

前言

首先要搭建好 python 开发环境，Anaconda 是一个非常不错的环境管理器和包管理器。相比 pip 安装第三方模块有时需要安装多个依赖包，Anaconda 实在是要方便很多。例如，安装 pandas 模块时，pip 中要依次安装 NumPy, dateutil, pytz ,setuptools, 而 Anaconda 中只要安装一个 pandas 包其他依赖包就会自动安装。conda 可以管理不同的运行环境，方便我们在不同版本的 python 间切换使用。非常好的一项功能就是共享环境。

目录

目录	2
1. Python 的安装	5
1.1 Windwos	5
1.2 Linux	5
1.2.1 yum 更新 yum 源	5
1.2.2 安装 Python	5
1.2.3 创建 python 虚拟环境	6
2. Anaconda 的安装和下载	7
2.1 Windows 安装	7
2.2 3.2 Linux 下安装:	8
2.2.1 下载	8
2.2.2 安装	8
2.2.3 配置环境变量	8
3. conda 安装及使用	9
3.1 conda 的安装	9
3.1.1 换国内源	9
3.1.2 临时使用镜像	10
3.1.3 版本导致的问题	10
3.2 Anaconda 的使用	11
3.2.1 Anaconda 安装后开始菜单配置问题	11

3.2.2	管理包.....	11
3.2.3	管理环境.....	11
3.2.4	共享环境.....	12
3.2.5	jupyter notebook 的配置	13
3.2.6	matplotlib 中文字体配置.....	14
4.	pip 的安装使用.....	14
4.1	pip 的安装	14
4.1.1	windows 下	14
4.1.2	Linux 下	15
4.1.3	安装包时临时配置	16
4.2	pip 的使用	16
4.2.1	安装包	16
4.2.2	查看所有版本	16
4.2.3	指定版本	16
4.2.4	卸载包	17
4.2.5	查看安装包信息	17
4.2.6	列出所有安装包	17
4.2.7	检测更新	17
4.2.8	更新包	17
4.2.9	环境导出导入	17
5.	TensorFlow&Pytorch GPU 版本安装.....	18
5.1	TesnsorFlow	22
5.2	Pytorch	24

1. Python 的安装

1.1 Windwos

如果不想用 Anaconda 可以到官网直接下载 python

Python 官网下载地址 <https://www.python.org/downloads/>

安装时添加环境变量

安装后确保环境变量中有如下的地址

根据自己的实际路径

D:\programs\python3.8\Scripts\

D:\programs\python3.8\

1.2 Linux

1.2.1 yum 更新 yum 源

yum update

✧ 安装 Python 3.7 所需的依赖否则安装后没有 pip3 包

yum install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel libffi-devel gcc make

✧ 在官网下载所需版本，这里用的是 3.8.5 版本

wget <https://www.python.org/ftp/python/3.8.5/Python-3.8.5.tgz>

1.2.2 安装 Python

✧ 解压

tar -xvf Python-3.8.5.tgz

✧ 配置编译

```
cd Python-3.8.5
```

✧ 配置编译的路径（这里--prefix 是指定编译安装的文件夹）

```
./configure --prefix=/usr/local/python3
```

✧ 执行该代码后，会编译安装到 /usr/local/bin/ 下，且不用添加软连接或环境变量

```
./configure --enable-optimizations
```

```
make && make install
```

✧ 添加软连接

```
ln -s /usr/local/python3/bin/python3 /usr/bin/python3
```

```
ln -s /usr/local/python3/bin/pip3 /usr/bin/pip3
```

✧ 将/usr/local/python3/bin 加入 PATH

```
vim /etc/profile
```

✧ 然后在文件末尾添加

```
export PATH=$PATH:/usr/local/python3/bin
```

✧ 修改完后，还需要让这个环境变量在配置信息中生效，执行命令

```
source /etc/profile
```

1.2.3 创建 python 虚拟环境

✧ 安装 virtualenv

```
yum install python-virtualenv
```

✧ 创建 python 虚拟环境

```
virtualenv env
```

✧ 执行后，在本地会生成一个与虚拟环境同名的文件夹

✧ 如果你的系统里安装有不同版本的 python，可以使用--python 参数指定虚拟环境的 python 版本：

```
virtualenv --python=/usr/local/python3/bin/python3 env
```

✧ 启动虚拟环境

```
source bin/activate
```

✧ 退出虚拟环境

```
deactivate #
```

2. Anaconda 的安装和下载

2.1 Windows 安装

官网地址：

<https://docs.conda.io/projects/conda/en/latest/user-guide/install/index.html>

官网下载地址：

<https://www.anaconda.com/products/individual>

官网下载慢的同学可移步到清华镜像源：

<https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/>

官网默认的版本一般是稳定版本，大家可以参考官网的版本到清华镜像源下载对应的版本

安装步骤参考附件：Anaconda 的下载和使用.pdf

2.2 3.2 Linux 下安装:

2.2.1 下载

wget https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/Anaconda3-2020.11-Linux-x86_64.sh

2.2.2 安装

```
bash Anaconda3-2020.11-Linux-x86_64.sh
```

2.2.3 配置环境变量

```
vi /etc/profile
```

在文件最后加入如下语句（路径需要根据自己的安装位置更改）

```
PATH=$PATH: /root/anaconda3/bin #路径名跟自己实际情况而定
```

```
export PATH
```

按住 shift 键+:键，输入 wq，保存文件并退出。最后使用如下命令刷新环境变量即可：

```
source /etc/profile
```

```
echo $PATH
```

2.2.4 Docker 安装 Anaconda

<https://docs.anaconda.com/anaconda/user-guide/tasks/docker/>

3. conda 安装及使用

3.1 conda 的安装

如果用户从来没有使用过 `conda config` 命令，就不会有配置文件，当用户第一次运行 `conda config` 命令时，将会在用户的家目录创建该文件，即一个名为`.condarc` 的文本文件，一般表示 conda 应用程序的配置文件，在用户的家目录之下：

windows: `C:\users\username\.condarc`

Linux: `/home/username/.condarc`)

注意：`condarc` 配置文件，是一种可选的（optional）运行期配置文件，其默认情况下是不存在的，但当用户第一次运行 `conda config` 命令时，才会在用户的家目录创建该文件。我可以通过 `conda config` 命令来配置该文件，也完全可以自己手动编辑也可以。

手动编辑

channels:

- <http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/main/>

- <http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/free/>

- <http://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/pytorch/>

- <http://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/msys2/>

ssl_verify: true

3.1.1 换国内源

✧ 查看源: `conda config --show-sources`

```
show_channel_urls: True
```

✧ 添加清华源: `conda config --add channels http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgsg/free/`

```
4 # 设置搜索时显示通道地址
5 conda config --set show_channel_urls yes
```

✧ 是否显示 channel 的来源

`conda config --set show_channel_urls yes/no`

如果是: `conda config --set show_channel_urls yes` 则配置文件中为 `show_channel_urls: True`。这表示在使用 `conda search package` 或者是 `conda install package` 的时候会显示这个包是来自于哪一个镜像源。

当然我也可以不显示, 则为: `conda config --set show_channel_urls no` 则配置文件中为 `show_channel_urls: False`

✧ 移除源

`conda config --remove channels http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgsg/free/`

3.1.2 临时使用镜像

`conda install -c 镜像源 包名`

例如:

`conda install -c http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgsg/free/ pandas`

3.1.3 版本导致的问题

Anaconda3 2020.11(Python 3.8.5)之前的版本镜像源使用 `https` 开头

<https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgsg/free/>

3.2 Anaconda 的使用

3.2.1 Anaconda 安装后开始菜单配置问题

win+R 运行 cmd, 输入一下命令

```
conda update menuinst #更新菜单栏 出现 conda install -f console_shortcut ipython ipy  
thon-notebook ipython-qtconsole launcher sp
```

3.2.2 管理包

```
conda list #查看已安装内容
```

```
conda upgrade --all #更新所有包
```

```
conda install package_name #安装包
```

```
conda search search_term 模糊查找包 如查找 numpy 可输入 conda search numpy
```

```
python setup.py install #进入下载好的第三方库路径下运行
```

```
conda install numpy=1.10 #指定所需的包版本
```

```
conda remove package_name #卸载包
```

```
conda update package_name #更新包
```

3.2.3 管理环境

✧ 创建环境和包

```
conda create -n env_name package_names
```

```
simple: conda create -n py2 pandas
```

✧ 指定版本

```
conda create -n py2 python=2
```

✧ 克隆环境

```
conda create -n new_env --clone old_env
```

✧ 进入环境

● Windows 上

```
conda activate my_env
```

● OSX/Linux 上

```
source activate my_env
```

✧ 在环境中安装包的命令与前面的一样：

```
conda install package_name
```

✧ 离开环境

● Windows 上

```
conda deactivate
```

● OSX/Linux 上

```
source deactivate
```

3.2.4 共享环境

✧ 当前命令窗口路径下输出环境中所有包的名称

```
conda env export>environment.yaml
```

✧ 进入你的环境，

```
activate py3
```

✧ 使用以下命令更新你的环境

```
conda env update -f=/path/to/environment.yaml
```

-f 表示你要导出的文件在本地的路径 所以/path/to/environment.yaml 要更换成你本地的实际路径

✧ 列出环境

```
conda env list
```

✧ 删除指定环境

```
conda env remove -n env_name
```

```
conda remove -n env_name --all
```

3.2.5 jupyter notebook 的配置

✧ conda 环境中安装 jupyter notebook

```
conda install jupyter notebook
```

✧ 安装环境自动关联包（目前只支持到 python3.5）

```
conda install nb_conda
```

```
conda install -n python_env ipykernel
```

进入 notebook

```
jupyter notebook
```

✧ 安装自动补全代码包

```
conda install pyreadline
```

jupyter notebook 中使用 Tab 键

✧ 为 jupyter notebook 添加目录

```
conda install -c conda-forge jupyter_contrib_nbextensions
```

方法：运行 Jupyter Notebook, 在打开的 Notebook 界面里, 你会发现多了一个 Nbextensions, 点击这个 tab 勾选 Table of Contents (有的版本是 toc2). 然后创建或者打开一个 Jupyter Notebook

3.2.6 matplotlib 中文字体配置

notebook 中输入

```
import matplotlib
```

```
matplotlib.matplotlib_fname()
```

查找配置文件路径

在 font 字体文件夹中的 ttf 文件夹中添加 SimHei.ttf 文件

修改 matplotlibrc 文件

```
font.family 去掉注释#
```

```
font.sans-serif 去掉注释# 添加 SimHei
```

```
axes.unicode_minus : False 改为 True
```

删除 C:\User\用户名下的 matplotlib 缓存

重启 jupyter Notebook

4. pip 的安装使用

4.1 pip 的安装

4.1.1 windows 下

✧ 查看当前镜像源

```
pip config list
```

✧ 更换镜像源

```
pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
```

```
pip config set install.trusted-host pypi.tuna.tsinghua.edu.cn
```

有时候镜像源找不到包可以更换镜像源

pip config set global.index-url <https://pypi.douban.com/simple>

pip config set install.trusted-host pypi.douban.com

网易 <https://mirrors.163.com/pypi/simple/>

阿里 <https://mirrors.aliyun.com/pypi/simple/>

✧ 查看镜像源配置地址以及当前使用的源

pip config -v list

pip 使用配置文件的搜索路径优先级是按照上述 list 的“从下往上”进行的

如果下面的最下面的 pip.ini 文件不存在就从倒数第二个开始

命令行默认配置的是类似 C:\Users\Nan\AppData\Roaming\pip\pip.ini

```
(tf2) C:\Users\Nan>pip config -v list
For variant 'global', will try loading 'C:\ProgramData\pip\pip.ini'
For variant 'user', will try loading 'C:\Users\Nan\pip\pip.ini'
For variant 'user', will try loading 'C:\Users\Nan\AppData\Roaming\pip\pip.ini'
For variant 'site', will try loading 'D:\programs\Anaconda3\envs\tf2\pip.ini'
global.index-url='https://mirrors.aliyun.com/pypi/simple/'
install.trusted-host='mirrors.aliyun'
```

✧ 移除镜像源

pip config unset global.index-url <https://pypi.tuna.tsinghua.edu.cn/simple>

pip config unset install.trusted-host [tuna.tsinghua.edu.cn/simple](https://pypi.tuna.tsinghua.edu.cn/simple)

或按照下面的方法

直接在 user 目录中创建一个 pip 目录，再新建文件 pip.ini。（例如：C:\Users\WQP\pip\pip.in

i) 内容同上。

注意：一定要确保系统环境变量中的有 pip.ini 文件所在的路径

4.1.2 Linux 下

在用户目录之下：~/.pip/pip.conf（没有就创建一个文件夹及文件。文件夹要加“.”，表示是隐藏文件夹）

添加如下内容：

[global]

iindex-url = <https://pypi.douban.com/simple> #豆瓣源, 可以换成其他的源

trusted-host = pypi.douban.com #添加豆瓣源为可信主机, 要不然可能报错

disable-pip-version-check = true #取消 pip 版本检查, 排除每次都报最新的

pip timeout = 120

4.1.3 安装包时临时配置

pip install -i 原地址 包名称

pip install -i <https://pypi.tuna.tsinghua.edu.cn/simple> numpy

常见的源有

豆瓣: <http://pypi.douban.com/simple/>

清华: <https://pypi.tuna.tsinghua.edu.cn/simple>

4.2 pip 的使用

4.2.1 安装包

pip install package_name

4.2.2 查看所有版本

pip install package_name==

4.2.3 指定版本

pip install package_name ==1.0.4

4.2.4 卸载包

```
pip uninstall package_name
```

4.2.5 查看安装包信息

```
pip show package_name
```

4.2.6 列出所有安装包

```
pip list
```

4.2.7 检测更新

```
pip list -outdated
```

4.2.8 更新包

```
pip install --upgrade package_name
```

4.2.9 环境导出导入

对于不使用 conda 的用户，可以使用这段命令 将一个 txt 文件导出并包括在其中

```
pip freeze >environment.txt
```

以管理员身份运行 cmd 下的命令，安装你刚导出来的 environment.txt /path/environment.txt 导出的文件在本地的实际路径

```
pip install -r /path/environment.txt
```

5. TensorFlow&Pytorch 安装

5.1 显卡配置

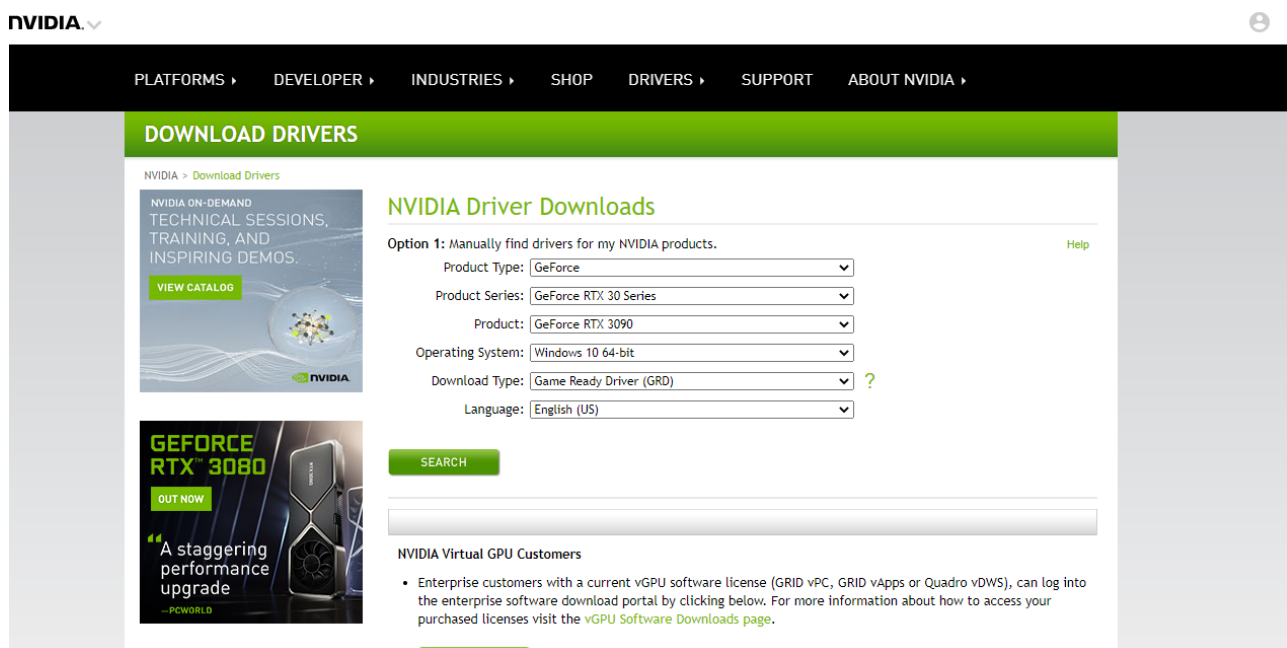
TensorFlow 和 Pytorch GPU 版本安装前还要根据自己的显卡安装对应的 cuda 和 CUDNN 版本

CUDA 是深度学习的 sdk CUDNN 是神经网络的 sdk

另外要注意 1.0 版本和 2.0 版本支持的 cuda 版本也不同

NVIDIA 官网: <https://developer.nvidia.com/cuda-gpus>

NVIDIA 驱动 <https://www.nvidia.com/Download/index.aspx>



NVIDIA > Download Drivers

NVIDIA ON-DEMAND TECHNICAL SESSIONS, TRAINING, AND INSPIRING DEMOS. [VIEW CATALOG](#)

GEFORCE RTX 3080 OUT NOW. "A staggering performance upgrade" -PCWORLD

NVIDIA Driver Downloads

Option 1: Manually find drivers for my NVIDIA products. [Help](#)

Product Type:

Product Series:

Product:

Operating System:

Download Type: ?

Language:

[SEARCH](#)

NVIDIA Virtual GPU Customers

- Enterprise customers with a current vGPU software license (GRID vPC, GRID vApps or Quadro vDWS), can log into the enterprise software download portal by clicking below. For more information about how to access your purchased licenses visit the [vGPU Software Downloads page](#).

CUDA 对应的驱动版本: <https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.htm>

docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html

DEVELOPER ZONE CUDA TOOLKIT DOCUMENTATION

CUDA 工具包 v11.1.1

发行说明

- CUDA 工具包主要组件
- CUDA 11.1 发行说明
 - CUDA 工具包主要组件版本
 - CUDA 11.1 Update 1 的新增功能
 - 通用 CUDA
 - CUDA 工具
 - CUDA 编译器
 - CUDA 开发人员工具
 - CUDA 库
 - cuFFT 库
 - cuSOLVER 库
 - CUDA 数学库
 - 不推荐使用的功能
 - 解决的问题
 - 通用 CUDA
 - CUDA 工具
 - cuBLAS 图书馆
 - cuFFT 库
 - 已知的问题

CUDA 驱动

运行 CUDA 应用程序要求系统至少具有一个支持 CUDA 的 GPU 和与 CUDA Toolkit 兼容的驱动程序。请参阅表 2。有关支持 CUDA 的各种 GPU 产品的更多信息，每个版本的 CUDA 工具包都需要最低版本的 CUDA 驱动程序。CUDA 驱动程序向后兼容，这意味着针对特定版本 CUDA 编译的应用程序将继续在后续（更高）有关兼容性的更多信息，请访问<https://docs.nvidia.com/cuda/cuda-c-best-practices-guide/index.html#cuda-runtime-and-driver-api-version>。

注意：从 CUDA 11.0 开始，工具箱组件分别进行了版本控制，并且工具箱本身也进行了版本控制，如下表所示。

CUDA 工具包	Linux x86_64 驱动程序版本	Windows x86_64 驱动程序版本
CUDA 11.1.1 更新 1	> = 455.32	> = 456.81
CUDA 11.1 GA	> = 455.23	> = 456.38
CUDA 11.0.3 更新 1	> = 450.51.06	> = 451.82
CUDA 11.0.2 GA	> = 450.51.05	> = 451.48
CUDA 11.0.1 RC	> = 450.36.06	> = 451.22
CUDA 10.2.89	> = 440.33	> = 441.22
CUDA 10.1 (10.1.105 常规发行版和更新)	> = 418.39	> = 418.96
CUDA 10.0.130	> = 410.48	> = 411.31
CUDA 9.2 (9.2.148 更新 1)	> = 396.37	> = 398.26
CUDA 9.2 (9.2.88)	> = 396.26	> = 397.44
CUDA 9.1 (9.1.85)	> = 390.46	> = 391.29
CUDA 9.0 (9.0.76)	> = 384.81	> = 385.54
CUDA 8.0 (8.0.61 GA2)	> = 375.26	> = 376.51
CUDA 8.0 (8.0.44)	> = 367.48	> = 369.30
CUDA 7.5 (7.5.16)	> = 352.31	> = 353.66
CUDA 7.0 (7.0.28)	> = 346.46	> = 347.62

为了方便起见，NVIDIA 驱动程序是作为 CUDA Toolkit 安装的一部分安装的。请注意，此驱动程序仅用于开发目的，不建议在 Tesla GPU 的生产中使用。

为了在生产中使用 Tesla GPU 运行 CUDA 应用程序，建议从 NVIDIA 驱动程序下载站点 (<http://www.nvidia.com/drivers>) 下载适用于 Tesla GPU 的最新驱动程序。

在 CUDA Toolkit 的安装过程中，可能会在 Windows（使用交互式或静默安装）或 Linux（通过使用元软件包）上跳过 NVIDIA 驱动程序的安装。

安装好驱动后查看驱动版本

NVIDIA 控制面板

文件(F) 编辑(E) 系统信息

选择一项任务...

- 3D 设置
 - 通过预览
 - 管理 3D
 - 配置 Surface
- 显示
 - 更改分辨率
 - 调整桌面
 - 旋转显示
 - 查看 HDCP
 - 设置数字
 - 调整桌面
 - 设置多个
- 视频
 - 调整视频
 - 调整视频

系统信息

NVIDIA 硬件及运行该硬件的系统的详细信息。

显示 组件

系统信息

操作系统: Windows 10 Home, 64-bit

DirectX 运行时版本: 12.0

图形卡信息

项目	细节
GeForce GTX 1050	驱动程序版本: 460.79 驱动器类型: DCH Direct3D 功能: 12_1 CUDA 核心: 640 图形时钟: 1354 MHz 内存数据速率: 7.01 Gbps 内存接口: 128-位 内存带宽: 112.13 GB/秒 全部可用的图: 12225MB 专用视频内存: 4096 MB GDDR5 系统视频内存: 8MB

关于(A)

保存(S) 关闭

CUDA Toolkit Archive: <https://developer.nvidia.com/cuda-toolkit-archive>



The screenshot shows the NVIDIA Developer website's CUDA Toolkit Archive page. The browser tabs include 'Start Locally | PyTorch', 'linux安装TensorFlow-GPU版本', and 'CUDA Toolkit Archive | NVIDIA'. The page header features the NVIDIA Developer logo and navigation links: HOME, BLOG, NEWS, FORUMS, DOCS, DOWNLOADS, and TRAINING. The main heading is 'CUDA Toolkit Archive'. Below it, a paragraph states that previous releases of the CUDA Toolkit, GPU Computing SDK, documentation, and developer drivers can be found using the link www.nvidia.com/drivers for more recent production drivers. Two buttons are present: 'Download Latest CUDA Toolkit' and 'Learn More about CUE'. The 'Latest Release' section highlights 'CUDA Toolkit 11.1.1 [Oct 2020], Versioned Online Documentation'. The 'Archived Releases' section lists various versions from 7.5 to 11.1.0, each with a link to its 'Versioned Online Documentation'.

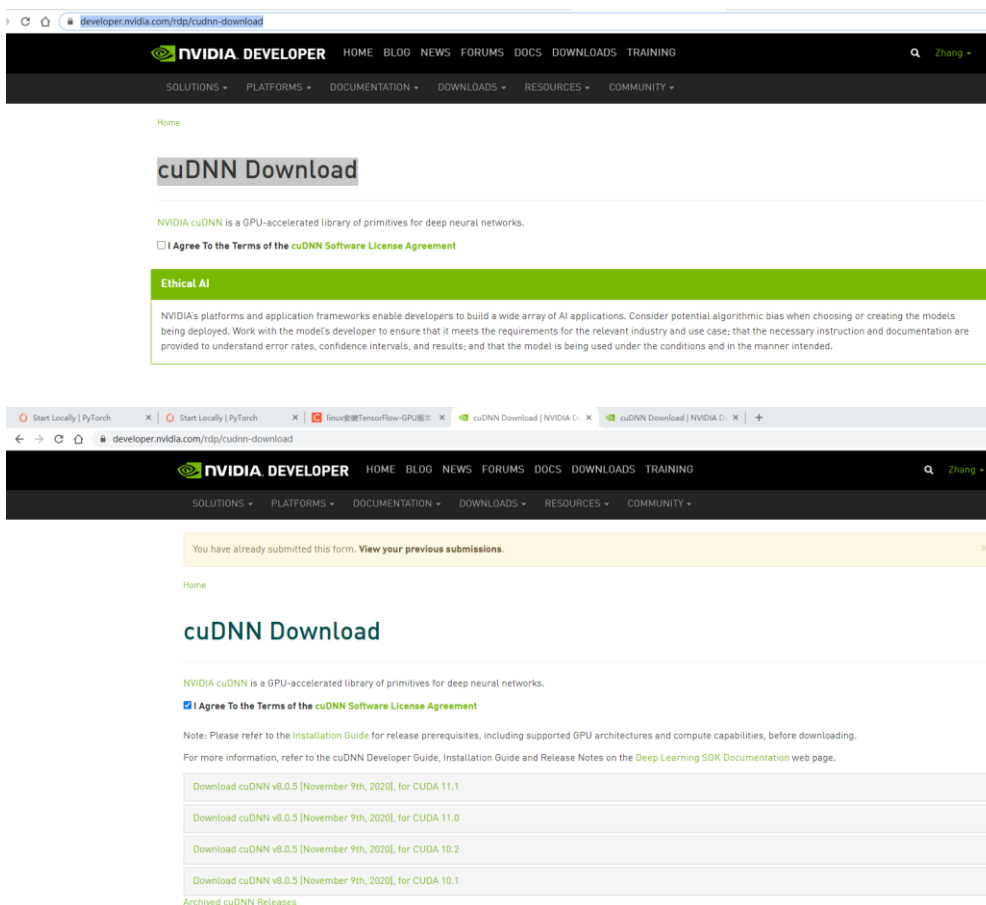
Previous releases of the CUDA Toolkit, GPU Computing SDK, documentation and developer drivers can be found using [below](https://www.nvidia.com/drivers), and be sure to check www.nvidia.com/drivers for more recent production drivers appropriate for your hardware

[Download Latest CUDA Toolkit](#) [Learn More about CUE](#)

Latest Release
CUDA Toolkit 11.1.1 [Oct 2020], [Versioned Online Documentation](#)

Archived Releases
CUDA Toolkit 11.1.0 [Sept 2020], [Versioned Online Documentation](#)
CUDA Toolkit 11.0 Update1 [Aug 2020], [Versioned Online Documentation](#)
CUDA Toolkit 11.0 [May 2020], [Versioned Online Documentation](#)
CUDA Toolkit 10.2 [Nov 2019], [Versioned Online Documentation](#)
CUDA Toolkit 10.1 update2 [Aug 2019], [Versioned Online Documentation](#)
CUDA Toolkit 10.1 update1 [May 2019], [Versioned Online Documentation](#)
CUDA Toolkit 10.1 [Feb 2019], [Online Documentation](#)
CUDA Toolkit 10.0 [Sept 2018], [Online Documentation](#)
CUDA Toolkit 9.2 [May 2018], [Online Documentation](#)
CUDA Toolkit 9.1 [Dec 2017], [Online Documentation](#)
CUDA Toolkit 9.0 [Sept 2017], [Online Documentation](#)
CUDA Toolkit 8.0 GA2 [Feb 2017], [Online Documentation](#)
CUDA Toolkit 8.0 GA1 [Sept 2016], [Online Documentation](#)
CUDA Toolkit 7.5 [Sept 2015]

cuDNN : <https://developer.nvidia.com/rdp/cudnn-download>



The first screenshot shows the NVIDIA Developer website's cuDNN Download page. The browser tabs include 'Start Locally | PyTorch', 'linux安装TensorFlow-GPU版本', and 'cuDNN Download | NVIDIA'. The page header features the NVIDIA Developer logo and navigation links: HOME, BLOG, NEWS, FORUMS, DOCS, DOWNLOADS, and TRAINING. The main heading is 'cuDNN Download'. Below it, a paragraph states that NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks. A checkbox labeled 'I Agree To the Terms of the cuDNN Software License Agreement' is present. An 'Ethical AI' section contains text about responsible AI development. The second screenshot shows the same page after a submission, with a yellow banner stating 'You have already submitted this form. View your previous submissions.' The 'I Agree To the Terms of the cuDNN Software License Agreement' checkbox is now checked. Below the agreement, a note refers to the 'Installation Guide' for release prerequisites. A table lists download links for cuDNN v8.0.5 for various CUDA versions: 11.1, 11.0, 10.2, and 10.1. The table is titled 'Archived cuDNN Releases'.

cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☐ I Agree To the Terms of the [cuDNN Software License Agreement](#)

Ethical AI

NVIDIA's platforms and application frameworks enable developers to build a wide array of AI applications. Consider potential algorithmic bias when choosing or creating the models being deployed. Work with the model's developer to ensure that it meets the requirements for the relevant industry and use case; that the necessary instruction and documentation are provided to understand error rates, confidence intervals, and results; and that the model is being used under the conditions and in the manner intended.

You have already submitted this form. [View your previous submissions.](#)

cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☒ I Agree To the Terms of the [cuDNN Software License Agreement](#)

Note: Please refer to the [Installation Guide](#) for release prerequisites, including supported GPU architectures and compute capabilities, before downloading. For more information, refer to the cuDNN Developer Guide, Installation Guide and Release Notes on the [Deep Learning SDK Documentation](#) web page.

Download cuDNN v8.0.5 [November 9th, 2020], for CUDA 11.1
Download cuDNN v8.0.5 [November 9th, 2020], for CUDA 11.0
Download cuDNN v8.0.5 [November 9th, 2020], for CUDA 10.2
Download cuDNN v8.0.5 [November 9th, 2020], for CUDA 10.1

Archived cuDNN Releases

解压 cuda 对应的 cudnn

Download > ThunderDownload > cuda

名称	修改日期	类型	大小
bin	2020/12/15 16:52	文件夹	
include	2020/12/15 16:51	文件夹	
lib	2020/12/15 16:51	文件夹	
NVIDIA_SL cuDNN_Support	2020/10/30 18:44	文本文档	18 KB

将以上的文件移到你的 cuda 安装目录下

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1

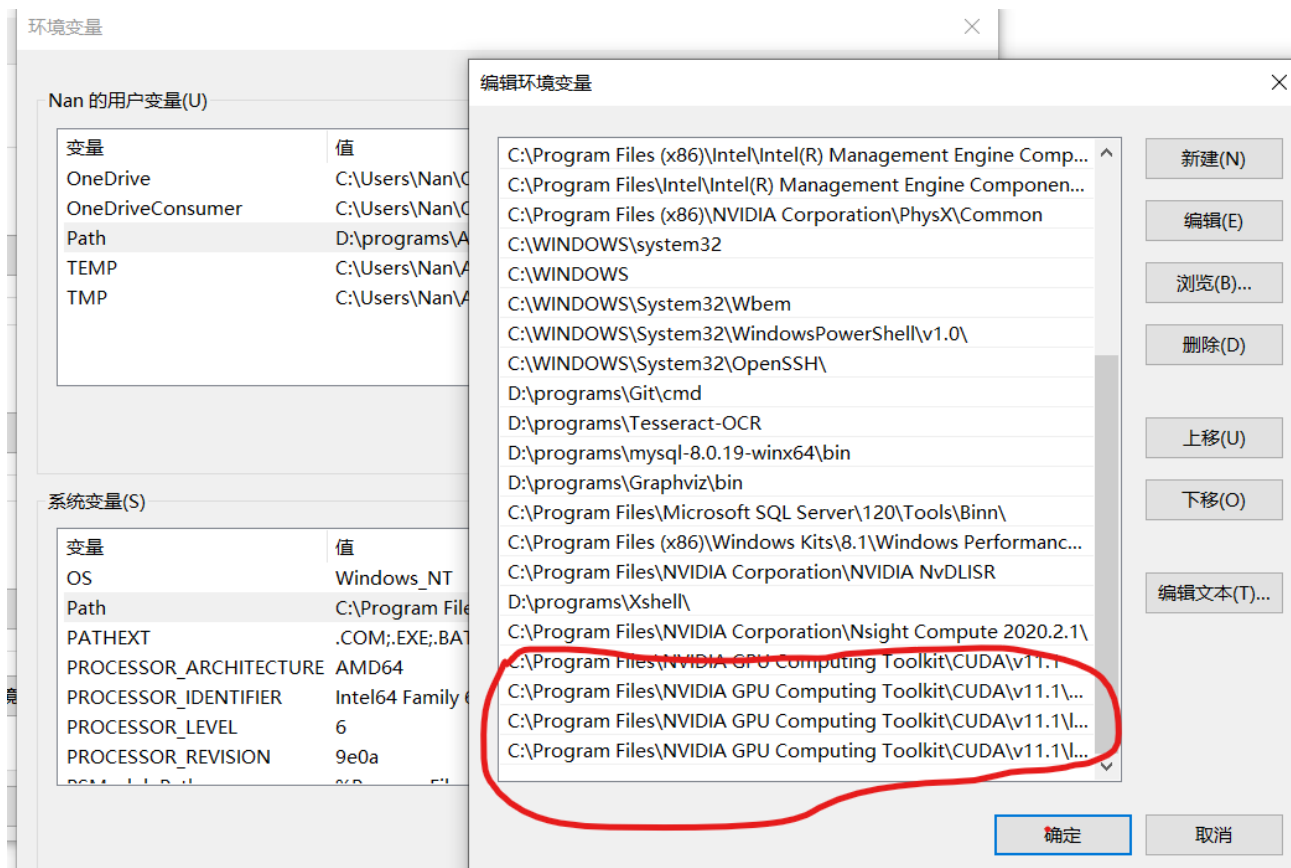
此电脑 > Windows (C:) > Program Files > NVIDIA GPU Computing Toolkit > CUDA > v11.1

名称	修改日期	类型	大小
bin	2020/12/15 16:59	文件夹	
compute-sanitizer	2020/12/15 16:30	文件夹	
extras	2020/12/15 16:30	文件夹	
include	2020/12/15 16:59	文件夹	
lib	2020/12/15 16:30	文件夹	
libnvvp	2020/12/15 16:30	文件夹	
nvml	2020/12/15 16:29	文件夹	
nvvm	2020/12/15 16:29	文件夹	
src	2020/12/15 16:30	文件夹	
tools	2020/12/15 16:30	文件夹	
CUDA_Toolkit_Release_Notes	2020/10/13 15:23	文本文档	16 KB
DOCS	2020/10/13 15:23	文件	1 KB
EULA	2020/10/13 15:23	文本文档	61 KB
NVIDIA_SL cuDNN_Support	2020/10/30 18:44	文本文档	18 KB
README	2020/10/13 15:23	文件	1 KB

添加环境变量

类似的：

- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1
- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\bin
- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\lib\x64
- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.1\libnvvp



查看 GPU 状态

nvidia-smi

查看 cuda 版本

nvcc --version

5.2 TesnsorFlow

官网介绍: <https://www.tensorflow.org/install/pip>

← → ↻ 🏠 tensorflow.google.cn/install/pip

TensorFlow Install Learn API Resources Community Why TensorFlow

Install TensorFlow

Packages

pip

Docker

Additional setup

GPU support

Problems

Build from source

Linux / macOS

Windows

Raspberry Pi

Language bindings

Java 🐘

Java (legacy) 🕒

C

Go

Google is committed to advancing racial equity for Black communities. [See how.](#)

TensorFlow > Install ☆☆☆☆

Install TensorFlow with pip

TensorFlow 2 packages are available

- `tensorflow` —Latest stable release with CPU and [GPU support](#) (*Ubuntu and Windows*)
- `tf-nightly` —Preview build (*unstable*). Ubuntu and Windows include [GPU support](#).

Older versions of TensorFlow

For TensorFlow 1.x, CPU and GPU packages are separate:

- `tensorflow==1.15` —Release for CPU-only
- `tensorflow-gpu==1.15` —Release with [GPU support](#) (*Ubuntu and Windows*)

System requirements

- Python 3.5–3.8
 - Python 3.8 support requires TensorFlow 2.2 or later.
- pip 19.0 or later (requires `manylinux2018` support)
- Ubuntu 16.04 or later (64-bit)
- macOS 10.12.6 (Sierra) or later (64-bit) (*no GPU support*)
- Windows 7 or later (64-bit)
 - [Microsoft Visual C++ Redistributable for Visual Studio 2015, 2017 and 2019](#)
- Raspbian 9.0 or later

对应版本信息查看 <https://tensorflow.google.cn/install/source>

Start Locally | PyTorch x Start Locally | PyTorch x linux安装TensorFlow-GPU版本 x Build from source on Windows x +

← → ↻ 🏠 tensorflow.google.cn/install/source_windows

TensorFlow Install Learn API Resources Community Why TensorFlow

Install TensorFlow

Packages

pip

Docker

Additional setup

GPU support

Problems

Build from source

Linux / macOS

Windows

Raspberry Pi

Language bindings

Java 🐘

Java (legacy) 🕒

C

Go

```
$ export PATH="/c:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v9.0/bin:$PATH"
$ export PATH="/c:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v9.0/extras/CUPTI/libx64:$PATH"
$ export PATH="/c:/tools/cuda/bin:$PATH"
```

Tested build configurations

CPU

Version	Python version	Compiler	Build tools
tensorflow-2.3.0	3.5-3.8	MSVC 2019	Bazel 3.1.0
tensorflow-2.2.0	3.5-3.8	MSVC 2019	Bazel 2.0.0
tensorflow-2.1.0	3.5-3.7	MSVC 2019	Bazel 0.27.1-0.29.1
tensorflow-2.0.0	3.5-3.7	MSVC 2017	Bazel 0.26.1
tensorflow-1.15.0	3.5-3.7	MSVC 2017	Bazel 0.26.1
tensorflow-1.14.0	3.5-3.7	MSVC 2017	Bazel 0.24.1-0.25.2
tensorflow-1.13.0	3.5-3.7	MSVC 2015 update 3	Bazel 0.19.0-0.21.0
tensorflow-1.12.0	3.5-3.6	MSVC 2015 update 3	Bazel 0.15.0
tensorflow-1.11.0	3.5-3.6	MSVC 2015 update 3	Bazel 0.15.0
tensorflow-1.10.0	3.5-3.6	MSVC 2015 update 3	Cmake v3.6.3
tensorflow-1.9.0	3.5-3.6	MSVC 2015 update 3	Cmake v3.6.3
tensorflow-1.8.0	3.5-3.6	MSVC 2015 update 3	Cmake v3.6.3
tensorflow-1.7.0	3.5-3.6	MSVC 2015 update 3	Cmake v3.6.3
tensorflow-1.6.0	3.5-3.6	MSVC 2015 update 3	Cmake v3.6.3

GPU

Version	Python version	Compiler	Build tools	cuDNN	CUDA
tensorflow_gpu-2.3.0	3.5-3.8	MSVC 2019	Bazel 3.1.0	7.6	10.1
tensorflow_gpu-2.2.0	3.5-3.8	MSVC 2019	Bazel 2.0.0	7.6	10.1
tensorflow_gpu-2.1.0	3.5-3.7	MSVC 2019	Bazel 0.27.1-0.29.1	7.6	10.1
tensorflow_gpu-2.0.0	3.5-3.7	MSVC 2017	Bazel 0.26.1	7.4	10
tensorflow_gpu-1.15.0	3.5-3.7	MSVC 2017	Bazel 0.26.1	7.4	10
tensorflow_gpu-1.14.0	3.5-3.7	MSVC 2017	Bazel 0.24.1-0.25.2	7.4	10
tensorflow_gpu-1.13.0	3.5-3.7	MSVC 2015 update 3	Bazel 0.19.0-0.21.0	7.4	10
tensorflow_gpu-1.12.0	3.5-3.6	MSVC 2015 update 3	Bazel 0.15.0	7	9
tensorflow_gpu-1.11.0	3.5-3.6	MSVC 2015 update 3	Bazel 0.15.0	7	9
tensorflow_gpu-1.10.0	3.5-3.6	MSVC 2015 update 3	Cmake v3.6.3	7	9

安装好后可在 python 编译器里查看是否可以运行

```
import tensorflow as tf
```

```
tf.test.is_gpu_available()
```

返回 True 证明成功

5.3 Pytorch

官网介绍: <https://pytorch.org/get-started/locally/>

根据自己的配置选择对用的版本

如果下载慢的话可以直接移步官网用迅雷下载

latest, not fully tested and supported, 1.8 builds that are generated nightly. Please ensure that you have **met the prerequisites below (e.g., numpy)**, depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also [install previous versions of PyTorch](#). Note that LibTorch is only available for C++.

PyTorch Build	Stable (1.7.1)			Preview (Nightly)	
Your OS	Linux		Mac	Windows	
Package	Conda	Pip		LibTorch	Source
Language	Python			C++ / Java	
CUDA	9.2	10.1	10.2	11.0	None
Run this Command:	<pre>pip install torch==1.7.1+cu101 torchvision==0.8.2+cu101 torchaudio==0.7.2 -f https://download.pytorch.org/whl/torch_stable.html</pre>				

例如：

[cu101/torch-1.5.1%2Bcu101-cp36-cp36m-linux_x86_64.whl](#)
[cu101/torch-1.5.1%2Bcu101-cp36-cp36m-win_amd64.whl](#)
[cu101/torch-1.5.1%2Bcu101-cp37-cp37m-linux_x86_64.whl](#)
[cu101/torch-1.5.1%2Bcu101-cp37-cp37m-win_amd64.whl](#)
[cu101/torch-1.5.1%2Bcu101-cp38-cp38-linux_x86_64.whl](#)
[cu101/torch-1.5.1%2Bcu101-cp38-cp38-win_amd64.whl](#)
[cu101/torch-1.6.0%2Bcu101-cp36-cp36m-linux_x86_64.whl](#)
[cu101/torch-1.6.0%2Bcu101-cp36-cp36m-win_amd64.whl](#)
[cu101/torch-1.6.0%2Bcu101-cp37-cp37m-linux_x86_64.whl](#)
[cu101/torch-1.6.0%2Bcu101-cp37-cp37m-win_amd64.whl](#)
[cu101/torch-1.6.0%2Bcu101-cp38-cp38-linux_x86_64.whl](#)
[cu101/torch-1.6.0%2Bcu101-cp38-cp38-win_amd64.whl](#)
[cu101/torch-1.7.0%2Bcu101-cp36-cp36m-linux_x86_64.whl](#)
[cu101/torch-1.7.0%2Bcu101-cp36-cp36m-win_amd64.whl](#)
[cu101/torch-1.7.0%2Bcu101-cp37-cp37m-linux_x86_64.whl](#)
[cu101/torch-1.7.0%2Bcu101-cp37-cp37m-win_amd64.whl](#)
[cu101/torch-1.7.0%2Bcu101-cp38-cp38-linux_x86_64.whl](#)
[cu101/torch-1.7.0%2Bcu101-cp38-cp38-win_amd64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp36-cp36m-linux_x86_64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp36-cp36m-win_amd64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp37-cp37m-linux_x86_64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp37-cp37m-win_amd64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp38-cp38-linux_x86_64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp38-cp38-win_amd64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp39-cp39-linux_x86_64.whl](#)
[cu101/torch-1.7.1%2Bcu101-cp39-cp39-win_amd64.whl](#)
[cu101/torchcsprng-0.1.0-cp36-cp36m-linux_x86_64.whl](#)



安装后 python 编译器输入

```
import torch
```

```
print(torch.cuda.is_available())
```

返回 True 证明成功