года для спортивного вещания.

Давайте пройдемся по этому процессу пошагово. В начале консолидация принимает видео в реальном времени; то есть консолидация обрабатывает видео непрерывно. Видео разбирается на части в реальном времени как набор кадров, расположенный во временной последовательности. Каждый кадр представляет собой изображение спортивного игрока, например бейсболиста, который вот-вот выполнит удар битой. Каждый кадр сначала обрабатывается совместным набором сверточных слоев (совместными сверточными слоями), который генерирует общую внутреннюю кодировку для нижестоящих задач. Другими словами, вместо того чтобы каждая нижестоящая задача (модель) начиналась с одного и того же входного изображения и обрабатывалась во внутреннюю кодировку, входное изображение кодируется один раз, а кодировка реиспользуется в нижестоящих компонентах. Это ускоряет отклик модели и сокращает ее размер в памяти.

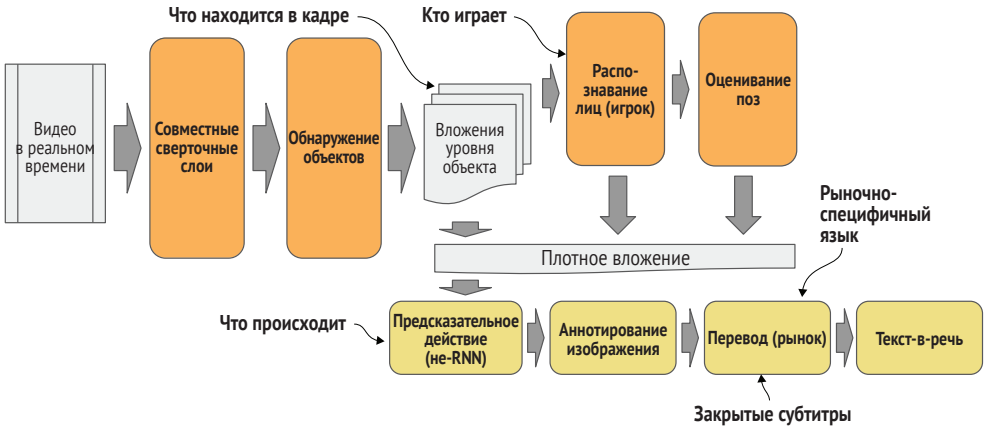

Рис. 1.5 Консолидация моделей применительно к спортивному вещанию

Затем сгенерированная общая кодировка проходит через модель обнаружения объектов, натренированную на общей кодировке вместо изображения на входе, сокращая размер и увеличивая скорость обнаружения объектов. Допустим, модель обнаружения объектов натренирована распознавать объекты, включая людей, игровое оборудование, персонал и поле. Для каждого объекта, который модель распознает в кадре, она будет выводить вложение уровня объекта, то есть низкоразмерное отображение/представление (например, коди-

ровку сокращенного размера) вместе с пространственными координатами внутри вышестоящего входного кадра.

Эти объектно-уровневые вложения теперь становятся входами для еще одного набора нижестоящих задач. Затем вы видите, что вложения, классифицированные как люди, передаются в модель распознавания лиц, которая была натренирована на вложениях, а не на изначальных изображениях. Модель может быть натренирована, например, распознавать игроков, официальных лиц, судей, тренеров и службу безопасности и соответственно помечать вложение. Объектные вложения, специфичные для каждого игрока, затем передаются в модель оценивания поз, которая отыскивает ключевые точки человека и классифицирует позу идентифицированного человека в кадре, например игрок А находится в положении отбивающего.

Затем вложения уровня объектов (игроки, механизмы, стадион и т. д.) объединяются с конкретной позой игрока в информационно обогащенное плотное вложение. И вся эта насыщенная информация передается в еще одну модель для предсказывания действий игрока, например игрок А находится у возвышения, готовый бросить мяч. Это предсказывающее действие потом передается в еще одну модель, конвертирующую действие в текст в виде закрытых субтитров, который накладывается на прямую трансляцию.

Давайте допустим, что спортивное событие транслируется по всему миру и его смотрят зрители на самых разных языках. Данные на выходе из модели аннотирования изображений (например, на английском языке) передаются в еще одну модель, которая выполняет перевод на язык, специфичный для каждого рынка. На каждом рынке переведенный текст конвертируется в речь для комментариев в реальном времени.

Как видите, модели эволюционировали от однозадачных предсказаний и автономных развертываний в модели, которые выполняют несколько задач, совместно используют модельные компоненты и интегрированы, чтобы формировать решения, такие как в примерах обработки медицинских документов и спортивного вещания. Такие интегрированные модельные решения можно описать и по-другому, как *конвейер обслуживания*. Конвейер состоит из соединенных между собой компонентов; выход из одного компонента является входом в другой, и каждый компонент можно конфигурировать, заменять, и каждый из них имеет версионный контроль и историю. Использование конвейеров в современном производственном машинном обучении распространяется на весь сквозной конвейер целиком.

1.3 Выгоды от шаблонов конструирования

До 2017 года большинство версий нейросетевых моделей во всех областях кодировались в стиле пакетных сценариев. По мере того как исследователи ИИ и опытные инженеры-программисты все активнее

вовлекались в исследования и конструирование, мы начали замечать сдвиг в кодировании моделей, отражавший принципы инженерии программно-информационного обеспечения, ориентированные на реиспользование и шаблоны конструирования.

Из *шаблона конструирования* следует, что существует лучший образец практики строительства и кодирования модели, который может применяться повторно в широком диапазоне случаев, таких как классифицирование изображений, обнаружение и отслеживание объектов, распознавание лиц, сегментирование изображений, наделение сверхразрешающей способностью и перенос стиля для данных, связанных со снимками и изображениями; классифицирование документов, анализ настроений, выделение сущностей и резюмирование текстовых данных; а также классифицирование, регрессия и предсказывание для неструктурированных данных.

Развитие шаблонов конструирования для глубокого обучения привело к консолидации моделей, сращиванию моделей и машинному конструированию, в которых модельные компоненты могут реиспользоваться и адаптироваться. Такие шаблоны конструирования модельных компонентов позволили исследователям и другим практикам глубокого обучения поступательно развивать как модельные компоненты, так и лучшие образцы практики для приложений, по всем моделям и по всем типам данных. Этот обмен знаниями ускорил развитие шаблонов конструирования и реиспользование модельных компонентов, что позволило развертывать глубокое обучение в широко распространенных производственных приложениях.

Многие исторически передовые модели, которые я рассматриваю в этой книге, раскрывают знания и концепции, встроенные в современное производство сегодняшнего дня. Несмотря на то что многие из этих моделей в конечном итоге перестают использоваться, понимание знаний, концепций и компонентов, лежащих в их фундаменте, имеет важное значение для практики глубокого обучения в современных крупных масштабах.

Одним из самых ранних шаблонов конструирования нейросетевых моделей стал шаблон *процедурного реиспользования* (procedural reuse), который был принят одновременно и повсеместно в компьютерном зрении, понимании естественного языка (NLU) и структурированных данных. Как и в случае с программно-информационным приложением, модель процедурного реиспользования конструируется в виде компонентов, которые отражают поток данных и раскладывают компоненты на реиспользуемые функции.

Многочисленные выгоды от применения шаблона процедурного реиспользования были и есть. Во-первых, он упрощает задачу представления моделей в архитектурных диаграммах. До использования формального шаблона каждый коллектив исследователей изобретал в публикуемых им статьях свой собственный способ представления своей модельной архитектуры. Указанный шаблон конструирования также определяет представление, которое задействуется для

структуры и потока модели. Наличие выверенного и усовершенствованного метода упростило изображение архитектурных диаграмм. Во-вторых, модельные архитектуры легче понимать другим исследователям и инженерам машинного обучения. Кроме того, работа по стандартному шаблону раскрывает внутреннюю работу конструкции, что, в свою очередь, облегчает работу по модифицированию моделей, устранению неполадок и проведению отладки.

В 2016 году авторы исследовательских статей начали показывать поток компонентов, традиционно именуемых *стержнем*, (представительным) *учеником* и (преобразовательной) *задачей*. До 2016 года авторы исследовательских работ представляли свои модели в виде монолитной архитектуры. Эти монолитные архитектуры усложняли исследователям задачу выстраивания доказательств того, что новая концепция являлась усовершенствованием любой отдельной части модели. Поскольку эти компоненты содержат повторяющиеся шаблоны потоков, в конечном итоге появилась концепция конфигурируемых компонентов. Эти повторяющиеся шаблоны потоков впоследствии реиспользовались и совершенствовались другими исследователями при конструировании своих модельных архитектур. Несмотря на то что применение модельных компонентов отставало в отрасли понимания естественного языка (NLU) и структурированных данных, к 2017 году мы начали встречать их появление в исследовательских работах и там. Сегодня, независимо от типа и сферы модели, модельная конструкция состоит из все тех же сопоставимых трех первичных модельных компонентов.

Более ранней версией шаблона конструирования, раскладывающего модель на компоненты, была архитектура SqueezeNet (https://arxiv.org/pdf/1602.07360.pdf), в которой использовались конфигурируемые компоненты на основе метапараметров. Введение метапараметров, описывающих процедуру конфигурирования модельных компонентов, помогло формализовать представление, конструкцию и имплементацию конфигурируемых компонентов. Разработка моделей на основе конфигурируемых компонентов предоставила исследователям возможность измерять улучшения производительности на покомпонентной основе, пробуя различные конфигурации компонентов. Такой подход к конструированию является стандартной практикой при разработке прикладного программно-информационного обеспечения; среди его многочисленных преимуществ следует отметить стимулирование им реиспользования исходного кода.

Шаблоны процедурного реиспользования были первыми и остаются наиболее фундаментальными реиспользуемыми конструкциями, поэтому они находятся в центре внимания данной книги. Позже для машинного конструирования будут введены так называемые шаблоны фабрики и абстрактной фабрики. В *фабричном шаблоне* конструирования в качестве фабрики и цели используются передовые строительные блоки, чтобы проводить поиск наилучшей соот-

ветствующей требованиям конструкции. *Абстрактный фабричный шаблон* абстрагируется на еще один уровень ниже и ищет наилучшую фабрику, которая затем используется для поиска наилучшей модели.

Однако в этой книге вы узнаете краеугольные конструкции, начав в части I с архитектур базовых глубоких нейросетей и сверточных нейросетей, перейдя в части II к эпохальным моделям, кодированным для процедурного реиспользования, и закончив в части III экскурсией по современному производственному конвейеру.

Резюме

- Глубокое обучение эволюционировало от классического ИИ к узкому ИИ, что привело к использованию ИИ для решения задач с высокоразмерными входными данными.
- Глубокое обучение эволюционировало от экспериментов с моделями к подходу на основе реиспользуемого и переконфигурируемого конвейера для обработки данных, тренировки, развертывания и обслуживания производственных запросов.
- Практики машинного обучения, работающие на переднем крае в масштабах производственного предприятия, используют консолидацию моделей, сращивание моделей и машинное конструирование.
- Шаблон процедурного реиспользования является строительным блоком и располагается непосредственно на переднем крае сегодняшнего дня в масштабе производственного предприятия.