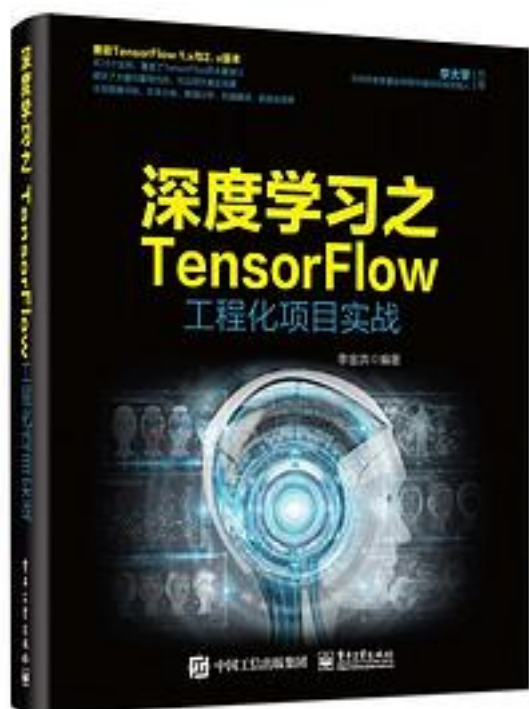


# 深度学习之TensorFlow工程化项目实战



[深度学习之TensorFlow工程化项目实战\\_下载链接1](#)

著者:李金洪

出版者:电子工业出版社

出版时间:2019-5

装帧:平装

isbn:9787121363924

《深度学习之TensorFlow工程化项目实战》是一本非常全面的、专注于实战的AI图书，兼容TensorFlow 1.x和2.x版本，共75个实例。

《深度学习之TensorFlow工程化项目实战》共分为5篇：第1篇，介绍了学习准备、搭建开发环境、使用AI模型来识别图像；第2篇，介绍了用TensorFlow开发实际工程的一些基础操作，包括使用TensorFlow制作自己的数据集、快速训练自己的图片分类模型、编写训练模型的程序；第3篇，介绍了机器学习算法相关内容，包括特征工程、卷积神经网络（CNN）、循环神经网络（RNN）；第4篇，介绍了多模型的组合训练技术，包括生成式模型、模型的攻与防；第5篇，介绍了深度学习在工程上的应用，侧重于提

升读者的工程能力，包括TensorFlow模型制作、部署TensorFlow模型、商业实例。

本书结构清晰、案例丰富、通俗易懂、实用性强。适合对人工智能、TensorFlow感兴趣的读者作为自学教程。另外，本书也适合社会培训学校作为培训教材，还适合大中专院校的相关专业作为教学参考书。

作者介绍:

本书由李金洪主笔编写，参与本书编写的还有以下作者。

石昌帅

代码医生工作室成员，具有丰富的嵌入式及算法开发经验，参与多款机器人、图像识别等项目开发，擅长机器人定位、导航技术、计算机视觉技术，熟悉NVIDIA Jetson系列、Raspberry Pi系列等平台软硬件开发、算法优化。从事的技术方向包括机器人导航、图像处理、自动驾驶等。

甘月

代码医生工作室成员，资深iOS高级工程师，有丰富的iOS研发经验，先后担任iOS主管、项目经理、iOS技术总监等职务，精通Objective-C、Swift、C等编程语言，参与过银行金融、娱乐机器人、婚庆、医疗等领域的多个项目。擅长Mac系统下的AI技术开发。

江泉宇

代码医生工作室成员，是大蛇智能社区成长最快的AI学者。半年时间，由普通读者升级为社区的资深辅导员。在校期间曾参加过电子设计大赛（获省级一等奖）、Google校企合作的AI创新项目、省级创新训练AI项目。熟悉Python、C和Java等编程语言。擅长图像处理方向、特征工程方向及语义压缩方向的AI任务。

目录: 第1篇 准备

第1章 学习准备 2

1.1 TensorFlow能做什么 2

1.2 学习TensorFlow的必备知识 3

1.3 学习技巧：跟读代码 4

1.4 如何学习本书 4

第2章 搭建开发环境 5

2.1 准备硬件环境 5

2.2 下载及安装Anaconda 6

2.3 安装TensorFlow 9

2.4 GPU版本的安装方法 10

2.4.1 在Windows中安装CUDA 10

2.4.2 在Linux中安装CUDA 13

2.4.3 在Windows中安装cuDNN 13

2.4.4 在Linux中安装cuDNN 14

2.4.5 常见错误及解决方案 16

2.5 测试显卡的常用命令 16

2.6 TensorFlow 1.x版本与2.x版本共存的解决方案 18

第3章 实例1：用AI模型识别图像是桌子、猫、狗，还是其他 21

3.1 准备代码环境并预训练模型 21

3.2 代码实现：初始化环境变量，并载入ImgNet标签 24

- 3.3 代码实现：定义网络结构 25
- 3.4 代码实现：载入模型进行识别 26
- 3.5 扩展：用更多预训练模型完成图片分类任务 28
- 第2篇 基础
- 第4章 用TensorFlow制作自己的数据集 30
- 4.1 快速导读 30
  - 4.1.1 什么是数据集 30
  - 4.1.2 TensorFlow的框架 31
  - 4.1.3 什么是TFDS 31
- 4.2 实例2：将模拟数据制作成内存对象数据集 32
  - 4.2.1 代码实现：生成模拟数据 32
  - 4.2.2 代码实现：定义占位符 33
  - 4.2.3 代码实现：建立会话，并获取数据 34
  - 4.2.4 代码实现：将模拟数据可视化 34
  - 4.2.5 运行程序 34
  - 4.2.6 代码实现：创建带有迭代值并支持乱序功能的模拟数据集 35
- 4.3 实例3：将图片制作成内存对象数据集 37
  - 4.3.1 样本介绍 38
  - 4.3.2 代码实现：载入文件名称与标签 39
  - 4.3.3 代码实现：生成队列中的批次样本数据 40
  - 4.3.4 代码实现：在会话中使用数据集 41
  - 4.3.5 运行程序 42
- 4.4 实例4：将Excel文件制作成内存对象数据集 42
  - 4.4.1 样本介绍 43
  - 4.4.2 代码实现：逐行读取数据并分离标签 43
  - 4.4.3 代码实现：生成队列中的批次样本数据 44
  - 4.4.4 代码实现：在会话中使用数据集 45
  - 4.4.5 运行程序 46
- 4.5 实例5：将图片文件制作成TFRecord数据集 46
  - 4.5.1 样本介绍 47
  - 4.5.2 代码实现：读取样本文件的目录及标签 47
  - 4.5.3 代码实现：定义函数生成TFRecord数据集 48
  - 4.5.4 代码实现：读取TFRecord数据集，并将其转化为队列 49
  - 4.5.5 代码实现：建立会话，将数据保存到文件 50
  - 4.5.6 运行程序 51
- 4.6 实例6：将内存对象制作成Dataset数据集 52
  - 4.6.1 如何生成Dataset数据集 52
  - 4.6.2 如何使用Dataset接口 53
  - 4.6.3 tf.data.Dataset接口所支持的数据集变换操作 54
  - 4.6.4 代码实现：以元组和字典的方式生成Dataset对象 58
  - 4.6.5 代码实现：对Dataset对象中的样本进行变换操作 59
  - 4.6.6 代码实现：创建Dataset迭代器 60
  - 4.6.7 代码实现：在会话中取出数据 60
  - 4.6.8 运行程序 61
  - 4.6.9 使用tf.data.Dataset.from\_tensor\_slices接口的注意事项 62
- 4.7 实例7：将图片文件制作成Dataset数据集 63
  - 4.7.1 代码实现：读取样本文件的目录及标签 64
  - 4.7.2 代码实现：定义函数，实现图片转换操作 64
  - 4.7.3 代码实现：用自定义函数实现图片归一化 65
  - 4.7.4 代码实现：用第三方函数将图片旋转30° 65
  - 4.7.5 代码实现：定义函数，生成Dataset对象 66
  - 4.7.6 代码实现：建立会话，输出数据 67
  - 4.7.7 运行程序 68
- 4.8 实例8：将TFRecord文件制作成Dataset数据集 69

- 4.8.1 样本介绍 69
- 4.8.2 代码实现：定义函数，生成Dataset对象 70
- 4.8.3 代码实现：建立会话输出数据 71
- 4.8.4 运行程序 72
- 4.9 实例9：在动态图中读取Dataset数据集 72
  - 4.9.1 代码实现：添加动态图调用 72
  - 4.9.2 制作数据集 73
  - 4.9.3 代码实现：在动态图中显示数据 73
  - 4.9.4 实例10：在TensorFlow 2.x中操作数据集 74
- 4.10 实例11：在不同场景中使用数据集 77
  - 4.10.1 代码实现：在训练场景中使用数据集 78
  - 4.10.2 代码实现：在应用模型场景中使用数据集 79
  - 4.10.3 代码实现：在训练与测试混合场景中使用数据集 80
- 4.11 tf.data.Dataset接口的更多应用 81
- 第5章 10分钟快速训练自己的图片分类模型 82
  - 5.1 快速导读 82
    - 5.1.1 认识模型和模型检查点文件 82
    - 5.1.2 了解“预训练模型”与微调（Fine-Tune） 82
    - 5.1.3 学习TensorFlow中的预训练模型库——TF-Hub库 83
  - 5.2 实例12：通过微调模型分辨男女 83
    - 5.2.1 准备工作 84
    - 5.2.2 代码实现：处理样本数据并生成Dataset对象 85
    - 5.2.3 代码实现：定义微调模型的类MyNASNetModel 88
    - 5.2.4 代码实现：构建MyNASNetModel类中的基本模型 88
    - 5.2.5 代码实现：实现MyNASNetModel类中的微调操作 89
    - 5.2.6 代码实现：实现与训练相关的其他方法 90
    - 5.2.7 代码实现：构建模型，用于训练、测试、使用 92
    - 5.2.8 代码实现：通过二次迭代来训练微调模型 94
    - 5.2.9 代码实现：测试模型 96
  - 5.3 扩展：通过摄像头实时分辨男女 100
  - 5.4 TF-slim接口中的更多成熟模型 100
  - 5.5 实例13：用TF-Hub库微调模型以评估人物的年龄 100
    - 5.5.1 准备样本 101
    - 5.5.2 下载TF-Hub库中的模型 102
    - 5.5.3 代码实现：测试TF-Hub库中的MobileNet\_V2模型 104
    - 5.5.4 用TF-Hub库微调MobileNet\_V2模型 107
    - 5.5.5 代码实现：用模型评估人物的年龄 109
    - 5.5.6 扩展：用TF-Hub库中的其他模型处理不同领域的分类任务 113
  - 5.6 总结 113
  - 5.7 练习题 114
    - 5.7.1 基于TF-slim接口的练习 115
    - 5.7.2 基于TF-Hub库的练习 115
- 第6章 用TensorFlow编写训练模型的程序 117
  - 6.1 快速导读 117
    - 6.1.1 训练模型是怎么回事 117
    - 6.1.2 用“静态图”方式训练模型 117
    - 6.1.3 用“动态图”方式训练模型 118
    - 6.1.4 什么是估算器框架接口（Estimators API） 119
    - 6.1.5 什么是tf.layers接口 120
    - 6.1.6 什么是tf.keras接口 121
    - 6.1.7 什么是tf.js接口 122
    - 6.1.8 什么是TFLearn框架 123
    - 6.1.9 该选择哪种框架 123
    - 6.1.10 分配运算资源与使用分布策略 124

- 6.1.11 用tfdbg调试TensorFlow模型 127
- 6.1.12 用钩子函数 (Training Hooks) 跟踪训练状态 127
- 6.1.13 用分布式运行方式训练模型 128
- 6.1.14 用T2T框架系统更方便地训练模型 128
- 6.1.15 将TensorFlow 1.x中的代码移植到2.x版本 129
- 6.1.16 TensorFlow 2.x中的新特性——自动图 130
- 6.2 实例14：用静态图训练一个具有保存检查点功能的回归模型 131
  - 6.2.1 准备开发步骤 131
  - 6.2.2 生成检查点文件 131
  - 6.2.3 载入检查点文件 132
  - 6.2.4 代码实现：在线性回归模型中加入保存检查点功能 132
  - 6.2.5 修改迭代次数，二次训练 135
- 6.3 实例15：用动态图 (eager) 训练一个具有保存检查点功能的回归模型 136
  - 6.3.1 代码实现：启动动态图，生成模拟数据 136
  - 6.3.2 代码实现：定义动态图的神经网络结构 137
  - 6.3.3 代码实现：在动态图中加入保存检查点功能 138
  - 6.3.4 代码实现：按指定迭代次数进行训练，并可视化结果 139
  - 6.3.5 运行程序，显示结果 140
  - 6.3.6 代码实现：用另一种方法计算动态图梯度 141
  - 6.3.7 实例16：在动态图中获取参数变量 142
  - 6.3.8 小心动态图中的参数陷阱 144
  - 6.3.9 实例17：在静态图中使用动态图 145
- 6.4 实例18：用估算器框架训练一个回归模型 147
  - 6.4.1 代码实现：生成样本数据集 147
  - 6.4.2 代码实现：设置日志级别 148
  - 6.4.3 代码实现：实现估算器的输入函数 148
  - 6.4.4 代码实现：定义估算器的模型函数 149
  - 6.4.5 代码实现：通过创建config文件指定硬件的运算资源 151
  - 6.4.6 代码实现：定义估算器 152
  - 6.4.7 用tf.estimator.RunConfig控制更多的训练细节 153
  - 6.4.8 代码实现：用估算器训练模型 153
  - 6.4.9 代码实现：通过热启动实现模型微调 155
  - 6.4.10 代码实现：测试估算器模型 158
  - 6.4.11 代码实现：使用估算器模型 158
  - 6.4.12 实例19：为估算器添加日志钩子函数 159
- 6.5 实例20：将估算器代码改写成静态图代码 161
  - 6.5.1 代码实现：复制网络结构 161
  - 6.5.2 代码实现：重用输入函数 163
  - 6.5.3 代码实现：创建会话恢复模型 163
  - 6.5.4 代码实现：继续训练 163
- 6.6 实例21：用tf.layers API在动态图上识别手写数字 165
  - 6.6.1 代码实现：启动动态图并加载手写图片数据集 165
  - 6.6.2 代码实现：定义模型的类 166
  - 6.6.3 代码实现：定义网络的反向传播 167
  - 6.6.4 代码实现：训练模型 167
- 6.7 实例22：用tf.keras API训练一个回归模型 168
  - 6.7.1 代码实现：用model类搭建模型 168
  - 6.7.2 代码实现：用sequential类搭建模型 169
  - 6.7.3 代码实现：搭建反向传播的模型 171
  - 6.7.4 代码实现：用两种方法训练模型 172
  - 6.7.5 代码实现：获取模型参数 172
  - 6.7.6 代码实现：测试模型与用模型进行预测 173
  - 6.7.7 代码实现：保存模型与加载模型 173
  - 6.7.8 代码实现：将模型导出成JSON文件，再将JSON文件导入模型 175

- 6.7.9 实例23：在tf.keras接口中使用预训练模型ResNet 176
- 6.7.10 扩展：在动态图中使用tf.keras接口 178
- 6.7.11 实例24：在静态图中使用tf.keras接口 178
- 6.8 实例25：用tf.js接口后方训练一个回归模型 180
  - 6.8.1 代码实现：在HTTP的头标签中添加tfjs模块 180
  - 6.8.2 代码实现：用JavaScript脚本实现回归模型 181
  - 6.8.3 运行程序：在浏览器中查看效果 181
  - 6.8.4 扩展：tf.js 接口中的应用场景 182
- 6.9 实例26：用估算器框架实现分布式部署训练 182
  - 6.9.1 运行程序：修改估算器模型，使其支持分布式 182
  - 6.9.2 通过TF\_CONFIG进行分布式配置 183
  - 6.9.3 运行程序 185
  - 6.9.4 扩展：用分布策略或KubeFlow框架进行分布式部署 186
- 6.10 实例27：在分布式估算器框架中用tf.keras接口训练ResNet模型，识别图片中是橘子还是苹果 186
  - 6.10.1 样本准备 186
  - 6.10.2 代码实现：准备训练与测试数据集 187
  - 6.10.3 代码实现：制作模型输入函数 187
  - 6.10.4 代码实现：搭建ResNet模型 188
  - 6.10.5 代码实现：训练分类器模型 189
  - 6.10.6 运行程序：评估模型 190
  - 6.10.7 扩展：全连接网络的优化 190
- 6.11 实例28：在T2T框架中用tf.layers接口实现MNIST数据集分类 191
  - 6.11.1 代码实现：查看T2T框架中的数据集（problems） 191
  - 6.11.2 代码实现：构建T2T框架的工作路径及下载数据集 192
  - 6.11.3 代码实现：在T2T框架中搭建自定义卷积网络模型 193
  - 6.11.4 代码实现：用动态图方式训练自定义模型 194
  - 6.11.5 代码实现：在动态图中用metrics模块评估模型 195
- 6.12 实例29：在T2T框架中，用自定义数据集训练中英文翻译模型 196
  - 6.12.1 代码实现：声明自己的problems数据集 196
  - 6.12.2 代码实现：定义自己的problems数据集 197
  - 6.12.3 在命令行下生成TFrecoder格式的数据 198
  - 6.12.4 查找T2T框架中的模型及超参，并用指定的模型及超参进行训练 199
  - 6.12.5 用训练好的T2T框架模型进行预测 201
  - 6.12.6 扩展：在T2T框架中，如何选取合适的模型及超参 202
- 6.13 实例30：将TensorFlow 1.x中的代码升级为可用于2.x版本的代码 203
  - 6.13.1 准备工作：创建Python虚环境 203
  - 6.13.2 使用工具转换源码 204
  - 6.13.3 修改转换后的代码文件 204
  - 6.13.4 将代码升级到TensorFlow 2.x版本的经验总结 205

第3篇 进阶

第7章 特征工程——会说话的数据 208

- 7.1 快速导读 208
  - 7.1.1 特征工程的基础知识 208
  - 7.1.2 离散数据特征与连续数据特征 209
  - 7.1.3 了解特征列接口 210
  - 7.1.4 了解序列特征列接口 210
  - 7.1.5 了解弱学习器接口——梯度提升树（TFBT接口） 210
  - 7.1.6 了解特征预处理模块（tf.Transform） 211
  - 7.1.7 了解因子分解模块 212
  - 7.1.8 了解加权矩阵分解算法 212
  - 7.1.9 了解Lattice模块——点阵模型 213
  - 7.1.10 联合训练与集成学习 214

- 7.2 实例31：用wide\_deep模型预测人口收入 214
  - 7.2.1 了解人口收入数据集 214
  - 7.2.2 代码实现：探索性数据分析 217
  - 7.2.3 认识wide\_deep模型 218
  - 7.2.4 部署代码文件 219
  - 7.2.5 代码实现：初始化样本常量 220
  - 7.2.6 代码实现：生成特征列 220
  - 7.2.7 代码实现：生成估算器模型 222
  - 7.2.8 代码实现：定义输入函数 223
  - 7.2.9 代码实现：定义用于导出冻结图文件的函数 224
  - 7.2.10 代码实现：定义类，解析启动参数 225
  - 7.2.11 代码实现：训练和测试模型 226
  - 7.2.12 代码实现：使用模型 227
  - 7.2.13 运行程序 228
- 7.3 实例32：用弱学习器中的梯度提升树算法预测人口收入 229
  - 7.3.1 代码实现：为梯度提升树模型准备特征列 230
  - 7.3.2 代码实现：构建梯度提升树模型 230
  - 7.3.3 代码实现：训练并导出梯度提升树模型 231
  - 7.3.4 代码实现：设置启动参数，运行程序 232
  - 7.3.5 扩展：更灵活的TFBT接口 233
- 7.4 实例33：用feature\_column模块转换特征列 233
  - 7.4.1 代码实现：用feature\_column模块处理连续值特征列 234
  - 7.4.2 代码实现：将连续值特征列转化成离散值特征列 237
  - 7.4.3 代码实现：将离散文本特征列转化为one-hot与词向量 239
  - 7.4.4 代码实现：根据特征列生成交叉列 246
- 7.5 实例34：用sequence\_feature\_column接口完成自然语言处理任务的数据预处理工作 248
  - 7.5.1 代码实现：构建模拟数据 248
  - 7.5.2 代码实现：构建词嵌入初始值 249
  - 7.5.3 代码实现：构建词嵌入特征列与共享特征列 249
  - 7.5.4 代码实现：构建序列特征列的输入层 250
  - 7.5.5 代码实现：建立会话输出结果 251
- 7.6 实例35：用factorization模块的kmeans接口聚类COCO数据集中的标注框 253
  - 7.6.1 代码实现：设置要使用的数据集 253
  - 7.6.2 代码实现：准备带聚类的数据样本 253
  - 7.6.3 代码实现：定义聚类模型 255
  - 7.6.4 代码实现：训练模型 256
  - 7.6.5 代码实现：输出图示化结果 256
  - 7.6.6 代码实现：提取并排序聚类结果 258
  - 7.6.7 扩展：聚类与神经网络混合训练 258
- 7.7 实例36：用加权矩阵分解模型实现基于电影评分的推荐系统 259
  - 7.7.1 下载并加载数据集 259
  - 7.7.2 代码实现：根据用户和电影特征列生成稀疏矩阵 260
  - 7.7.3 代码实现：建立WALS模型，并对其进行训练 261
  - 7.7.4 代码实现：评估WALS模型 263
  - 7.7.5 代码实现：用WALS模型为用户推荐电影 264
  - 7.7.6 扩展：使用WALS的估算器接口 265
- 7.8 实例37：用Lattice模块预测人口收入 265
  - 7.8.1 代码实现：读取样本，并创建输入函数 266
  - 7.8.2 代码实现：创建特征列，并保存校准关键点 267
  - 7.8.3 代码实现：创建校准线性模型 270
  - 7.8.4 代码实现：创建校准点阵模型 270
  - 7.8.5 代码实现：创建随机微点阵模型 271

- 7.8.6 代码实现：创建集合的微点阵模型 271
- 7.8.7 代码实现：定义评估与训练函数 272
- 7.8.8 代码实现：训练并评估模型 273
- 7.8.9 扩展：将点阵模型嵌入神经网络中 274
- 7.9 实例38：结合知识图谱实现基于电影的推荐系统 278
  - 7.9.1 准备数据集 278
  - 7.9.2 预处理数据 279
  - 7.9.3 搭建MKR模型 279
  - 7.9.4 训练模型并输出结果 286
- 7.10 可解释性算法的意义 286
- 第8章 卷积神经网络（CNN）——在图像处理中应用最广泛的模型 287
  - 8.1 快速导读 287
    - 8.1.1 认识卷积神经网络 287
    - 8.1.2 什么是空洞卷积 288
    - 8.1.3 什么是深度卷积 290
    - 8.1.4 什么是深度可分离卷积 290
    - 8.1.5 了解卷积网络的缺陷及补救方法 291
    - 8.1.6 了解胶囊神经网络与动态路由 292
    - 8.1.7 了解矩阵胶囊网络与EM路由算法 297
    - 8.1.8 什么是NLP任务 298
    - 8.1.9 了解多头注意力机制与内部注意力机制 298
    - 8.1.10 什么是带有位置向量的词嵌入 300
    - 8.1.11 什么是目标检测任务 300
    - 8.1.12 什么是目标检测中的上采样与下采样 301
    - 8.1.13 什么是图片分割任务 301
  - 8.2 实例39：用胶囊网络识别黑白图中服装的图案 302
    - 8.2.1 熟悉样本：了解Fashion-MNIST数据集 302
    - 8.2.2 下载Fashion-MNIST数据集 303
    - 8.2.3 代码实现：读取及显示Fashion-MNIST数据集中的数据 304
    - 8.2.4 代码实现：定义胶囊网络模型类CapsuleNetModel 305
    - 8.2.5 代码实现：实现胶囊网络的基本结构 306
    - 8.2.6 代码实现：构建胶囊网络模型 309
    - 8.2.7 代码实现：载入数据集，并训练胶囊网络模型 310
    - 8.2.8 代码实现：建立会话训练模型 311
    - 8.2.9 运行程序 313
    - 8.2.10 实例40：实现带有EM路由的胶囊网络 314
  - 8.3 实例41：用TextCNN模型分析评论者是否满意 322
    - 8.3.1 熟悉样本：了解电影评论数据集 322
    - 8.3.2 熟悉模型：了解TextCNN模型 322
    - 8.3.3 数据预处理：用preprocessing接口制作字典 323
    - 8.3.4 代码实现：生成NLP文本数据集 326
    - 8.3.5 代码实现：定义TextCNN模型 327
    - 8.3.6 代码实现：训练TextCNN模型 330
    - 8.3.7 运行程序 332
    - 8.3.8 扩展：提升模型精度的其他方法 333
  - 8.4 实例42：用带注意力机制的模型分析评论者是否满意 333
    - 8.4.1 熟悉样本：了解tf.keras接口中的电影评论数据集 333
    - 8.4.2 代码实现：将tf.keras接口中的IMDB数据集还原成句子 334
    - 8.4.3 代码实现：用tf.keras接口开发带有位置向量的词嵌入层 336
    - 8.4.4 代码实现：用tf.keras接口开发注意力层 338
    - 8.4.5 代码实现：用tf.keras接口训练模型 340
    - 8.4.6 运行程序 341
    - 8.4.7 扩展：用Targeted Dropout技术进一步提升模型的性能 342
  - 8.5 实例43：搭建YOLO V3模型，识别图片中的酒杯、水果等物体 343



- 8.5.1 YOLO V3模型的样本与结构 343
- 8.5.2 代码实现：Darknet-53 模型的darknet块 344
- 8.5.3 代码实现：Darknet-53 模型的下采样卷积 345
- 8.5.4 代码实现：搭建Darknet-53模型，并返回3种尺度特征值 345
- 8.5.5 代码实现：定义YOLO检测模块的参数及候选框 346
- 8.5.6 代码实现：定义YOLO检测块，进行多尺度特征融合 347
- 8.5.7 代码实现：将YOLO检测块的特征转化为bbox attrs单元 347
- 8.5.8 代码实现：实现YOLO V3的检测部分 349
- 8.5.9 代码实现：用非极大值抑制算法对检测结果去重 352
- 8.5.10 代码实现：载入预训练权重 355
- 8.5.11 代码实现：载入图片，进行目标实物的识别 356
- 8.5.12 运行程序 358
- 8.6 实例44：用YOLO V3模型识别门牌号 359
  - 8.6.1 工程部署：准备样本 359
  - 8.6.2 代码实现：读取样本数据，并制作标签 359
  - 8.6.3 代码实现：用tf.keras接口构建YOLO V3模型，并计算损失 364
  - 8.6.4 代码实现：在动态图中训练模型 368
  - 8.6.5 代码实现：用模型识别门牌号 372
  - 8.6.6 扩展：标注自己的样本 374
- 8.7 实例45：用Mask R-CNN模型定位物体的像素点 375
  - 8.7.1 下载COCO数据集及安装pycocotools 376
  - 8.7.2 代码实现：验证pycocotools及读取COCO数据集 377
  - 8.7.3 拆分Mask R-CNN模型的处理步骤 383
  - 8.7.4 工程部署：准备代码文件及模型 385
  - 8.7.5 代码实现：加载数据构建模型，并输出模型权重 385
  - 8.7.6 代码实现：搭建残差网络ResNet 387
  - 8.7.7 代码实现：搭建Mask R-CNN模型的骨干网络ResNet 393
  - 8.7.8 代码实现：可视化Mask R-CNN模型骨干网络的特征输出 396
  - 8.7.9 代码实现：用特征金字塔网络处理骨干网络特征 400
  - 8.7.10 计算RPN中的锚点 402
  - 8.7.11 代码实现：构建RPN 403
  - 8.7.12 代码实现：用非极大值抑制算法处理RPN的结果 405
  - 8.7.13 代码实现：提取RPN的检测结果 410
  - 8.7.14 代码实现：可视化RPN的检测结果 412
  - 8.7.15 代码实现：在MaskRCNN类中对ROI区域进行分类 415
  - 8.7.16 代码实现：金字塔网络的区域对齐层（ROIAlign）中的区域框与特征的匹配算法 416
  - 8.7.17 代码实现：在金字塔网络的ROIAlign层中按区域边框提取内容 418
  - 8.7.18 代码实现：调试并输出ROIAlign层的内部运算值 421
  - 8.7.19 代码实现：对ROI内容进行分类 422
  - 8.7.20 代码实现：用检测器DetectionLayer检测ROI内容，得到最终的实物矩形 426
  - 8.7.21 代码实现：根据ROI内容进行实物像素分割 432
  - 8.7.22 代码实现：用Mask R-CNN模型分析图片 436
- 8.8 实例46：训练Mask R-CNN模型，进行形状的识别 439
  - 8.8.1 工程部署：准备代码文件及模型 440
  - 8.8.2 样本准备：生成随机形状图片 440
  - 8.8.3 代码实现：为Mask R-CNN模型添加损失函数 442
  - 8.8.4 代码实现：为Mask R-CNN模型添加训练函数，使其支持微调与全网训练 444
  - 8.8.5 代码实现：训练并使用模型 446
  - 8.8.6 扩展：替换特征提取网络 449
- 第9章 循环神经网络（RNN）——处理序列样本的神经网络 450
  - 9.1 快速导读 450
    - 9.1.1 什么是循环神经网络 450
    - 9.1.2 了解RNN模型的基础单元LSTM与GRU 451

- 9.1.3 认识QRNN单元 451
- 9.1.4 认识SRU单元 451
- 9.1.5 认识IndRNN单元 452
- 9.1.6 认识JANET单元 453
- 9.1.7 优化RNN模型的技巧 453
- 9.1.8 了解RNN模型中多项式分布的应用 453
- 9.1.9 了解注意力机制的Seq2Seq框架 454
- 9.1.10 了解BahdanauAttention与LuongAttention 456
- 9.1.11 了解单调注意力机制 457
- 9.1.12 了解混合注意力机制 458
- 9.1.13 了解Seq2Seq接口中的采样接口 (Helper) 460
- 9.1.14 了解RNN模型的Wrapper接口 460
- 9.1.15 什么是时间序列 (TFTS) 框架 461
- 9.1.16 什么是梅尔标度 461
- 9.1.17 什么是短时傅立叶变换 462
- 9.2 实例47: 搭建RNN模型, 为女孩生成英文名字 463
  - 9.2.1 代码实现: 读取及处理样本 463
  - 9.2.2 代码实现: 构建Dataset数据集 466
  - 9.2.3 代码实现: 用tf.keras接口构建生成式RNN模型 467
  - 9.2.4 代码实现: 在动态图中训练模型 468
  - 9.2.5 代码实现: 载入检查点文件并用模型生成名字 469
  - 9.2.6 扩展: 用RNN模型编写文章 471
- 9.3 实例48: 用带注意力机制的Seq2Seq模型为图片添加内容描述 471
  - 9.3.1 设计基于图片的Seq2Seq 471
  - 9.3.2 代码实现: 图片预处理——用ResNet提取图片特征并保存 472
  - 9.3.3 代码实现: 文本预处理——过滤处理、字典建立、对齐与向量化处理 475
  - 9.3.4 代码实现: 创建数据集 477
  - 9.3.5 代码实现: 用tf.keras接口构建Seq2Seq模型中的编码器 477
  - 9.3.6 代码实现: 用tf.keras接口构建Bahdanau类型的注意力机制 478
  - 9.3.7 代码实现: 搭建Seq2Seq模型中的解码器Decoder 478
  - 9.3.8 代码实现: 在动态图中计算Seq2Seq模型的梯度 480
  - 9.3.9 代码实现: 在动态图中为Seq2Seq模型添加保存检查点功能 480
  - 9.3.10 代码实现: 在动态图中训练Seq2Seq模型 481
  - 9.3.11 代码实现: 用多项式分布采样获取图片的内容描述 482
- 9.4 实例49: 用IndRNN与IndyLSTM单元制作聊天机器人 485
  - 9.4.1 下载及处理样本 486
  - 9.4.2 代码实现: 读取样本, 分词并创建字典 487
  - 9.4.3 代码实现: 对样本进行向量化、对齐、填充预处理 489
  - 9.4.4 代码实现: 在Seq2Seq模型中加工样本 489
  - 9.4.5 代码实现: 在Seq2Seq模型中, 实现基于IndRNN与IndyLSTM的动态多层RNN编码器 491
  - 9.4.6 代码实现: 为Seq2Seq模型中的解码器创建Helper 491
  - 9.4.7 代码实现: 实现带有Bahdanau注意力、dropout、OutputProjectionWrapper的解码器 492
  - 9.4.8 代码实现: 在Seq2Seq模型中实现反向优化 493
  - 9.4.9 代码实现: 创建带有钩子函数的估算器, 并进行训练 494
  - 9.4.10 代码实现: 用估算器框架评估模型 496
  - 9.4.11 扩展: 用注意力机制的Seq2Seq模型实现中英翻译 498
- 9.5 实例50: 预测飞机发动机的剩余使用寿命 498
  - 9.5.1 准备样本 499
  - 9.5.2 代码实现: 预处理数据——制作数据集的输入样本与标签 500
  - 9.5.3 代码实现: 构建带有JANET单元的多层动态RNN模型 504
  - 9.5.4 代码实现: 训练并测试模型 505

- 9.5.5 运行程序 507
- 9.5.6 扩展：为含有JANET单元的RNN模型添加注意力机制 508
- 9.6 实例51：将动态路由用于RNN模型，对路透社新闻进行分类 509
  - 9.6.1 准备样本 509
  - 9.6.2 代码实现：预处理数据——对齐序列数据并计算长度 510
  - 9.6.3 代码实现：定义数据集 510
  - 9.6.4 代码实现：用动态路由算法聚合信息 511
  - 9.6.5 代码实现：用IndyLSTM单元搭建RNN模型 513
  - 9.6.6 代码实现：建立会话，训练网络 514
  - 9.6.7 扩展：用分级网络将文章（长文本数据）分类 515
- 9.7 实例52：用TFTS框架预测某地区每天的出生人数 515
  - 9.7.1 准备样本 515
  - 9.7.2 代码实现：数据预处理——制作TFTS框架中的读取器 515
  - 9.7.3 代码实现：用TFTS框架定义模型，并进行训练 516
  - 9.7.4 代码实现：用TFTS框架评估模型 517
  - 9.7.5 代码实现：用模型进行预测，并将结果可视化 517
  - 9.7.6 运行程序 518
  - 9.7.7 扩展：用TFTS框架进行异常值检测 519
- 9.8 实例53：用Tacotron模型合成中文语音（TTS） 520
  - 9.8.1 准备安装包及样本数据 520
  - 9.8.2 代码实现：将音频数据分帧并转为梅尔频谱 521
  - 9.8.3 代码实现：用多进程预处理样本并保存结果 523
  - 9.8.4 拆分Tacotron网络模型的结构 525
  - 9.8.5 代码实现：搭建CBHG网络 527
  - 9.8.6 代码实现：构建带有混合注意力机制的模块 529
  - 9.8.7 代码实现：构建自定义wrapper 531
  - 9.8.8 代码实现：构建自定义采样器 534
  - 9.8.9 代码实现：构建自定义解码器 537
  - 9.8.10 代码实现：构建输入数据集 539
  - 9.8.11 代码实现：构建Tacotron网络 542
  - 9.8.12 代码实现：构建Tacotron网络模型的训练部分 545
  - 9.8.13 代码实现：训练模型并合成音频文件 546
  - 9.8.14 扩展：用pypinyin模块实现文字到声音的转换 551

## 第4篇 高级

## 第10章 生成式模型——能够输出内容的模型 554

- 10.1 快速导读 554
  - 10.1.1 什么是自编码网络模型 554
  - 10.1.2 什么是对抗神经网络模型 554
  - 10.1.3 自编码网络模型与对抗神经网络模型的关系 555
  - 10.1.4 什么是批量归一化中的自适应模式 555
  - 10.1.5 什么是实例归一化 556
  - 10.1.6 了解SwitchableNorm及更多的归一化方法 556
  - 10.1.7 什么是图像风格转换任务 557
  - 10.1.8 什么是人脸属性编辑任务 558
  - 10.1.9 什么是TFgan框架 558
- 10.2 实例54：构建DeblurGAN模型，将模糊相片变清晰 559
  - 10.2.1 获取样本 559
  - 10.2.2 准备SwitchableNorm算法模块 560
  - 10.2.3 代码实现：构建DeblurGAN中的生成器模型 560
  - 10.2.4 代码实现：构建DeblurGAN中的判别器模型 562
  - 10.2.5 代码实现：搭建DeblurGAN的完整结构 563
  - 10.2.6 代码实现：引入库文件，定义模型参数 563
  - 10.2.7 代码实现：定义数据集，构建正反向模型 564
  - 10.2.8 代码实现：计算特征空间损失，并将其编译到生成器模型的训练模型中 566

- 10.2.9 代码实现：按指定次数训练模型 568
- 10.2.10 代码实现：用模型将模糊相片变清晰 569
- 10.2.11 练习题 572
- 10.2.12 扩展：DeblurGAN模型的更多妙用 572
- 10.3 实例55：构建AttGAN模型，对照片进行加胡子、加头帘、加眼镜、变年轻等修改 573
  - 10.3.1 获取样本 573
  - 10.3.2 了解AttGAN模型的结构 574
  - 10.3.3 代码实现：实现支持动态图和静态图的数据集工具类 575
  - 10.3.4 代码实现：将CelebA做成数据集 577
  - 10.3.5 代码实现：构建AttGAN模型的编码器 581
  - 10.3.6 代码实现：构建含有转置卷积的解码器模型 582
  - 10.3.7 代码实现：构建AttGAN模型的判别器模型部分 584
  - 10.3.8 代码实现：定义模型参数，并构建AttGAN模型 585
  - 10.3.9 代码实现：定义训练参数，搭建正反向模型 587
  - 10.3.10 代码实现：训练模型 592
  - 10.3.11 实例56：为人脸添加不同的眼镜 595
  - 10.3.12 扩展：AttGAN模型的局限性 597
- 10.4 实例57：用RNN.WGAN模型模拟生成恶意请求 597
  - 10.4.1 获取样本：通过Panabit设备获取恶意请求样本 597
  - 10.4.2 了解RNN.WGAN模型 600
  - 10.4.3 代码实现：构建RNN.WGAN模型 601
  - 10.4.4 代码实现：训练指定长度的RNN.WGAN模型 607
  - 10.4.5 代码实现：用长度依次递增的方式训练模型 612
  - 10.4.6 运行代码 613
  - 10.4.7 扩展：模型的使用及优化 614
- 第11章 模型的攻与防——看似智能的AI也有脆弱的一面 616
  - 11.1 快速导读 616
    - 11.1.1 什么是FGSM方法 616
    - 11.1.2 什么是cleverhans模块 616
    - 11.1.3 什么是黑箱攻击 617
    - 11.1.4 什么是基于雅可比矩阵的数据增强方法 618
    - 11.1.5 什么是数据中毒攻击 620
  - 11.2 实例58：用FGSM方法生成样本，并攻击PNASNet模型，让其将“狗”识别成“盘子” 621
    - 11.2.1 代码实现：创建PNASNet模型 621
    - 11.2.2 代码实现：搭建输入层并载入图片，复现PNASNet模型的预测效果 623
    - 11.2.3 代码实现：调整参数，定义图片的变化范围 624
    - 11.2.4 代码实现：用梯度下降方式生成对抗样本 625
    - 11.2.5 代码实现：用生成的样本攻击模型 626
    - 11.2.6 扩展：如何防范攻击模型的行为 627
    - 11.2.7 代码实现：将数据增强方式使用在使用场景，以加固PNASNet模型，防范攻击 627
  - 11.3 实例59：击破数据增强防护，制作抗旋转对抗样本 629
    - 11.3.1 代码实现：对输入的数据进行多次旋转 629
    - 11.3.2 代码实现：生成并保存鲁棒性更好的对抗样本 630
    - 11.3.3 代码实现：在PNASNet模型中比较对抗样本的效果 631
  - 11.4 实例60：以黑箱方式攻击未知模型 633
    - 11.4.1 准备工程代码 633
    - 11.4.2 代码实现：搭建通用模型框架 634
    - 11.4.3 代码实现：搭建被攻击模型 637
    - 11.4.4 代码实现：训练被攻击模型 638
    - 11.4.5 代码实现：搭建替代模型 639
    - 11.4.6 代码实现：训练替代模型 639

- 11.4.7 代码实现：黑箱攻击目标模型 641
- 11.4.8 扩展：利用黑箱攻击中的对抗样本加固模型 645
- 第5篇 实战——深度学习实际应用
- 第12章 TensorFlow模型制作——一种功能，多种身份 648
- 12.1 快速导读 648
  - 12.1.1 详细分析检查点文件 648
  - 12.1.2 什么是模型中的冻结图 649
  - 12.1.3 什么是TF Serving模块与saved\_model模块 649
  - 12.1.4 用编译子图（defun）提升动态图的执行效率 649
  - 12.1.5 什么是TF\_Lite模块 652
  - 12.1.6 什么是TFjs-converter模块 653
- 12.2 实例61：在源码与检查点文件分离的情况下，对模型进行二次训练 653
  - 12.2.1 代码实现：在线性回归模型中，向检查点文件中添加指定节点 654
  - 12.2.2 代码实现：在脱离源码的情况下，用检查点文件进行二次训练 657
  - 12.2.3 扩展：更通用的二次训练方法 659
- 12.3 实例62：导出/导入冻结图文件 661
  - 12.3.1 熟悉TensorFlow中的freeze\_graph工具脚本 661
  - 12.3.2 代码实现：从线性回归模型中导出冻结图文件 662
  - 12.3.3 代码实现：导入冻结图文件，并用模型进行预测 664
- 12.4 实例63：逆向分析冻结图文件 665
  - 12.4.1 使用import\_to\_tensorboard工具 666
  - 12.4.2 用TensorBoard工具查看模型结构 666
- 12.5 实例64：用saved\_model模块导出与导入模型文件 668
  - 12.5.1 代码实现：用saved\_model模块导出模型文件 668
  - 12.5.2 代码实现：用saved\_model模块导入模型文件 669
  - 12.5.3 扩展：用saved\_model模块导出带有签名的模型文件 670
- 12.6 实例65：用saved\_model\_cli工具查看及使用saved\_model模型 672
  - 12.6.1 用show参数查看模型 672
  - 12.6.2 用run参数运行模型 673
  - 12.6.3 扩展：了解scan参数的黑名单机制 674
- 12.7 实例66：用TF-Hub库导入、导出词嵌入模型文件 674
  - 12.7.1 代码实现：模拟生成通用词嵌入模型 674
  - 12.7.2 代码实现：用TF-Hub库导出词嵌入模型 675
  - 12.7.3 代码实现：导出TF-Hub模型 678
  - 12.7.4 代码实现：用TF-Hub库导入并使用词嵌入模型 680
- 第13章 部署TensorFlow模型——模型与项目的深度结合 681
- 13.1 快速导读 681
  - 13.1.1 什么是gRPC服务与HTTP/REST API 681
  - 13.1.2 了解TensorFlow对移动终端的支持 682
  - 13.1.3 了解树莓派上的人工智能 683
- 13.2 实例67：用TF\_Serving部署模型并进行远程使用 684
  - 13.2.1 在Linux系统中安装TF\_Serving 684
  - 13.2.2 在多平台中用Docker安装TF\_Serving 685
  - 13.2.3 编写代码：固定模型的签名信息 686
  - 13.2.4 在Linux中开启TF\_Serving服务 688
  - 13.2.5 编写代码：用gRPC访问远程TF\_Serving服务 689
  - 13.2.6 用HTTP/REST API访问远程TF\_Serving服务 691
  - 13.2.7 扩展：关于TF\_Serving的更多例子 694
- 13.3 实例68：在安卓手机上识别男女 694
  - 13.3.1 准备工程代码 694
  - 13.3.2 微调预训练模型 695
  - 13.3.3 搭建安卓开发环境 698
  - 13.3.4 制作lite模型文件 701
  - 13.3.5 修改分类器代码，并运行App 702

- 13.4 实例69：在iPhone手机上识别男女并进行活体检测 703
  - 13.4.1 搭建iOS开发环境 703
  - 13.4.2 部署工程代码并编译 704
  - 13.4.3 载入Lite模型，实现识别男女功能 706
  - 13.4.4 代码实现：调用摄像头并采集视频流 707
  - 13.4.5 代码实现：提取人脸特征 710
  - 13.4.6 活体检测算法介绍 712
  - 13.4.7 代码实现：实现活体检测算法 713
  - 13.4.8 代码实现：完成整体功能并运行程序 714
- 13.5 实例70：在树莓派上搭建一个目标检测器 717
  - 13.5.1 安装树莓派系统 718
  - 13.5.2 在树莓派上安装TensorFlow 721
  - 13.5.3 编译并安装Protobuf 725
  - 13.5.4 安装OpenCV 726
  - 13.5.5 下载目标检测模型SSDLite 726
  - 13.5.6 代码实现：用SSDLite 模型进行目标检测 727
- 第14章 商业实例——科技源于生活，用于生活 730
  - 14.1 实例71：将特征匹配技术应用在商标识别领域 730
    - 14.1.1 项目背景 730
    - 14.1.2 技术方案 730
    - 14.1.3 预处理图片——统一尺寸 731
    - 14.1.4 用自编码网络加夹角余弦实现商标识别 731
    - 14.1.5 用卷积网络加triplet-loss提升特征提取效果 731
    - 14.1.6 进一步的优化空间 732
  - 14.2 实例72：用RNN抓取蠕虫病毒 732
    - 14.2.1 项目背景 733
    - 14.2.2 判断是否恶意域名不能只靠域名 733
    - 14.2.3 如何识别恶意域名 733
  - 14.3 实例73：迎宾机器人的技术关注点——体验优先 734
    - 14.3.1 迎宾机器人的产品背景 734
    - 14.3.2 迎宾机器人的实现方案 734
    - 14.3.3 迎宾机器人的同类产品 736
  - 14.4 实例74：基于摄像头的路边停车场项目 737
    - 14.4.1 项目背景 737
    - 14.4.2 技术方案 738
    - 14.4.3 方案缺陷 738
    - 14.4.4 工程化补救方案 738
  - 14.5 实例75：智能冰箱产品——硬件成本之痛 739
    - 14.5.1 智能冰箱系列的产品背景 739
    - 14.5.2 智能冰箱的技术基础 740
    - 14.5.3 真实的非功能性需求——低成本 740
    - 14.5.4 未来的技术趋势及应对策略 741

• • • • • ([收起](#))

[深度学习之TensorFlow工程化项目实战\\_下载链接1](#)

标签

深度学习

案例多

实践书籍

代码讲解详细

TensorFlow丰富

机器学习

中国

2019

## 评论

可以做进阶学习和参考书

-----  
在书上读大段大段的代码体验真的很差，另外书中的代码和数据要关注公众号才能获取也是很令人无语的。

补充：书中的整体实践过程只是到完成一个可以使用的项目，对于模型是否合适以及相应的调优过程没有涉及，对于工程实践来说并不推荐。

-----  
最近在看这本书，感觉这本书相比其他TF书籍来讲写的要精细的多，很多原理性的内容阐述的比较清楚，对于一些基础知识和相对深入一些的理论都讲的比较透彻。书的整体脉络还是比较清晰的，跟着书上的内容走能够很快的进入状态。整体感觉书还不错。作者思路比较清晰，案例简单易懂，还有公众号、论坛和qq群。作者服务很到位。强烈推荐

-----  
书很厚，案例丰富，里面主要是针对实际工程项目介绍代码。与市面上的主流书籍不同，这本书对代码的介绍特别详细，将大段大段的代码进行拆分讲解，并配有详细的注释

。里面涵盖了包括模型微调、结构优化、多终端部署、团队合作、分布训练、继续优化模型的扩展方案及实例等各个开发过程中所遇到的工程化问题。目前正在学相关的内容，作为一个将AI成果用于工程领域的开发人员很有用。

-----  
书很厚，案例丰富，里面主要是针对实际工程项目介绍代码。与市面上的主流书籍不同，这本书对代码的介绍特别详细，将大段大段的代码进行拆分讲解，并配有详细的注释。里面涵盖了包括模型微调、结构优化、多终端部署、团队合作、分布训练、继续优化模型的扩展方案及实例等各个开发过程中所遇到的工程化问题。目前正在学相关的内容，作为一个将AI成果用于工程领域的开发人员很有用。

-----  
这是深度学习方向比较新的一本书，几乎覆盖了TensorFlow中的各个子模块，当然也有tf2.0的内容，是一本很到位的实践书籍。该书的配套资源提供了网盘、github、官网主页3种下载渠道。同时还有论坛和qq群、微信群，方便读者沟通交流，并由作者亲自答疑，学习起来来也不孤单。

-----  
[深度学习之TensorFlow工程化项目实战\\_下载链接1](#)

## 书评

书很厚，案例丰富，里面主要是针对实际工程项目介绍代码。与市面上的主流书籍不同，这本书对代码的介绍特别详细，将大段大段的代码进行拆分讲解，并配有详细的注释。里面涵盖了包括模型微调、结构优化、多终端部署、团队合作、分布训练、继续优化模型的扩展方案及实例等各个开...

-----  
[深度学习之TensorFlow工程化项目实战\\_下载链接1](#)