

# 深入浅出PyTorch

## Contents

- [人员安排](#)
- [四、课程编排与使用方法](#)
- [五、关于贡献](#)
- [六、更新计划](#)
- [七、鸣谢与反馈](#)
- [八、关注我们](#)
- [LICENSE](#)

PyTorch是利用深度学习进行数据科学研究的重要工具，在灵活性、可读性和性能上都具备相当的优势，近年来已成为学术界实现深度学习算法最常用的框架。

考虑到PyTorch的学习兼具理论储备和动手训练，两手都要抓两手都要硬的特点，我们开发了《深入浅出PyTorch》课程，期望以组队学习的形式，帮助大家从入门到熟练掌握PyTorch工具，进而实现自己的深度学习算法。

我们的愿景是：通过组队学习，大家能够掌握由浅入深地PyTorch的基本知识和内容，经过自己的动手实践加深操作的熟练度。同时通过项目实战，充分锻炼编程能力，掌握PyTorch进行深度学习的基本流程，提升解决实际问题的能力。

学习的先修要求是，会使用Python编程，了解包括神经网络在内的机器学习算法，勤于动手实践。

《深入浅出PyTorch》是一个系列，一共有三个部分。已经上线的是本系列的第一、二部分，后续会不断更新《深入浅出PyTorch》（下），给出更贴合实际应用的实战案例。

## 目录

- [第零章：前置知识](#)
  - [人工智能简史](#)
  - [模型评价指标](#)
  - [常用包的学习](#)
  - [Jupyter notebook/Lab 简述](#)
- [第一章：PyTorch的简介和安装](#)
  - [1.1 PyTorch简介](#)
  - [1.2 PyTorch的安装](#)
  - [1.3 PyTorch相关资源](#)
- [第二章：PyTorch基础知识](#)
  - [2.1 张量](#)
  - [2.2 自动求导](#)
  - [2.3 并行计算简介](#)
  - [AI硬件加速设备](#)
- [第三章：PyTorch的主要组成模块](#)
  - [3.1 思考：完成深度学习的必要部分](#)
  - [3.2 基本配置](#)
  - [3.3 数据读入](#)
  - [3.4 模型构建](#)
  - [3.5 模型初始化](#)
  - [3.6 损失函数](#)

- [3.7 训练和评估](#)
  - [3.8 可视化](#)
  - [3.9 PyTorch优化器](#)
- [第四章：PyTorch基础实战](#)
  - [4.1 ResNet](#)
  - [基础实战——FashionMNIST时装分类](#)
- [第五章：PyTorch模型定义](#)
  - [5.1 PyTorch模型定义的方式](#)
  - [5.2 利用模型块快速搭建复杂网络](#)
  - [5.3 PyTorch修改模型](#)
  - [5.4 PyTorch模型保存与读取](#)
- [第六章：PyTorch进阶训练技巧](#)
  - [6.1 自定义损失函数](#)
  - [6.2 动态调整学习率](#)
  - [6.3 模型微调-torchvision](#)
  - [6.3 模型微调 - timm](#)
  - [6.4 半精度训练](#)
  - [6.5 数据增强-imgaug](#)
  - [6.6 使用argparse进行调参](#)
- [第七章：PyTorch可视化](#)
  - [7.1 可视化网络结构](#)
  - [7.2 CNN可视化](#)
  - [7.3 使用TensorBoard可视化训练过程](#)
  - [7.4 使用wandb可视化训练过程](#)
- [第八章：PyTorch生态简介](#)
  - [8.1 本章简介](#)
  - [8.2 torchvision](#)
  - [8.3 PyTorchVideo简介](#)
  - [8.4 torchtext简介](#)
  - [8.5 torchaudio简介](#)
- [第九章：PyTorch的模型部署](#)
  - [9.1 使用ONNX进行部署并推理](#)
- [第十章：常见代码解读](#)
  - [10.1 图像分类简介（补充中）](#)
  - [目标检测简介](#)
  - [10.3 图像分割简介（补充中）](#)
  - [ResNet源码解读](#)
  - [文章结构](#)
  - [为什么需要 RNN?](#)
  - [RNN 理解及其简单实现](#)
  - [RNN 完成文本分类任务](#)
  - [RNN 存在的问题](#)
  - [文章结构](#)
  - [LSTM 理解](#)
  - [LSTM 实战](#)
  - [关于梯度问题](#)
  - [Transformer 解读](#)
  - [ViT解读](#)
  - [Swin Transformer解读](#)

# 人员安排

成员	个人简介	个人主页
牛志康	DataWhale成员，西安电子科技大学本科生	<a href="#">[知乎]</a> <a href="#">[个人主页]</a>
李嘉骐	DataWhale成员，清华大学研究生	<a href="#">[知乎]</a>
刘洋	Datawhale成员，中国科学院数学与系统科学研究所研究生	<a href="#">[知乎]</a>
陈安东	DataWhale成员，中央民族大学研究生	<a href="#">[个人主页]</a>

教程贡献情况（已上线课程内容）：

李嘉骐：第三章；第四章；第五章；第六章；第七章；第八章；内容整合

牛志康：第一章；第三章；第六章；第七章；第八章，第九章，第十章；文档部署

刘洋：第二章；第三章

陈安东：第二章；第三章；第七章

## 四、 课程编排与使用方法

部分章节直播讲解请观看B站回放（持续更新）：<https://www.bilibili.com/video/BV1L44y1472Z>

- 课程编排： 深入浅出PyTorch分为三个阶段：PyTorch深度学习基础知识、PyTorch进阶操作、PyTorch案例分析。
- 使用方法:  
我们的课程内容都以markdown格式或jupyter notebook的形式保存在本仓库内。除了多看加深课程内容的理解外，最重要的还是动手练习、练习、练习
- 组队学习安排:  
第一部分：第一章到第四章，学习周期：10天；  
第二部分：第五章到第八章，学习周期：11天

## 五、关于贡献

本项目使用Forking工作流，具体参考[atlassian文档](#)

大致步骤如下：

- 在GitHub上Fork本仓库
- Clone Fork后的个人仓库
- 设置upstream仓库地址，并禁用push
- 使用分支开发，课程分支名为lecture{#NO}，#NO保持两位，如lecture07，对应课程目录
- PR之前保持与原始仓库的同步，之后发起PR请求

命令示例：

```
# fork
# clone
git clone git@github.com:USERNAME/thorough-pytorch.git
# set upstream
git remote add upstream git@github.com:datawhalechina/thorough-pytorch.git
# disable upstream push
git remote set-url --push upstream DISABLE
# verify
git remote -v
# some sample output:
# origin      git@github.com:NoFish-528/thorough-pytorch.git (fetch)
# origin      git@github.com:NoFish-528/thorough-pytorch.git (push)
# upstream    git@github.com:datawhalechina/thorough-pytorch.git (fetch)
# upstream    DISABLE (push)
# do your work
git checkout -b lecture07
# edit and commit and push your changes
git push -u origin lecture07
# keep your fork up to date
## fetch upstream main and merge with forked main branch
git fetch upstream
git checkout main
git merge upstream/main
## rebase brach and force push
git checkout lecture07
git rebase main
git push -f
```

## Commit Message

提交信息使用如下格式: `<type>: <short summary>`

`<type>: <short summary>`

Summary in present tense. Not capitalized. No period at the end.

Commit Type: lecture{#NO}|others

others包括非课程相关的改动，如本README.md中的变动，.gitignore的调整等。


## 六、更新计划

内容	更新时间	内容
visdom可视化		Visdom的使用
apex		apex的简介和使用
模型部署		Flask部署PyTorch模型
TorchScript		TorchScript
并行训练		并行训练
模型预训练 - torchhub	2022.4.16	torchhub的简介和使用方法
目标检测 - SSD		SSD的简介和实现
目标检测 - RCNN系列		Fast-RCNN & Mask-RCNN
目标检测 - DETR		DETR的实现
图像分类 - GoogLeNet	2022.5.11	GoogLeNet的介绍与实现
图像分类 - MobileNet系列	2022.4月	MobileNet系列介绍与实现
图像分类 - GhostNet	2022.4月	GhostNet代码讲解
生成式对抗网络 - 生成手写数字实战	2022.5.25	生成数字并可视化
生成式对抗网络 - DCGAN		
风格迁移 - StyleGAN		
生成网络 - VAE		
图像分割 Deeplab系列		Deeplab系列代码讲解
自然语言处理 LSTM		LSTM情感分析实战
自然语言处理 Transformer		
自然语言处理 BERT		
视频		待定
音频		待定
自定义CUDA扩展和算子		

## 七、鸣谢与反馈

- 非常感谢DataWhale成员 叶前坤 @PureBuckwheat 和 胡锐锋 @Relph1119 对文档的细致校对！
- 如果有任何想法可以联系我们DataWhale也欢迎大家多多提出issue。

# 八、关注我们

Datawhale是一个专注AI领域的开源组织，以“for the learner，和学习者一起成长”为愿景，构建对学习者最有价值的开源学习社区。关注我们，一起学习成长。

## LICENSE

license CC BY-NC-SA 4.0

本作品采用[知识共享署名-非商业性使用-相同方式共享 4.0 国际许可协议](#)进行许可。

By ZhikangNiu  
© Copyright 2022, ZhikangNiu.