Оглавление

Предисловие 20

Введение 22

Благодарности 24

Об этой книге 26

Для кого предназначена эта книга 26

Структура издания 27

О коде 29

Прочие источники информации в интернете 31

От издательства 32

Об авторах 33

Об иллюстрации на обложке 34

ЧАСТЬ I
Основы PyTorch

Глава 1. Знакомство с глубоким обучением и библиотекой PyTorch 36

1.1. Революция глубокого обучения 37

1.2. Использование PyTorch для глубокого обучения 39

1.3. Почему PyTorch? 41
1.3.1. Общая картина сферы глубокого обучения 42
1.4. Обзор средств поддержки библиотекой PyTorch проектов глубокого обучения 44
1.5. Аппаратные и программные требования 48
1.5.1. Блокноты Jupyter 49
1.6. Упражнения 50
1.7. Резюме 51

Глава 2. Предобученные сети 52
2.1. Предобученные сети для распознавания тематики изображения 53
2.1.1. Получение предобученной сети для распознавания изображений 56
2.1.2. AlexNet 57
2.1.3. ResNet 59
2.1.4. На старт, внимание, почти что марш 59
2.1.5. Марш! 62
2.2. Предобученная модель, создающая все лучшие подделки 64
2.2.1. Игра GAN. 65
2.2.2. CycleGAN. 67
2.2.3. Сеть, превращающая лошадей в зебр 68
2.3. Предобученная сеть для описания обстановки 72
2.3.1. NeuralTalk2 73
2.4. Torch Hub. 74
2.5. Итоги главы 76
2.6. Упражнения 76
2.7. Резюме 77

Глава 3. В начале был тензор... 78
3.1. Мир как числа с плавающей запятой 79
3.2. Тензоры: многомерные массивы 81
3.2.1. От списков Python к тензорам PyTorch 81
3.2.2. Создаем наши первые тензоры 82
3.2.3. Что такое тензоры 83

3.3. Доступ к тензорам по индексам 86
3.4. Поименованные тензоры 86
3.5. Типы элементов тензоров. 90
3.5.1. Задание числового типа с помощью dtype 91
3.5.2. dtype на все случаи жизни 91
3.5.3. Работа с атрибутом dtype тензоров. 92
3.6. API тензоров. 93
3.7. Тензоры: хранение в памяти 95
3.7.1. Доступ к хранилищу по индексу 95
3.7.2. Модификация хранимых значений: операции с заменой на месте. 97
3.8. Метаданные тензоров: размер, сдвиг и шаг. 97
3.8.1. Представления хранилища другого тензора 97
3.8.2. Транспонирование без копирования 100
3.8.3. Транспонирование при более высокой размерности 101
3.8.4. Непрерывные тензоры 102
3.9. Перенос тензоров на GPU 104
3.9.1. Работа с атрибутом device тензоров 105
3.10. Совместимость с NumPy 106
3.11. Обобщенные тензоры тоже тензоры 107
3.12. Сериализация тензоров 108
3.12.1. Сериализация в HDF5 с помощью h5py. 109
3.13. Итоги главы 110
3.14. Упражнения 111
3.15. Резюме. 111

Глава 4. Представление реальных данных с помощью тензоров 112
4.1. Работа с изображениями 113
4.1.1. Добавление цветовых каналов. 114
4.1.2. Загрузка файла изображения 115
4.1.3. Изменение схемы расположения 115
4.1.4. Нормализация данных 117
4.2. Трехмерные изображения: объемные пространственные данные . . .118
4.2.1. Загрузка данных в специализированном формате 119

4.3. Представление табличных данных 120
4.3.1. Реальный набор данных. 120
4.3.2. Загрузка тензора данных по вину. 122
4.3.3. Представление оценок. 124
4.3.4. Быстрое кодирование 125
4.3.5. Когда считать данные категориальными 127
4.3.6. Поиск пороговых значений 128
4.4. Временные ряды 131
4.4.1. Добавляем измерение времени 132
4.4.2. Компоновка данных по периоду времени. 133
4.4.3. Готов для обучения 135
4.5. Представление текста 138
4.5.1. Преобразование текста в числа 139
4.5.2. One-hot кодирование символов 139
4.5.3. Унитарное кодирование целых слов 141
4.5.4. Вложения текста 144
4.5.5. Вложения текста как схема. 146
4.6. Итоги главы 147
4.7. Упражнения 147
4.8. Резюме 148

Глава 5. Внутренняя кухня обучения. 149
5.1. Всегда актуальный урок моделирования 150
5.2. Обучение — это просто оценка параметров 152
5.2.1. «Жаркая» задача 154
5.2.2. Сбор данных 154
5.2.3. Визуализация данных 154
5.2.4. Выбираем линейную модель для первой попытки 155
5.3. Наша цель — минимизация потерь 156
5.3.1. Возвращаемся от задачи к PyTorch. 157
5.4. Вниз по градиенту 160
5.4.1. Снижение потерь 161
5.4.2. Выражаем аналитически 162
5.4.3. Подгонка модели в цикле. 164

5.4.4. Нормализация входных сигналов 167
5.4.5. Визуализируем (снова) 170
5.5. Компонент autograd PyTorch: обратное распространение всего чего угодно 171
5.5.1. Автоматическое вычисление градиента 171
5.5.2. Оптимизаторы на выбор 175
5.5.3. Обучение, проверка и переобучение 180
5.5.4. Нюансы автоматического вычисления градиентов и его отключение 186
5.6. Итоги главы 189
5.7. Упражнение 189
5.8. Резюме 190

Глава 6. Аппроксимация данных с помощью нейронной сети 191
6.1. Искусственные нейроны 192
6.1.1. Формирование многослойной сети 194
6.1.2. Функция ошибки 195
6.1.3. Все, что нам нужно, — это функция активации 195
6.1.4. Другие функции активации 198
6.1.5. Выбор наилучшей функции активации 199
6.1.6. Что обучение дает нейронной сети 200
6.2. Модуль nn PyTorch 203
6.2.1. Использование метода __call__ вместо метода forward 204
6.2.2. Обратно к линейной модели 205
6.3. Наконец-то нейронная сеть 210
6.3.1. Замена линейной модели 210
6.3.2. Просматриваем информацию о параметрах 212
6.3.3. Сравнение с линейной моделью 214
6.4. Итоги главы 215
6.5. Упражнения 215
6.6. Резюме 216

Глава 7. Различаем птиц и самолеты: обучение на изображениях 217
7.1. Набор крошечных изображений 218
7.1.1. Скачиваем CIFAR-10 218
7.1.2. Класс Dataset 219

7.1.3. Преобразования объектов Dataset 221
7.1.4. Нормализация данных 223
7.2. Различаем птиц и самолеты 226
7.2.1. Формирование набора данных 227
7.2.2. Полносвязная модель 227
7.2.3. Выходной сигнал классификатора 229
7.2.4. Представление выходного сигнала в качестве вероятностей 230
7.2.5. Функция потерь для классификации 234
7.2.6. Обучение классификатора 237
7.2.7. Ограничения, накладываемые полносвязностью 244
7.3. Итоги главы 246
7.4. Упражнения 247
7.5. Резюме 248

Глава 8. Обобщение с помощью сверток 249
8.1. Аргументы в пользу сверток 250
8.1.1. Что делают свертки 250
8.2. Свертки в действии 253
8.2.1. Дополнение нулями по краям 255
8.2.2. Обнаружение признаков с помощью сверток 257
8.2.3. Расширяем кругозор с помощью субдискретизации и повышения глубины сети 260
8.2.4. Собираем нашу нейронную сеть воедино 264
8.3. Создание подклассов nn.Module 266
8.3.1. Наша сеть как подкласс nn.Module 267
8.3.2. Как PyTorch отслеживает параметры и подмодули 269
8.3.3. Функциональные API 270
8.4. Обучаем нашу сверточную сеть 272
8.4.1. Измерение степени безошибочности 274
8.4.2. Сохранение и загрузка модели 275
8.4.3. Обучение на GPU 275
8.5. Архитектура модели 277
8.5.1. Расширение объема памяти: ширина 278

8.5.2. Улучшаем сходимость модели и ее способности к обобщению: регуляризация 280
8.5.3. Забираемся глубже для усвоения более сложных структур: глубина сети 285
8.5.4. Сравнение архитектур этого раздела. 291
8.5.5. Описанное здесь уже устарело 292
8.6. Итоги главы 292
8.7. Упражнения 293
8.8. Резюме 294

ЧАСТЬ II
Обучение на изображениях на практике: раннее обнаружение рака легких

Глава 9. Применение PyTorch в борьбе с раком 296
9.1. Постановка задачи 296
9.2. Подготовка к масштабному проекту 298
9.3. Что такое компьютерная томография 300
9.4. Проект: сквозной детектор рака легких 303
9.4.1. Почему нельзя просто передавать данные в нейронную сеть, пока она не заработает 308
9.4.2. Что такое узелок 313
9.4.3. Наш источник данных: The LUNA Grand Challenge 315
9.4.4. Загрузка данных LUNA 315
9.5. Итоги главы 317
9.6. Резюме 317

Глава 10. Объединение источников данных. 319
10.1. Файлы необработанных данных КТ 321
10.2. Парсинг данных аннотаций LUNA 322
10.2.1. Обучающие и проверочные наборы. 324
10.2.2. Объединение аннотаций и данных кандидатов 324
10.3. Загрузка сканов КТ. 327
10.3.1. Единицы Хаунсфилда 330
10.4. Определение положения узелка в системе координат пациента. . . 331
10.4.1. Система координат пациента. 332

10.4.2. Форма КТ-скана и размеры вокселя 334
10.4.3. Преобразование миллиметров в адреса вокселей 336
10.4.4. Извлечение узелка из скана КТ 337
10.5. Простая реализация Dataset 339
10.5.1. Кэширование массивов-кандидатов с помощью функции getCtRawCandidate 342
10.5.2. Построение набора данных в LunaDataset.__init__. 343
10.5.3. Разделение данных на обучающие и проверочные 343
10.5.4. Отображение данных. 345
10.6. Итоги главы 346
10.7. Упражнения 346
10.8. Резюме. 347

Глава 11. Обучение модели классификации обнаружению потенциальных опухолей 348
11.1. Базовая модель и цикл обучения 348
11.2. Точка входа приложения 352
11.3. Предварительная настройка и инициализация 354
11.3.1. Инициализация модели и оптимизатора 355
11.3.2. Передача данных загрузчикам 356
11.4. Первый сквозной дизайн нейронной сети. 359
11.4.1. Основы свертки 360
11.4.2. Полная модель. 363
11.5. Обучение и проверка модели. 366
11.5.1. Функция calculateBatchLoss 368
11.5.2. Цикл проверки работает аналогично 370
11.6. Вывод метрик производительности 371
11.6.1. Функция logMetrics 372
11.7. Запуск обучения 375
11.7.1. Необходимые для обучения данные 377
11.7.2. Интерлюдия: функция enumerateWithEstimate 378
11.8. Оценка модели: 99,7 % правильных ответов — это отличный результат, не так ли? 379
11.9. Построение графиков для метрик обучения с помощью TensorBoard 381

11.9.1. Запуск TensorBoard 381
11.9.2. Внедрение TensorBoard в функцию регистрации метрик . . .385
11.10. Почему модель не учится обнаруживать узелки? 387
11.11. Итоги главы. 388
11.12. Упражнения. 389
11.13. Резюме 389

Глава 12. Улучшение процесса обучения с помощью метрик и дополнений 391
12.1. План модернизации 392
12.2. Хорошие собаки против плохих парней: ложноположительные и ложноотрицательные результаты 393
12.3. Визуализация положительных и отрицательных результатов 395
12.3.1. Высокий отклик Рокси 398
12.3.2. Высокая точность Престона 400
12.3.3. Реализация точности и отклика в logMetrics 401
12.3.4. Готовая метрика производительности: метрика F1 402
12.3.5. Как модель работает с новыми метриками 407
12.4. Как выглядит идеальный набор данных. 408
12.4.1. Как сделать данные более «идеальными» 411
12.4.2. Сравнение результатов обучения по сбалансированному и несбалансированному набору 417
12.4.3. Распознавание симптомов переобучения 419
12.5. Вернемся к проблеме переобучения 421
12.5.1. Модель прогнозирования с переобучением по возрасту 421
12.6. Предотвращение переобучения путем увеличения набора данных 422
12.6.1. Методы дополнения данных 423
12.6.2. Наблюдение за улучшением данных после дополнения 429
12.7. Итоги главы 431
12.8. Упражнения 432
12.9. Резюме. 433

Глава 13. Поиск потенциальных узелков с помощью сегментации 435
13.1. Добавим в проект вторую модель 436
13.2. Различные типы сегментации 438

13.3. Семантическая сегментация: попиксельная классификация 439
13.3.1. Архитектура U-Net 443
13.4. Обновление модели сегментации 445
13.4.1. Адаптация готовой модели к нашему проекту 447
13.5. Модификация набора данных для сегментации 449
13.5.1. Особые требования U-Net к размеру входных данных 450
13.5.2. Компромиссы U-Net при работе с 3D- и 2D-данными 450
13.5.3. Формирование достоверных данных 452
13.5.4. Реализация Luna2dSegmentationDataset 459
13.5.5. Разработка наших данных для обучения и проверки 464
13.5.6. Реализация набора данных TrainingLuna2dSegmentation . . .465
13.5.7. Дополнение данных на ГП 466
13.6. Внедрение сегментации в сценарий обучения 469
13.6.1. Инициализация наших моделей сегментации и увеличения 470
13.6.2. Использование оптимизатора Adam 471
13.6.3. Потеря Дайса 471
13.6.4. Получение изображений в TensorBoard 475
13.6.5. Обновление логирования метрик 479
13.6.6. Сохранение модели 481
13.7. Результаты 482
13.8. Итоги главы 485
13.9. Упражнения 486
13.10. Резюме 487

Глава 14. Сквозной анализ узелков и дальнейшее развитие проекта 488
14.1. Финишная прямая 488
14.2. Независимость проверочного набора 492
14.3. Объединение сегментации КТ и классификации узелков-кандидатов 493
14.3.1. Сегментация 494
14.3.2. Группировка вокселей в узелки-кандидаты 495
14.3.3. Узелок или не узелок? Классификация и снижение числа ложноположительных результатов 497
14.4. Количественная оценка 502

14.5. Прогнозирование злокачественности 503
14.5.1. Получение информации о злокачественных новообразованиях 503
14.5.2. Базовый уровень для вычисления площади под кривой: классификация по диаметру 504
14.5.3. Повторное использование весов: тонкая настройка. 508
14.5.4. Больше данных в TensorBoard 515
14.6. Каков диагноз? 518
14.6.1. Наборы для обучения, проверки и тестирования 520
14.7. Что дальше? Дополнительные источники вдохновения (и данных). 521
14.7.1. Борьба с переобучением: выбор лучшей регуляризации. . . .522
14.7.2. Подготовка обучающих данных 525
14.7.3. Итоги конкурса и научные работы 527
14.8. Итоги главы 528
14.8.1. За кулисами 529
14.9. Упражнения 530
14.10. Резюме 531

ЧАСТЬ III
Развертывание

Глава 15. Развертывание в производстве 534
15.1. Поставка моделей PyTorch 535
15.1.1. Размещение модели на сервере Flask. 536
15.1.2. Требования к развертыванию 538
15.1.3. Пакетная обработка запросов 539
15.2. Экспорт модели 545
15.2.1. Совместимость за пределами PyTorch с ONNX 546
15.2.2. Встроенный механизм экспорта PyTorch: отслеживание . . .547
15.2.3. Сервер с отслеженной моделью 549
15.3. Взаимодействие с PyTorch JIT. 549
15.3.1. Что за пределами Python/PyTorch 550
15.3.2. Двойственная природа PyTorch как интерфейса и бекэнда . .552
15.3.3. TorchScript 552

15.3.4. Использование сценариев как лучшей замены отслеживания 556
15.4. LibTorch: PyTorch в C++ 557
15.4.1. Запуск JIT-моделей из C++ 558
15.4.2. Сразу работаем с C++ и API C++ 561
15.5. Добавим мобильности 565
15.5.1. Повышение эффективности: проектирование моделей и квантование 569
15.6. Новые технологии: корпоративная поставка моделей PyTorch . . . 571
15.7. Итоги главы 571
15.8. Упражнение 572
15.9. Резюме 572