

引入

0.1 本次课讲了什么

- 安装 Anaconda;
- 配置 Anaconda 的环境变量;
- 设置 Jupyter Notebook;
- Anaconda 虚拟环境;
- 安装 GPU 版本的 PyTorch 库。

0.2 适宜人群

- Windows 系统的用户，且不区分 win7、win10 和 win11;
- 没有任何 Python 常识、计算机内没装过任何 Python 软件的新手;
- 先后安装过 Anaconda 以及十几款不同版本 Python 解释器的老司机。

0.3 视频特点

- 本视频分辨率为 1080P，请调高分辨率;
- 此讲义 100%原创，请勿商用;
- Bilibili 的视频简介与置顶评论中均附有网盘链接，内含讲义与问题集锦。

0.4 安装常识

Python 解释器的名称由 Python 与版本号组成，如 Python3.9.0。解释器是向计算机解释 Python 语言的工具，只有下载解释器，计算机才能使用 Python 编程。因此，下载完 Python 3 后就可以在命令提示符（cmd）中进行 Python 编程。

代码编辑器是编辑 Python 代码的工具，使用代码编辑器前必须下载 Python 解释器。常用的 Python 代码编辑器包括 Jupyter Notebook、PyCharm、Visual Studio Code。类似的，C 的代码编辑器包括 Visual C++、Dev C++、Visual Studio Code。代码编辑器任选其一即可，本教程使用 Jupyter Notebook 作为代码编辑器。

Anaconda 是一款巨大的 Python 环境集成平台。其内含 Python 解释器、Jupyter Notebook 代码编辑器以及很多三方库，如 Numpy 数组库、Pandas 标签库、Matplotlib 绘图库等（但唯独缺少深度学习库）。

0.5 交流平台

本次课程所需的安装包，若官网无法下载，请前往 QQ 群的群文件进行下载。



一、安装 Anaconda

经实践，低版本的 Anaconda 仍然可以安装较新版本的三方库，因此已安装过的老司机不必卸载重装 Anaconda，哪怕你的计算机内还安装有十几款不同版本的 Python 解释器，只要你的 Anaconda 还能创建虚拟环境，就可跳过本章。

但是请注意，整个视频，老司机只能跳过本章，车速不要太快。

1.1 去镜像源下载 Anaconda

进入网址：<https://mirrors.bfsu.edu.cn/anaconda/archive/>，下载最新版本，视频中为 2022.10-Win 版本，其内部主环境（base 环境）下的 Python 为 3.9 版本。

Index of /			
Filename	Size	Last Modified	
.winzip/	-		
Anaconda3-2022.10-Linux-aarch64.sh	534.5M	2022-10-17 16:15:40	
Anaconda3-2022.10-Windows-x86_64.exe	621.2M	2022-10-17 16:15:39	
Anaconda3-2022.10-Linux-x86_64.sh	737.6M	2022-10-17 16:15:39	
Anaconda3-2022.10-MacOSX-x86_64.pkg	688.6M	2022-10-17 16:15:38	
Anaconda3-2022.10-MacOSX-arm64.sh	472.5M	2022-10-17 16:15:38	
Anaconda3-2022.10-MacOSX-x86_64.sh	681.6M	2022-10-17 16:15:37	
Anaconda3-2022.10-Linux-s390x.sh	282.4M	2022-10-17 16:15:37	
Anaconda3-2022.10-Linux-ppc64le.sh	360.0M	2022-10-17 16:15:37	
Anaconda3-2022.10-MacOSX-arm64.pkg	484.1M	2022-10-17 16:15:36	
Anaconda3-2022.05-MacOSX-arm64.sh	304.8M	2022-06-07 12:40:25	
Anaconda3-2022.05-MacOSX-arm64.pkg	316.4M	2022-06-07 12:40:24	

图 1-1 下载 Anaconda 安装包

考虑到后面会用虚拟环境，创建虚拟环境时可以设置新环境中的 Python 解释器版本，所以这里下载哪一版 Anaconda 并不重要。

1.2 安装 Anaconda

双击刚刚下载的 exe 文件，会有三个分岔口，分别按下列规则选择。

- ① Just me 和 All Users，选择 Just me；
- ② 安装路径选择最大的盘（一般是 D 盘），放在新建的【D:\Anaconda】里；
- ③ 最后一个分岔口，不勾选第一个方框，按照如图 1-2 所示选择。

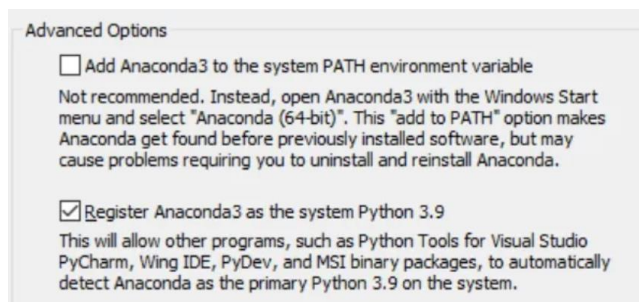


图 1-2 最后一个分岔口

安装过程可能很漫长，进度条会停在约 90% 的位置共 20 分钟，请耐心等待。安装完毕后，我们于第二章手动添加环境变量。

二、配置 Anaconda 的环境变量

请跳过第一章的老司机检查自己的环境变量是否安装正确。

环境变量的打开方法：

① 桌面按下鼠标右键，点击“显示设置”，如图 2-1 所示。



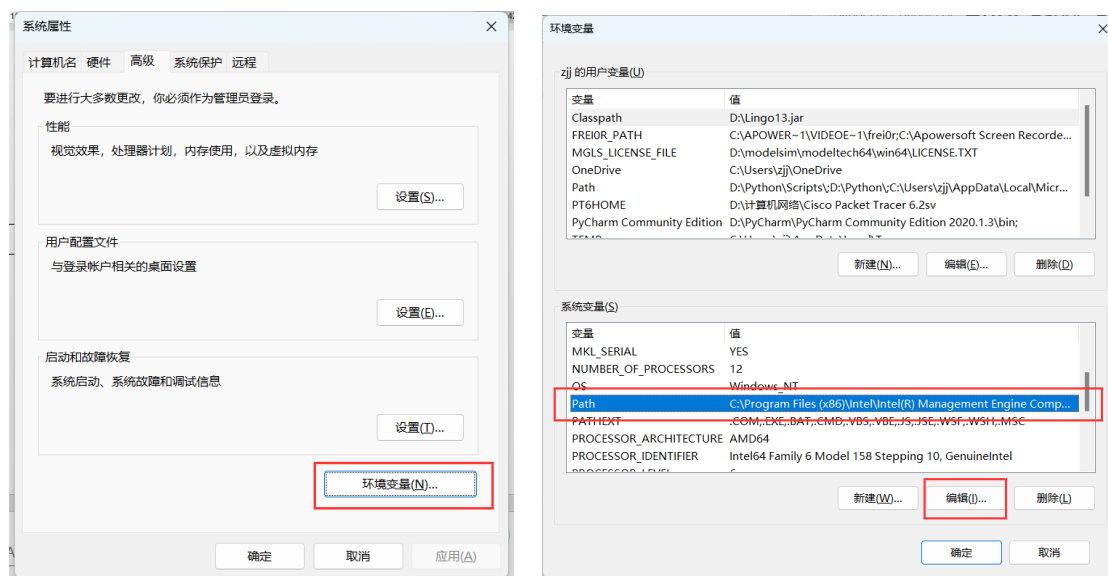
图 2-1 显示设置

② 在左上角“查找设置”中输入“环境变量”，点击“编辑系统环境变量”。



图 2-2 点击“编辑系统环境变量”

③ 在弹出的“系统属性”窗口中点击“环境变量”，再在弹出的“环境变量”窗口中选中 path 路径，并点击编辑。



(a) “系统属性”窗口

(b) “环境变量”窗口

图 2-3 打开“编辑系统环境变量”

④ 通过右侧的“新建”按钮，可新建环境变量的路径，将【D:\Anaconda】、【D:\Anaconda\Scripts】与【D:\Anaconda\Library\bin】添加到环境变量。

请注意，根据此视频旧版的读者反馈，若此前用户为其它多余的 Python 解释器添加过环境变量，请删除之，否则 Anaconda 的环境变量会被挤掉。

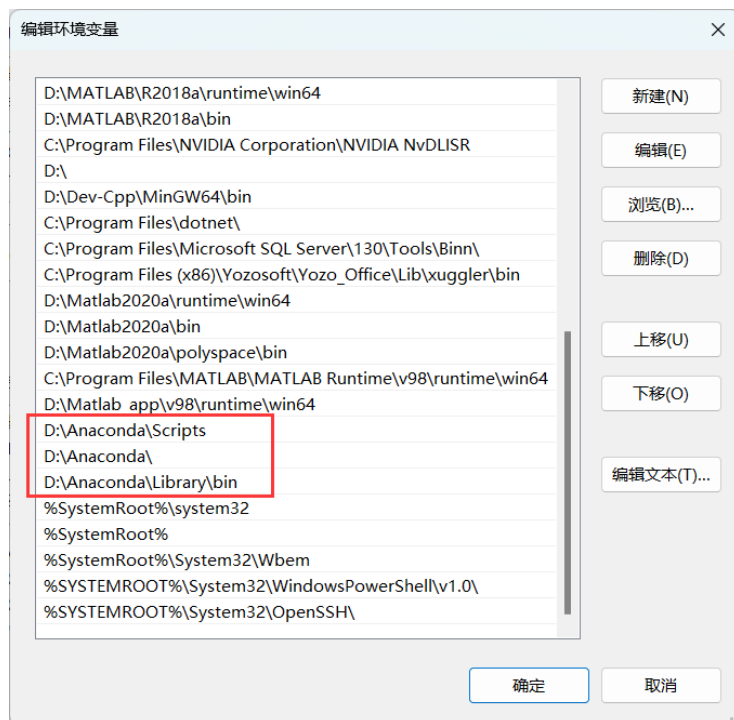


图 2-4 添加环境变量

此外，最后请注意，若某人的 Anaconda 安装路径不是 D:\Anaconda，而是 E:\Anaconda，以上三个环境变量请改为【E:\Anaconda】、【E:\Anaconda\Scripts】与【E:\Anaconda\Library\bin】。

三、设置 Jupyter Notebook

3.1 添加快捷方式

先找到 Jupyter 的安装路径，win10 和 win11 的方法如图 3-1 所示。



图 3-1 找到 Jupyter 的位置

找到 Jupyter 的位置后，把 Jupyter 和 Prompt 复制到桌面，如图 3-2 所示。

名称	修改日期	类型	大小
Anaconda Navigator (Anaconda)	2023/2/1 10:36	快捷方式	1 KB
Anaconda Powershell Prompt (Anaconda)	2023/2/1 10:36	快捷方式	3 KB
Anaconda Prompt (Anaconda)	2023/2/1 10:36	快捷方式	2 KB
Jupyter Notebook (Anaconda)	2023/2/1 10:36	快捷方式	1 KB
Reset Spyder Settings (Anaconda)	2023/2/1 10:36	快捷方式	1 KB

图 3-2 直接把快捷方式复制到桌面

3.2 用户名为中文的解决办法

计算机用户名（即 C:\Users\用户名）为中文，无法兼容 Jupyter。大家可以打开 Prompt 检查自己的用户名，如图 3-3。

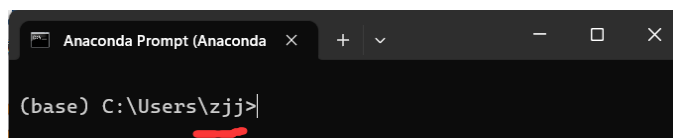


图 3-3 查看自己的用户名

如果发现自己的用户名携带有中文，在 Prompt 里输入以下两端代码：

```
pip uninstall pyzmq
pip install pyzmq==19.0.2
```

输入后，再双击 Jupyter，应该就可以了。若还解决不了，就重装系统盘（C 盘），重装后用户名自动变英文，但这个风险很大，小白不要轻易尝试。不行的话，听完本次课程后，下载 PyCharm 代码编辑器，创建项目时导入 Anaconda 环境，即可替代 Jupyter。

3.3 修改 Jupyter 的工作路径

Jupyter 初始的工作路径为【C:\Users\用户名】，需要进行修正，将其转移到新建的【D:\Jupyter】位置。

- ① 新建 D:\Jupyter;
- ② 打开桌面快捷方式中的 Prompt;
- ③ 输入 `jupyter notebook --generate-config` 命令并执行;
- ④ 打开上一步生成的配置文件地址，即

C:\Users\用户名\.jupyter

⑤ 在 `jupyter_notebook_config.py`（以记事本方式打开）中使用 `Ctrl + F` 查找并且修改如下配置项：

修改前：# `c.NotebookApp.notebook_dir = "`

修改后： `c.NotebookApp.notebook_dir = 'D:\Jupyter'`

也即删除前面的#号注释，在后面的单引号里输入要设置的目录路径，注意，'D:\Jupyter' 中不能有空格，否则 Jupyter 打开就闪退。保存后关闭。

⑥ 找到桌面的 `jupyter notebook` 快捷图标，鼠标右键>>属性>>快捷方式>>目标，删除最后的"%USERPROFILE%/"。

3.4 修改 Jupyter 字体

初始字体可以进行修改，修改流程如下。

- ① 打开如下地址

D:\Anaconda\Lib\site-packages\notebook\static\components\codemirror\lib

- ② 打开 `codemirror.css` 文件;

- ③ `Ctrl+F`，搜索“`font-family: monospace;`”的文字，并将其改为

`font-family: 'Fira Code Light','Consolas';`

改完后，如图 3-4 所示。

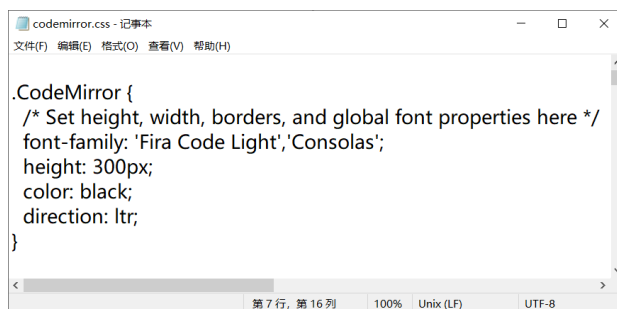


图 3-4 修改 Jupyter 字体

3.5 熟悉 Jupyter 界面

见视频。

四、Anaconda 虚拟环境

在 2022.10 版本的 Anaconda 中，其 base 环境中主要几个库的版本分别是：Python 3.9、NumPy 1.21.5、Pandas 1.2.4、Matplotlib 3.5.1。当然，Anaconda 的 base 环境中的库非常非常多，这只是深度学习需要的几个。

很多时候我们需要复刻其它演示代码中的环境，因此，虚拟环境必须掌握。虚拟环境想创建多少个，就创建多少，而且初始的虚拟环境基本没什么库，演示代码里说需要什么版本的库，我们就手动安装什么版本的库。

4.1 虚拟环境基础命令

点击 Prompt 进入 Anaconda 的环境中，接下来的命令均在 Prompt 中执行。

(1) Prompt 清屏

```
# 清屏  
cls
```

(2) base 环境下的操作

```
# 列出所有的虚拟环境  
conda env list
```

```
# 创建名为“环境名”的虚拟环境，并指定 Python 解释器的版本  
conda create -n 环境名 python=3.9
```

```
# 删除名为“环境名”的虚拟环境  
conda remove -n 环境名 --all
```

```
# 进入名为“环境名”的虚拟环境  
conda activate 环境名
```

(3) 虚拟环境内的操作

(pip 安装若失败，在【pip install 库==版本】后加【-i <https://pypi.tuna.tsinghua.edu.cn/simple>】即可)

```
# 列出当前环境下的所有库  
conda list
```

```
# 安装 NumPy 库，并指定版本 1.12.5  
pip install numpy==1.21.5 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
# 安装 Pandas 库，并指定版本 1.2.4  
pip install Pandas==1.2.4 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
# 安装 Matplotlib 库，并指定版本 3.5.1  
pip install Matplotlib==3.5.1 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
# 查看当前环境下某个库的版本（以 numpy 为例）  
pip show numpy
```

```
# 退出虚拟环境  
conda deactivate
```

4.2 虚拟环境连接 Jupyter 内核

经 4.1 小节，我们在 Anaconda 里创建了一个叫 DL 的虚拟环境，并在其中安装了数据科学三大基库——Numpy 数组库、Pandas 标签库、Matplotlib 绘图库。

但是现在这个叫 DL 的虚拟环境没有连接 Jupyter 内核，换句话说，Jupyter 现在只有与 base 环境相连。

为让虚拟环境与 Jupyter 内核相连，请在 Prompt 的虚拟环境下操作下列命令。若列出 Jupyter 的内核列表时反馈“Jupyter 不是内部或外部命令”，则先操作后面两步，再回头列出 Jupyter 的内核列表即可。

```
# 列出 Jupyter 的内核列表
```

```
jupyter kernelspec list
```

```
# 安装 ipykernel
```

```
pip install -i https://pypi.tuna.tsinghua.edu.cn/simple ipykernel
```

```
# 将虚拟环境导入 Jupyter 的 kernel 中
```

```
python -m ipykernel install --user --name=环境名
```

```
# 删除虚拟环境的 kernel 内核
```

```
jupyter kernelspec remove 环境名
```


五、安装 GPU 版本的 PyTorch 库

PyTorch 虽然是一个库，但安装时的核心组件叫 torch，还额外有两个小组件：torchvision 和 torchaudio。

由于 PyTorch 库的下载组件内部含有 cudatoolkit，它是 CUDA 的子集，里面的东西足够 PyTorch 使用，因此本教程不用单独安装 CUDA 和 CUDNN，也不用考虑 PyTorch 内置 cuda 与计算机显卡 CUDA 版本之间的关系（已亲测）。

本计算机的 CUDA 版本为 11.1，如图 5-1，后续直接安装内置 cuda11.3 版本的 torch。

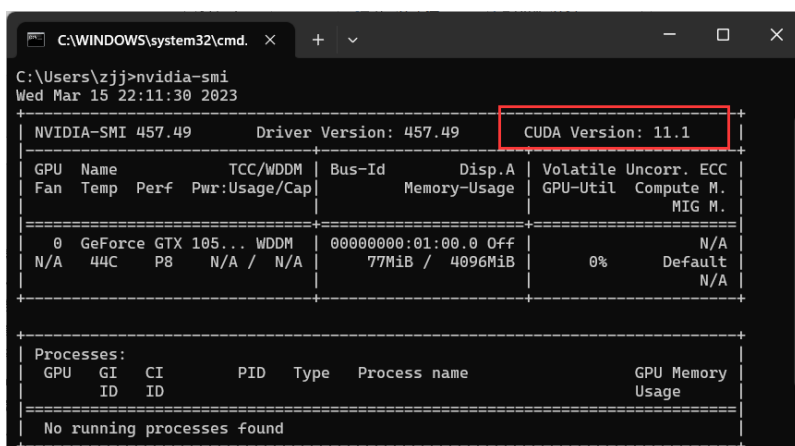


图 5-1 本机 CUDA 版本不重要

5.1 基本的 GPU 概念

对于计算机而言，中央处理器 CPU 是主板上的一块芯片，图形处理器 GPU 是显卡上的一块芯片。每台计算机必有主板，但少数计算机可能没有显卡。

显卡的全称是“显示适配器”，显卡最初被发明是单纯为了大型 3D 游戏用，后来被发现还可以用来顺带加速 PyTorch 的运行速度（比 CPU 快 10-100 倍）。

查看自己的计算机的显卡为：任务管理器——性能——左侧栏划到最下面。

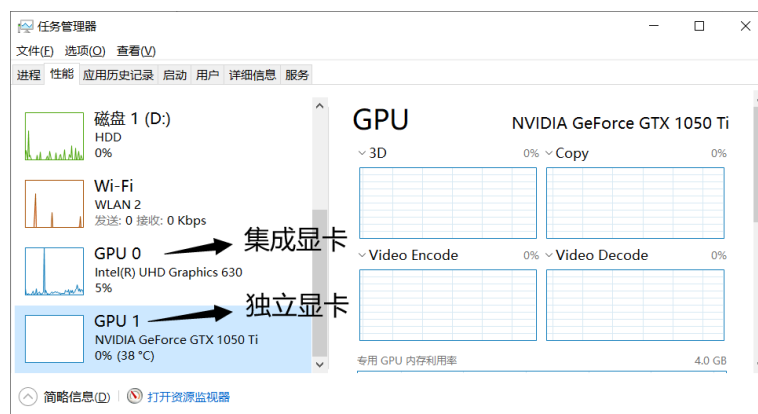


图 5-2 查看显卡

NVIDIA 的独立显卡必须有！而且只能是 NVIDIA，不能是其他牌子。没有 NVIDIA 显卡的同学也没事，经测试，可以跟我们一起安装 GPU 版本的 PyTorch，只不过 torch.cuda.is_available() 命令返回 false（也就是无法使用显卡的意思）。

5.2 安装 PyTorch

安装 torch 前，先给出一张推荐表，其中 cu113 即 torch 内部的 cudatoolkit 是 cuda 11.3 版本。所需的 Python 版本优先选择 3.9，除非演示代码告知具体版本。

表 5-1 目前所有 torch 版本的推荐按照表

torch 版本	推荐的内置 cuda 版本	所需的 Python 版本
torch 1.13.1	cu116	3.7、3.8、3.9、3.10
torch 1.13.0	cu116	3.7、3.8、3.9、3.10
torch 1.12.1	cu113	3.7、3.8、3.9、3.10
torch 1.12.0	cu113	3.7、3.8、3.9、3.10
torch 1.11.0	cu113	3.7、3.8、3.9、3.10
torch 1.10.2	cu113	3.6、3.7、3.8、3.9
torch 1.10.1	cu113	3.6、3.7、3.8、3.9
torch 1.10.0	cu113	3.6、3.7、3.8、3.9
torch 1.9.1	cu111	3.6、3.7、3.8、3.9
torch 1.9.0	cu111	3.6、3.7、3.8、3.9
torch 1.8.1	cu111	3.6、3.7、3.8、3.9
torch 1.8.0	cu111	3.6、3.7、3.8、3.9
torch 1.7.1	cu110	3.6、3.7、3.8、3.9
torch 1.7.0	cu110	3.6、3.7、3.8
torch 1.6.0	cu101	3.6、3.7、3.8
torch 1.5.1	cu101	3.5、3.6、3.7、3.8
torch 1.5.0	cu101	3.5、3.6、3.7、3.8

注：英伟达显卡 30 系列（如 NVIDIA GeForce RTX 3050）只能 cuda11.0 及其以上的版本。

现假设某个演示代码中所安装的 torch 是 1.12.0 版本（给定了 torch 版本后，torchvision 和 torchaudio 也唯一确定了），根据表 5-1，推荐的内置 cuda 是 11.3，优先选择 python 3.9 版本，进入 PyTorch 官网：

<https://pytorch.org/get-started/previous-versions/>

在其中 Ctrl + F 搜索 **【 pip install torch==1.12.0 】**，如图 5-3 所示。

Linux and Windows

```
# ROCm 5.1.1 (Linux only)
pip install torch==1.12.0+rocm5.1.1 torchvision==0.13.0+rocm5.1.1 torchaudio==0.12.0 --extra-index-url http
# CUDA 11.6
pip install torch==1.12.0+cu116 torchvision==0.13.0+cu116 torchaudio==0.12.0 --extra-index-url https://down
# CUDA 11.3
pip install torch==1.12.0+cu113 torchvision==0.13.0+cu113 torchaudio==0.12.0 --extra-index-url https://down
# CUDA 10.2
pip install torch==1.12.0+cu102 torchvision==0.13.0+cu102 torchaudio==0.12.0 --extra-index-url https://down
# CPU only
pip install torch==1.12.0+cpu torchvision==0.13.0+cpu torchaudio==0.12.0 --extra-index-url https://download
```

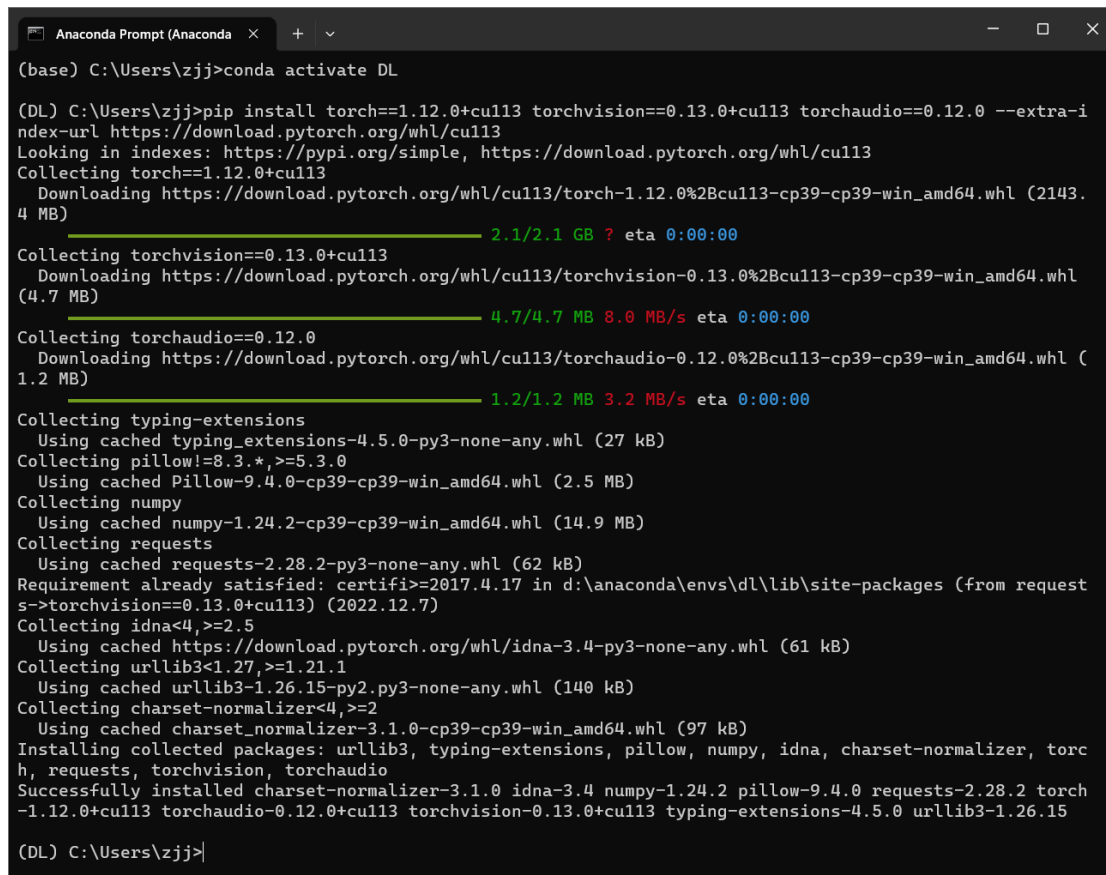
图 5-3 搜索结果

(1) 方法一：直接法

复制网页里的那段代码，也即

```
pip install torch==1.12.0+cu113 torchvision==0.13.0+cu113 torchaudio==0.12.0 --extra-index-url https://download.pytorch.org/whl/cu113
```

双击 Prompt，进入 Python 3.9 的虚拟环境 DL 下运行，如图 5-4。



```
Anaconda Prompt (Anaconda) x + v
(base) C:\Users\zjj>conda activate DL
(DL) C:\Users\zjj>pip install torch==1.12.0+cu113 torchvision==0.13.0+cu113 torchaudio==0.12.0 --extra-index-url https://download.pytorch.org/whl/cu113
Looking in indexes: https://pypi.org/simple, https://download.pytorch.org/whl/cu113
Collecting torch==1.12.0+cu113
  Downloading https://download.pytorch.org/whl/cu113/torch-1.12.0%2Bcu113-cp39-cp39-win_amd64.whl (2143.4 MB)
  2.1/2.1 GB ? eta 0:00:00
Collecting torchvision==0.13.0+cu113
  Downloading https://download.pytorch.org/whl/cu113/torchvision-0.13.0%2Bcu113-cp39-cp39-win_amd64.whl (4.7 MB)
  4.7/4.7 MB 8.0 MB/s eta 0:00:00
Collecting torchaudio==0.12.0
  Downloading https://download.pytorch.org/whl/cu113/torchaudio-0.12.0%2Bcu113-cp39-cp39-win_amd64.whl (1.2 MB)
  1.2/1.2 MB 3.2 MB/s eta 0:00:00
Collecting typing-extensions
  Using cached typing_extensions-4.5.0-py3-none-any.whl (27 kB)
Collecting pillow!=8.3.*,>=5.3.0
  Using cached Pillow-9.4.0-cp39-cp39-win_amd64.whl (2.5 MB)
Collecting numpy
  Using cached numpy-1.24.2-cp39-cp39-win_amd64.whl (14.9 MB)
Collecting requests
  Using cached requests-2.28.2-py3-none-any.whl (62 kB)
Requirement already satisfied: certifi<=2017.4.17 in d:\anaconda\envs\dl\lib\site-packages (from requests->torchvision==0.13.0+cu113) (2022.12.7)
Collecting idna<4,>=2.5
  Using cached https://download.pytorch.org/whl/idna-3.4-py3-none-any.whl (61 kB)
Collecting urllib3<1.27,>=1.21.1
  Using cached urllib3-1.26.15-py2.py3-none-any.whl (140 kB)
Collecting charset-normalizer<4,>=2
  Using cached charset_normalizer-3.1.0-cp39-cp39-win_amd64.whl (97 kB)
Installing collected packages: urllib3, typing-extensions, pillow, numpy, idna, charset-normalizer, torch, requests, torchvision, torchaudio
Successfully installed charset-normalizer-3.1.0 idna-3.4 numpy-1.24.2 pillow-9.4.0 requests-2.28.2 torch-1.12.0+cu113 torchaudio-0.12.0+cu113 torchvision-0.13.0+cu113 typing-extensions-4.5.0 urllib3-1.26.15
(DL) C:\Users\zjj>
```

图 5-4 虚拟环境下安装 torch

看到最后几行代码里有 Successfully installed 就算成功。

(2) 方法二：轮子法

按视频中的方式下载轮子文件，或进入 QQ 群，在群文件中下载。下载好后，将三个 whl 文件放在新建的 D:\whl 文件夹中。

安装命令为（以下命令在 Prompt 的虚拟环境 DL 中执行）：

```
pip install D:\whl\torch-1.12.0+cu113-cp39-cp39-win_amd64.whl
pip install D:\whl\torchvision-0.13.0+cu113-cp39-cp39-win_amd64.whl
pip install D:\whl\torchaudio-0.12.0+cu113-cp39-cp39-win_amd64.whl
```

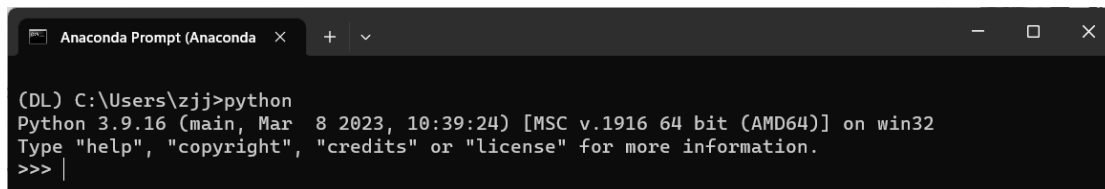
这三行代码的结构都是：pip install 路径\轮子名.whl。

5.3 检验 PyTorch 是否安装成功

接下来的操作，可以在 Prompt 里以直接运行 Python 解释器的方式检验，也可以在 Jupyter 代码编辑器里检验。

(1) 直接在 Python 解释器里检验

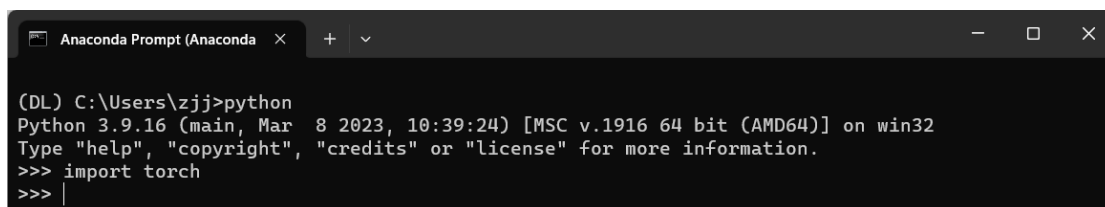
首先，进入虚拟环境 DL 后，输入 python 以进入解释器，如图 5-5。



```
(DL) C:\Users\zjj>python
Python 3.9.16 (main, Mar 8 2023, 10:39:24) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

图 5-5 进入 DL 虚拟环境的 python 解释器

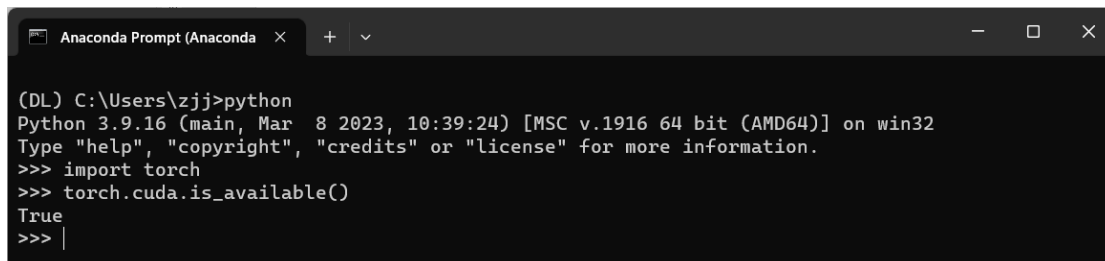
输入 import torch 导入 torch 库，如图 5-6 所示。若安装失败，则会返回 No module named 'torch'。若安装成功，不会返回任何语句，同时在下一行出现“>>>”，提示我们可以继续敲代码。



```
(DL) C:\Users\zjj>python
Python 3.9.16 (main, Mar 8 2023, 10:39:24) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> |
```

图 5-6 测试 torch 是否可以导入

最后一步，输入 torch.cuda.is_available()，如图 5-7 所示。



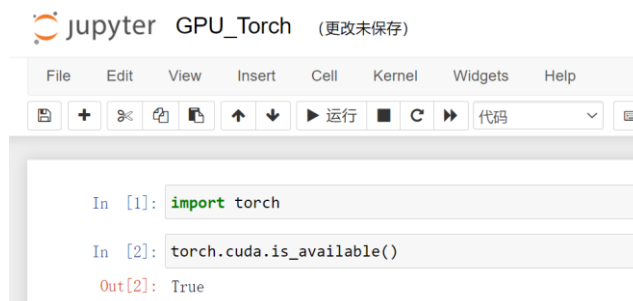
```
(DL) C:\Users\zjj>python
Python 3.9.16 (main, Mar 8 2023, 10:39:24) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> torch.cuda.is_available()
True
>>> |
```

图 5-7 测试 torch 是否能连接 cuda

如果你是 CPU 用户，那么会返回 false，这对你来说是正常现象。但对于 GPU 用户来说应当返回 True。

(2) 在 Jupyter 代码编辑器里检验

在 Jupyter 里，切换到 DL 内核后，输入两段代码后运行，如图 5-8 所示。



```
jupyter GPU_Torch (更改未保存)

File Edit View Insert Cell Kernel Widgets Help

In [1]: import torch

In [2]: torch.cuda.is_available()

Out[2]: True
```

图 5-8 Jupyter 替代 Prompt