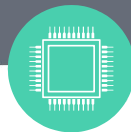




# 深度学习之 PyTorch 实战

## PyTorch 预备知识



主讲老师: 土豆老师

版权所有，侵权必究

# 目录

01

获取代码和安装运行环境

02

PyTorch 基本数据操作

03

PyTorch 自动求梯度

04

如何查阅文档和寻求帮助



# 获取代码和安装运行环境



- Anaconda 安装及环境管理

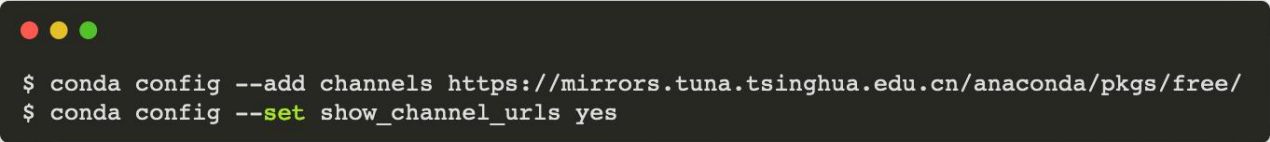
- Anaconda 是目前非常流行的一个 Python 包管理器，自带很多流行的 Python 库，包括 NumPy, Pandas 等，当然还有 Conda。而 Conda 是一个开源的软件包管理系统和环境管理系统，用于安装多个版本的软件包及其依赖关系，并在它们之间轻松切换。我们非常推荐使用 Anaconda 来管理你的 Python 环境，并且在 Jupyter Notebook/Lab 中完成所有的实战演练，随着你后面的不断学习，你会感受到它的精髓和好用。

- 官方网址: <https://www.anaconda.com/>
    - 操作系统? Windows / Mac / Linux ...
    - 虚拟环境? Conda / Miniconda / Pyenv / ...

# 获取代码和安装运行环境



- PyTorch 安装与调试
  - 官方网址: <https://pytorch.org/>
  - 操作系统? Windows / Mac / Linus ...
  - CPU版本? GPU 版本?
  - Conda 更换国内源下载:



```
$ conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/  
$ conda config --set show_channel_urls yes
```

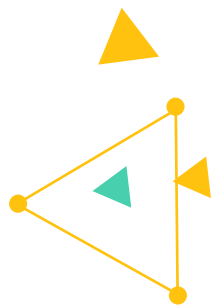
A terminal window with a black background and white text. It contains two lines of code. The first line is a command to add a new channel to the conda configuration. The second line is a command to set the show\_channel\_urls option to yes. The terminal window has a yellow triangle pointing to it from the right.

- 运行环境测试。
- 跳过本节不会影响后面的阅读。

# PyTorch 基本数据操作

“Talk is cheap. Show me the code.”

- PyTorch 是一个基于 Python 的科学计算包，充分发挥 GPU 能力而设计的 NumPy 的 GPU 版本替代方案，提供更大灵活性和速度的深度学习研究平台。
- 在深度学习中，我们通常会频繁地对数据进行操作。作为动手学深度学习的基础，本节将介绍如何对内存中的数据进行操作。



# PyTorch 基本数据操作

**“Talk is cheap. Show me the code.”**

- 创建 Tensor
  - 在 PyTorch 中，Tensor 是一个类，也是存储和变换数据的主要工具。为了简洁，人们常将 Tensor 实例直接称作 Tensor。如果你之前用过 NumPy，你会发现 Tensor 和 NumPy 的多维数组非常类似。然而，Tensor 提供 GPU 计算和自动求梯度等更多功能，这些使 Tensor 更加适合深度学习。
- Tensor 运算
- 广播机制
- 索引
- 与 NumPy 之间的相互转换
- CUDA Tensor

# PyTorch 自动求梯度

“Talk is cheap. Show me the code.”

- 在深度学习中，我们经常需要对函数求梯度（gradient）。PyTorch 提供的 `autograd` 包能够根据输入和前向传播过程自动构建计算图，并执行反向传播。本节将介绍如何使用 `autograd` 包来进行自动求梯度的有关操作。
- 基本概念
  - Tensor 是 PyTorch 实现多维数组计算和自动微分的关键数据结构。一方面，它类似于 NumPy 的 `NDArray`，用户可以对 Tensor 进行各种数学运算；另一方面，当设置 `.requires_grad = True` 之后，在其上进行的各种操作就会被记录下来，它将开始追踪在其上的所有操作，从而利用链式法则进行梯度传播。完成计算后，可以调用 `.backward()` 来完成所有梯度计算。此 Tensor 的梯度将累积到 `.grad` 属性中。



# PyTorch 自动求梯度

“Talk is cheap. Show me the code.”

- 如果不想被继续追踪，可以调用 `.detach()` 将其从追踪记录中分离出来，可以防止将来的计算被追踪，这样梯度就传不过去了。此外，还可以用 `with torch.no_grad()` 将不想被追踪的操作代码块包裹起来，这种方法在评估模型的时候很常用，因为在评估模型时，我们并不需要计算可训练参数 (`requires_grad=True`) 的梯度。

- 训练模式和预测模式
- 对 Python 控制流求梯度

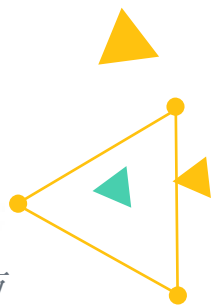




# 如何查阅文档和寻求帮助

“Talk is cheap. Show me the code.”

- 受课时所限，本课程无法对所有用到的 PyTorch 函数和类一一详细介绍。学员可以查阅相关文档来做更深入的了解。
- 查找模块里的所有函数和类。
  - dir
- 查找特定函数和类的使用。
  - help
- 在 PyTorch 网站上查阅。
  - <https://pytorch.org/>
- 自学路上必会之“上下求索”的技能([链接](#))
  - 土豆语录：没有哪个人会知道所有的答案，但所有与答案相关的信息都一定能在互联网上找到!
  - 活用百度和谷歌搜索。
- 自学路上必会之“提问的艺术”
  - [“如何提问的技巧”](#) from Fast.ai
  - [“提问的智慧”](#)





# 谢谢观看

更多好课，请关注万门大学APP

