

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 Windwos5

1.2 Linux5

1.2.1 yum 更新 yum 源 5

1.2.2 安装 Python6

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 11

3.1 conda 的安装 11

3.1.1 换国内源 12

3.1.2 临时使用镜像 13

3.1.3 版本导致的问题 13

3.2 Anaconda 的使用 14

3.2.1 Anaconda 安装后开始菜单配置问题 14

- 3.2.2 管理包 14
- 3.2.3 管理环境 14
- 3.2.4 共享环境 15
- 3.2.5 jupyter notebook 的配置 16
- 3.2.6 matplotlib 中文字体配置 18

4. pip 的安装使用 19

4.1 pip 的安装 19

- 4.1.1 windows 下 19
- 4.1.2 Linux 下 20
- 4.1.3 安装包时临时配置 20

4.2 pip 的使用 21

- 4.2.1 安装包 21
- 4.2.2 查看所有版本 21
- 4.2.3 指定版本 21
- 4.2.4 卸载包 21
- 4.2.5 查看安装包信息 21
- 4.2.6 列出所有安装包 21
- 4.2.7 检测更新 21
- 4.2.8 更新包 22
- 4.2.9 环境导出导入 22

5. TensorFlow&Pytorch GPU 版本安装 22

5.1 TesnsorFlow38

5.2 Pytorch40

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 3.7.9 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 3.7.7 版本

wget <https://www.python.org/ftp/python/3.7.9/Python-3.7.9.tgz>

✧ 国内镜像源下载

wget <https://npm.taobao.org/mirrors/python/3.7.9/Python-3.7.9.tgz>

1.2.2 安装 Python

✧ 解压

```
tar -xvf Python-3.7.9.tgz
```

```
rm Python-3.7.9.tgz
```

✧ 配置编译

```
cd Python-3.7.9
```

✧ 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

具体网址根据版本自行选择

2.2.2 安装

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

新版的安装在一路 yes 最后是更改安装地址，默认是

```
Prefix= /root/anaconda3
```

可以自定义

```
/root/anaconda3>>>/home/anaconda3
```

最后一步有一个 init 初始化，选择 yes 的话可以自动 conda 命令到环境变量

默认是添加在.bashrc

手动激活

```
source ~/.bashrc
```

2.2.3 配置环境变量

root 用户下

```
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
```

或

```
vim ~/.bash_profile
```

在最后一行加上

```
export PATH=$PATH:/home/nan/Anaconda3
```

激活

```
source ~/.bash_profile
```

禁止自动进入 base 环境

```
conda config --set auto_activate_base false
```

3. Miniconda 的安装和下载

Miniconda 是 Anaconda 的简化版，只有部分基础功能用户可根据自己的需求安装所需的包。

官网下载地址：

<https://docs.conda.io/en/latest/miniconda.html>

官网下载慢的同学可移步到清华镜像源：

<https://mirrors.tuna.tsinghua.edu.cn/anaconda/miniconda/?C=M&O=D>

File Name ↓	File Size ↓	Date ↓
Parent directory/	-	-
Miniconda3-py39_4.9.2-Windows-x86_64.exe	57.7 MiB	2020-12-22 01:10
Miniconda3-py39_4.9.2-MacOSX-x86_64.pkg	49.7 MiB	2020-12-22 01:10
Miniconda3-py39_4.9.2-MacOSX-x86_64.sh	42.2 MiB	2020-12-22 01:10
Miniconda3-py39_4.9.2-Windows-x86.exe	54.9 MiB	2020-12-22 01:10
Miniconda3-py39_4.9.2-Linux-ppc64le.sh	60.3 MiB	2020-12-22 01:10
Miniconda3-py39_4.9.2-Linux-x86_64.sh	58.6 MiB	2020-12-22 01:10
Miniconda3-py38_4.9.2-Windows-x86_64.exe	57.0 MiB	2020-11-24 03:22
Miniconda3-py38_4.9.2-Windows-x86.exe	54.2 MiB	2020-11-24 03:22
Miniconda3-py38_4.9.2-MacOSX-x86_64.sh	54.5 MiB	2020-11-24 03:22
Miniconda3-py38_4.9.2-MacOSX-x86_64.pkg	62.0 MiB	2020-11-24 03:21
Miniconda3-py38_4.9.2-Linux-x86_64.sh	89.9 MiB	2020-11-24 03:21
Miniconda3-py38_4.9.2-Linux-ppc64le.sh	91.9 MiB	2020-11-24 03:21
Miniconda3-py37_4.9.2-Windows-x86_64.exe	55.8 MiB	2020-11-24 03:21
Miniconda3-py37_4.9.2-Windows-x86.exe	52.9 MiB	2020-11-24 03:21

3.1 3.2 Linux 下安装:

3.1.1 下载

wget https://mirrors.tuna.tsinghua.edu.cn/anaconda/miniconda/Miniconda3-py39_4.10.3-Linux-x86_64.sh

具体网址根据版本自行选择

3.1.2 安装

```
bash Miniconda3-py39_4.10.3-Linux-x86_64.sh
```

取消自动进入 base 环境

```
conda config --set auto_activate_base false
```

环境变量会自动添加到 ~/.bashrc

可以手动激活 source ~/.bashrc 或者进入 bash 命令

```
conda env list
```

测试命令

3.1.3 Docker 安装 miniconda

miniconda 体谅下很适合部署在 docker 容器中

官网参考

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

搜索镜像

`docker search continuumio`

拉取镜像

`docker pull continuumio/miniconda3`

以后台方式启动镜像创建容器

```
docker run -itd --name="anaconda3_jupyter" -p 8888:8888 continuumio/miniconda3  
/bin/bash
```

如果使用 gpu

```
docker run --gpus all -itd --name="anaconda3_jupyter" -p 8888:8888 docker pull  
gpuci/miniconda-cuda:11.0-devel-centos7 /bin/bash
```

4. Conda 配置使用

4.1 conda 的初始化

root 权限下

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

4.1.1 换国内源

✧ 查看源: conda config --show-sources

```
==> C:\Users\wh19012\.condarc <==
channels:
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/free/
  - defaults
show_channel_urls: True
```

✧ 添加清华源:

conda config --add channels <http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/free/>

conda config --add channels <http://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/main/>

移除默认源

conda config --remove channels defaults

```
1 #添加清华的源
2 conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgsg/free/
3
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-key channels

或

conda config --remove channels

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

4.1.2 临时使用镜像

conda install -c 镜像源 包名

例如：

conda install -c <https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgsg/free/> pandas

4.1.3 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/>

4.2 Anaconda 的使用

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 ipython-  
notebook ipython-qtconsole launcher sp
```

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

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
```

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
```

4.2.1 jupyter notebook 的安装配置

Anaconda3 base 环境自带 jupyter 和 spyder，如果安装的是 Miniconda 需要自行安装配置

conda 环境中安装 jupyter noterbook

```
pip install jupyter notebook
```

安装环境自动关联包

让 jupyter 可以切换多个 python 环境

```
conda install nb_conda
```

如果镜像源找不到最新本本移步官网

https://anaconda.org/conda-forge/nb_conda

```
conda install -c conda-forge nb_conda
```

https://anaconda.org/anaconda/nb_conda

```
conda install -c anaconda nb_conda
```

进入虚拟环境

```
conda activate python_env
```

给虚拟环境安装 ipykernel

```
pip install ipykernel
```


进入 notebook

jupyter notebook

```
[nbextensions_configurator] (0.2.0)
Installing collected packages: pyyaml, jupyter-contrib-core, jupyter-nbextensions-configurator
Successfully installed jupyter-contrib-core-0.3.3 jupyter-nbextensions-configurator-0.4.1 pyyaml-5.4.1

C:\Users\Nan>jupyter notebook
[I 20:07:03.962 NotebookApp] [nb_conda_kernels] enabled, 3 kernels found
[I 20:07:06.570 NotebookApp] [jupyter_nbextensions_configurator] enabled 0.4.1
[I 20:07:06.767 NotebookApp] [nb_conda] enabled
[I 20:07:07.031 NotebookApp] Serving notebooks from local directory: C:\Users\Nan
[I 20:07:07.031 NotebookApp] Jupyter Notebook 6.2.0 is running at:
[I 20:07:07.032 NotebookApp] http://localhost:8888/?token=65a4e1b1deccb65090bf7ee79e545f8b8d7d135044ac5497
[I 20:07:07.032 NotebookApp] or http://127.0.0.1:8888/?token=65a4e1b1deccb65090bf7ee79e545f8b8d7d135044ac5497
[I 20:07:07.032 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 20:07:07.199 NotebookApp]
```

jupyter notebook 中使用 Tab 键补全代码

为 jupyter notebook 添加目录

conda install -c conda-forge jupyter_contrib_nbextensions

或

pip install jupyter_nbextensions_configurator

方法：运行 Jupyter Notebook, 在打开的 Notebook 界面里, 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

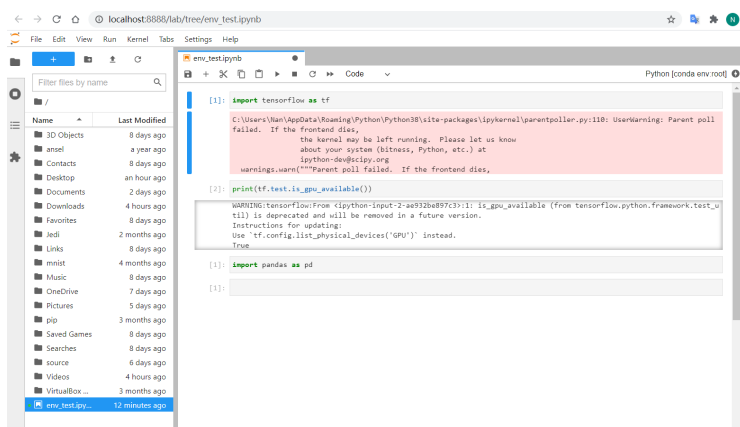
4.2.2 jupyter lab 的安装

jupyter lab 可实现终端控制

pip install jupyterlab

启动

Jupyter lab



4.2.3 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

5. pip 的安装使用

5.1 pip 的安装

5.1.1 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

pip config set global.ssl_verify false

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
```

```
pip config set global.index-url https://pypi.org/simple
```

```
pip config set install.trusted-host pypi.org/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.ini）
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

5.1.2 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]
```

```
index-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
```

5.1.3 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>

5.2 pip 的使用

5.2.1 安装包

```
pip install package_name
```

5.2.2 查看所有版本

```
pip install package_name==
```

5.2.3 指定版本

```
pip install package_name ==1.0.4
```

5.2.4 卸载包

```
pip uninstall package_name
```

5.2.5 查看安装包信息

```
pip show package_name
```

5.2.6 列出所有安装包

```
pip list
```

5.2.7 检测更新

```
pip list -outdated
```

5.2.8 更新包

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

5.2.9 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
```

在 conda 环境中使用 `pip freeze > requirements.txt` 命令导出已安装的模块，其中部分模块显示了 `@ file:///...`，而不是具体的版本号如下图

```
soupsieve==2.2
tokenizers==0.10.1
torch @ file:///D:/Downloads/torch-1.8.0%2Bcu111-cp37-cp37m-win_amd64.whl
torchaudio @ file:///D:/Downloads/torchaudio-0.8.0-cp37-none-win_amd64.whl
torchvision @ file:///D:/Downloads/torchvision-0.9.0%2Bcu111-cp37-cp37m-win_amd64.whl
tornado==6.1
tqdm==4.59.0
traitlets==5.0.5
```

当你遇到此类问题时，可以暂时考虑使用如下命令生成 requirements.txt 文件

```
pip list --format=freeze > requirements.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
```

6. TensorFlow&Pytorch 安装

6.1 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

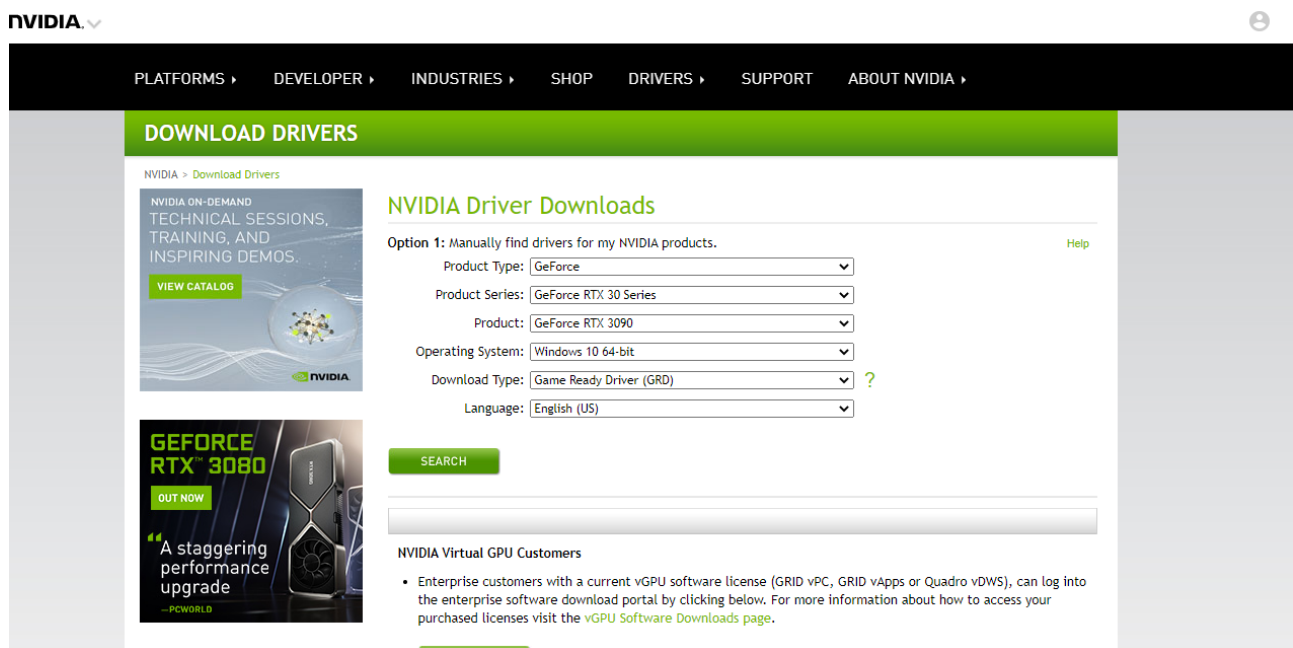
另外要注意 1.0 版本和 2.0 版本支持的 cuda 版本也不同


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

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

6.1.1 windows

驱动



NVIDIA 

PLATFORMS ▾ DEVELOPER ▾ INDUSTRIES ▾ SHOP DRIVERS ▾ SUPPORT ABOUT NVIDIA ▾

DOWNLOAD DRIVERS

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->

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

DEVELOPER ZONE CUDA TOOLKIT DOCUMENTATION

CUDA 11.1.1 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-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。

notes/index.html

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 Sur
- 显示
 - 更改分辨率
 - 调整桌面
 - 旋转显示
 - 查看 HDC
 - 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
CeForce 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: 4096 MB GDDR5

关于(A)

保存(S) 关闭

CUDA 安装

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 has navigation links: HOME, BLOG, NEWS, FORUMS, DOCS, DOWNLOADS, TRAINING. The main heading is 'CUDA Toolkit Archive'. Below it, a paragraph states: 'Previous releases of the CUDA Toolkit, GPU Computing SDK, documentation and developer drivers can be found using below, and be sure to check www.nvidia.com/drivers for more recent production drivers appropriate for your hardware'. There are two links: 'Download Latest CUDA Toolkit' and 'Learn More about CUC'. Under 'Latest Release', it says 'CUDA Toolkit 11.1.1 [Oct 2020], Versioned Online Documentation'. Under 'Archived Releases', it lists various versions from 11.1.0 down to 7.5, each with a link to 'Versioned Online Documentation' or 'Online Documentation'.

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

cuDNN 可能得通过科学上网不然速度实在慢的惊人

The screenshot shows the NVIDIA Developer website's cuDNN Download page. The browser tab is 'developer.nvidia.com/rdp/cudnn-download'. The page header has navigation links: HOME, BLOG, NEWS, FORUMS, DOCS, DOWNLOADS, TRAINING, and a search bar with 'Zhang'. Below the header, there's a 'Home' link and a 'cuDNN Download' heading. A paragraph states: 'NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.' Below this is a checkbox labeled 'I Agree To the Terms of the cuDNN Software License Agreement'. At the bottom, there's a green box titled 'Ethical AI' with text: '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.'

cuDNN 需要注册账号登录

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 v8.1.0 (January 26th, 2021), for CUDA 11.0,11.1 and 11.2
- Download cuDNN v8.1.0 (January 26th, 2021), for CUDA 10.2

[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_A_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_A_cuDNN_Support	2020/10/30 18:44	文本文档	18 KB
README	2020/10/13 15:23	文件	1 KB

6.1.2 Linux

安装 NVIDIA 显卡驱动

Centos 为例 ubuntu GUI 软件商店一般就能搜到

➤ 查看系统内核版本

```
uname -r
```

➤ 查看显卡列信息

```
lspci| grep -i vga
```

➤ 安装依赖

```
sudo yum install gcc dkms
```

```
sudo yum install kernel-devel
```

```
yum install dnf
```

```
dnf groupinstall "Development Tools"
```

```
dnf install libglvnd-devel elfutils-libelf-devel
```

安装完成后，执行

```
rpm -qa|grep gcc
```

```
rpm -qa|grep kernel
```

检查安装版本，这里可能遇到的情况有 kernel 存在两个版本，这时候要卸载一个，确保存在的 kernel 与 kernel-devel 和 kernel-header 包的版本号一致

卸载命令(不检查依赖关系直接删除)

例如

```
rpm -e --nodeps kernel-3.10.0-1160.el7.x86_64
```

➤ 屏蔽 nouveau 驱动

```
vim /etc/modprobe.d/nvidia-installer-disable-nouveau.conf
```

➤ 添加

blacklist nouveau

options nouveau modeset=0

保存退出

vim /lib/modprobe.d/nvidia-installer-disable-nouveau.conf

➤ 添加

blacklist nouveau

options nouveau modeset=0

如果不手动添加安装过程可能会提示自动添加，然后重启生效

备份 initramfs 镜像

mv /boot/initramfs-\$(uname -r).img /boot/initramfs-\$(uname -r).img.bak

➤ 建立新的镜像

dracut /boot/initramfs-\$(uname -r).img \$(uname -r)

➤ 查看 nouveau 有没有被禁用

系统默认安装 nouveau kernel driver, 与 NVIDIA 驱动冲突，所以要先检查其是否被禁用，

执行命令

lsmod | grep nouveau

有输出信息说明没有被禁用，禁用方法如下

➤ 改为终端模式重启

systemctl set-default multi-user.target

init 3

reboot

➤ 安装

chmod +x ./NVIDIA-Linux-x86_64-460.32.03.run

```
./NVIDIA-Linux-x86_64-460.32.03.run -no-x-check -no-nouveau-check -no-opengl-  
files
```

-no-opengl-files 只安装驱动文件，不安装 OpenGL 文件。这个参数最重要

-no-x-check 安装驱动时不检查 X 服务

-no-nouveau-check 安装驱动时不检查 nouveau

如果安装后再进入图形界面显示器不亮可尝试在图形界面中安装

➤ 查看 GPU 状态

```
nvidia-smi
```

Ubuntu

```
sudo apt-add-repository -r ppa:graphics-drivers/ppa
```

```
sudo apt remove nvidia*
```

```
sudo apt autoremove
```

```
sudo ubuntu-drivers autoinstall
```

如果仍然对保留的软件包有问题，可能是因为在启用 PPA 的同时更新了一些软件包，在这种情况下，运行

```
sudo apt install aptitude
```

```
sudo aptitude install <name_of_package_with_conflicts>
```

cuda

Please Note: We advise customers updating to Linux Kernel 5.9+ to use the latest NVIDIA Linux GPU driver R455 that will be available for download from NVIDIA website and repositories, starting today.

Select Target Platform

Click on the green buttons that describe your target platform. Only supported platforms will be shown. By downloading and using the software, you agree to fully comply with the terms and conditions of the [CUDA EULA](#).

Operating System

Linux Windows

Architecture

x86_64 ppc64le sbsa

Distribution

CentOS Debian Fedora OpenSUSE RHEL SLES Ubuntu WSL-Ubuntu

Version

8 7

Download Installer for Linux CentOS 7 x86_64

The base installer is available for download below.

Base Installer

Installation Instructions:

```
$ wget https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda_11.0.3_450.51.06_linux.run
$ sudo sh cuda_11.0.3_450.51.06_linux.run
```

The CUDA Toolkit contains Open-Source Software. The source code can be found [here](#).
The checksums for the installer and patches can be found in [Installer Checksums](#).
For further information, see the [Installation Guide for Linux](#) and the [CUDA Quick Start Guide](#).

[Documentation >](#)

[Release Notes >](#)

[MacOS Tools >](#)

[Code Samples >](#)

[Legacy Releases >](#)

安装

```
chmod +x sh cuda_11.0.3_450.51.06_linux.run
```

```
sudo sh cuda_11.0.3_450.51.06_linux.run
```

安装过程只选择安装 cuda 即可

安装完成后会有提示添加环境变量

```
vim ~/.bashrc
```

```
export PATH=$PATH:/usr/local/cuda-11.0/bin
```

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda-11.0/lib64
```

```
source ~/.bashrc
```

cuDNN

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

Library for Windows and Linux, Ubuntu(x86_64 & PPC architecture)

[cuDNN Library for Linux \(x86_64\)](#)

[cuDNN Library for Linux \(PPC\)](#)

[cuDNN Library for Windows \(x86\)](#)

[cuDNN Runtime Library for Ubuntu20.04 x86_64 \(Deb\)](#)

[cuDNN Developer Library for Ubuntu20.04 x86_64 \(Deb\)](#)

[cuDNN Code Samples and User Guide for Ubuntu20.04 x86_64 \(Deb\)](#)

[cuDNN Runtime Library for Ubuntu18.04 x86_64 \(Deb\)](#)

[cuDNN Developer Library for Ubuntu18.04 x86_64 \(Deb\)](#)

[cuDNN Code Samples and User Guide for Ubuntu18.04 x86_64 \(Deb\)](#)

[cuDNN Runtime Library for Ubuntu16.04 x86_64 \(Deb\)](#)

[cuDNN Developer Library for Ubuntu16.04 x86_64 \(Deb\)](#)

[cuDNN Code Samples and User Guide for Ubuntu16.04 x86_64 \(Deb\)](#)

解压

```
tar -xzf cudnn-11.0-linux-x64-v8.0.5.39.tgz
```

复制到 cuda 的安装目录

```
sudo cp cuda/include/cudnn.h /usr/local/cuda/include
```

```
sudo cp cuda/lib64/libcudnn* /usr/local/cuda/lib64
```

```
sudo chmod a+r /usr/local/cuda/include/cudnn.h /usr/local/cuda/lib64/libcudnn*
```

如果运行 tensorflow 缺少上个版本的组件（例如 libcusolver.so.10），请自行下载 copy 到

cuda-11.1/lib64 目录

卸载

在 cuda 的安装目录如 /usr/local/cuda-11.1/bin 运行以下命令

```
cuda-uninstaller
```

ubuntu

➤ 安装


```
wget https://mirrors.aliyun.com/nvidia-cuda/ubuntu2004/x86_64/cuda-ubuntu2004.pin
sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600

sudo apt-key adv --fetch-keys https://mirrors.aliyun.com/nvidia-cuda/ubuntu2004/x86_64/7fa2af80.pub

sudo add-apt-repository "deb https://mirrors.aliyun.com/nvidia-cuda/ubuntu2004/x86_64/ /"

sudo apt-get update

sudo apt-get -y install cuda-11-1
```

- 卸载

```
sudo apt-get remove --auto-remove nvidia-cuda-toolkit

sudo apt-get purge --auto-remove nvidia-cuda-toolkit
```

- Uninstall just nvidia-cuda-toolkit

```
sudo apt-get remove nvidia-cuda-toolkit
```

- Uninstall nvidia-cuda-toolkit and it's dependencies

```
sudo apt-get remove --auto-remove nvidia-cuda-toolkit
```

- Purging config/data

```
sudo apt-get purge nvidia-cuda-toolkit or sudo apt-get purge --auto-remove nvidia-cuda-toolkit
```

cuDNN 与 centos 类似

6.2 Nvidia-docker 安装

6.2.1 Docker 安装

Linux 中建议用 docker 构建 cuda 环境，这是比较快捷和有效的方法

官 网 地 址：<https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html#docker>

设置稳定存储库 yum 源为阿里 docker 源

```
yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
```

查看已安装的 docker 列表

```
yum list installed |grep docke
```

删除已安装的 docker

```
yum -y remove docker-ce.x86_64
```

使用 Docker Engineshequ 社区的存储库安装 docker。

在新主机上首次安装 Docker Engine-Community 之前，需要设置 Docker 仓库。之后，您可以从仓库安装和更新 Docker。

设置仓库

安装所需的软件包。

yum-utils 提供了 yum-config-manager，并且 device mapper 存储驱动程序需要 device-mapper-persistent-data 和 lvm2。

```
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
```

设置稳定存储库 yum 源为阿里 docker 源

```
yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
```

安装 Docker Engine-Community

➤ 安装最新版本的 Docker Engine-Community 和 containerd，或者转到下一步安装特定版

本:

```
sudo yum install docker-ce docker-ce-cli containerd.io
```

如果提示您接受 GPG 密钥，请选是

Docker 安装完默认未启动。并且已经创建好 docker 用户组，但该用户组下没有用户。

要安装特定版本的 Docker Engine-Community，请在存储库中列出可用版本，然后选择并安装：

列出并排序您存储库中可用的版本。此示例按版本号（从高到低）对结果进行排序。

```
yum list docker-ce --showduplicates | sort -r
```

docker-ce.x86_64 3:18.09.1-3.el7	docker-ce-stable
docker-ce.x86_64 3:18.09.0-3.el7	docker-ce-stable
docker-ce.x86_64 18.06.1.ce-3.el7	docker-ce-stable
docker-ce.x86_64 18.06.0.ce-3.el7	docker-ce-stable

通过其完整的软件包名称安装特定版本，该软件包名称是软件包名称（docker-ce）加上版本字符串（第二列），从第一个冒号（:）一直到第一个连字符，并用连字符（-）分隔。例如：docker-ce-18.09.1。

验证安装是否成功(有 client 和 service 两部分表示 docker 安装启动都成功了)

```
...  
docker version  
...
```

启动并加入开机启动

```
sudo systemctl enable docker.service
```

➤ docker 启动

```
sudo systemctl start docker
```

➤ 重启 docker

```
sudo systemctl restart docker
```

6.2.2 设置 NVIDIA Container Toolkit

以下以 centos7 为示例，其他版本可参照官网

设置稳定的存储库和 GPG 密钥

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
```

```
&& curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.repo | sudo  
tee /etc/yum.repos.d/nvidia-docker.repo
```

更新软件包清单后，安装 nvidia-docker2 软件包（和依赖项）

```
sudo yum clean expire-cache
```

```
sudo yum install -y nvidia-docker2
```

设置默认运行时后，重新启动 Docker 守护程序以完成安装：

```
sudo systemctl restart docker
```

可以通过运行基本 CUDA 容器来测试有效的设置：

```
sudo docker run --rm --gpus all nvidia/cuda:11.0-base nvidia-smi
```

这应该导致控制台输出如下所示

```

+-----+
| NVIDIA-SMI 450.51.06      Driver Version: 450.51.06    CUDA Version: 11.0     |
+-----+-----+-----+
| GPU  Name          Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|                                       MIG M. |
+-----+-----+-----+
|   0   Tesla T4              On   | 00000000:00:1E.0 Off |           0         |
| N/A   34C    P8             9W / 70W |  0MiB / 15109MiB |      0%    Default  |
|                                       N/A         |
+-----+-----+-----+

+-----+
| Processes: |
| GPU  GI    CI          PID    Type    Process name          GPU Memory |
| ID   ID   ID                          Process name          Usage       |
+-----+-----+-----+
| No running processes found |
+-----+

```

6.3 Anaconda 管理 cuda 环境

用 conda 管理 cuda 是最简单快捷的，但是往往最新版的 cuda 还是的自己官网下载安装

进入虚拟环境

conda activate env

搜索 cuda 版本

conda search cudatoolkit -c conda-forge

选择需要的版本安装

conda install -c conda-forge cudatoolkit=11.0

搜索 cuDNN

conda search cuDNN -c conda-forge

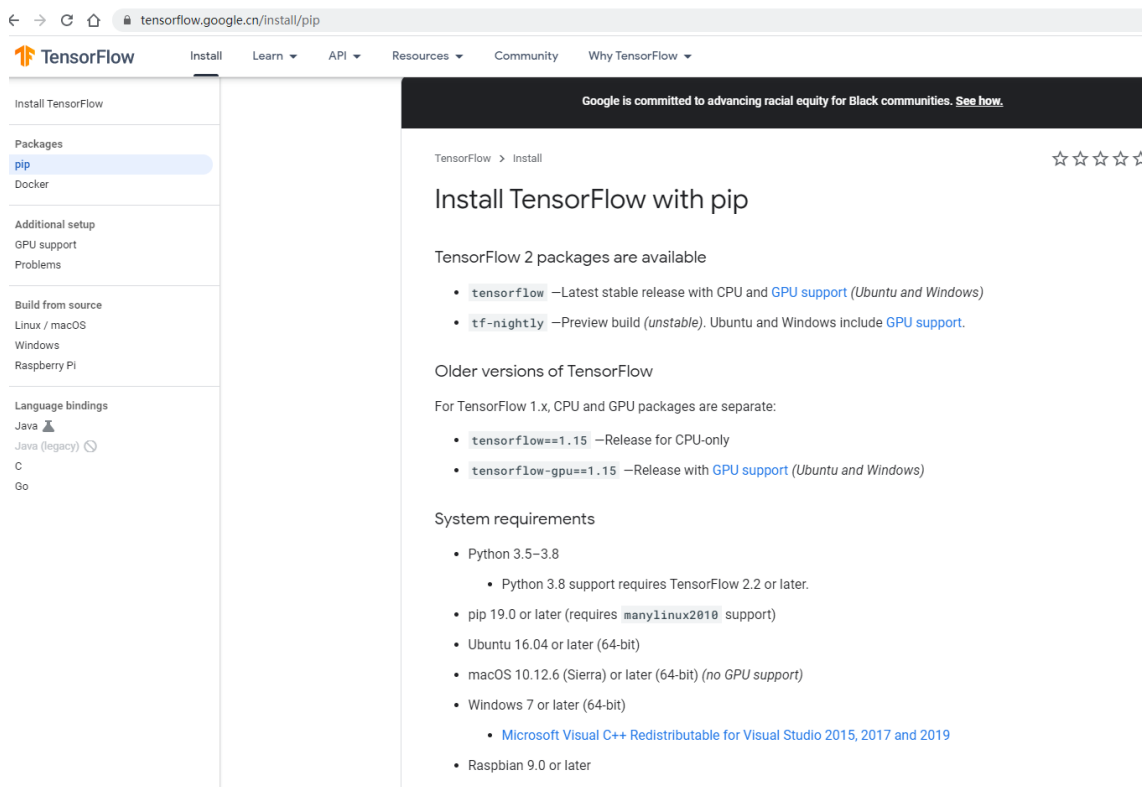
选择需要的版本安装

conda install -c conda-forge cuDNN=8.0

如果直接安装 cuDNN 会自动安装对应的 cuda 最高版本，这时可能出现不是对应的 tensorflow 支持版本，所以按照 tensorflow 对应版本先装 cuda。

6.4 TesnsorFlow

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



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

Start Locally | PyTorch

Start Locally | PyTorch

linux安装TensorFlow-GPU版本

Build from source on Windows

tensorflow.google.cn/install/source_windows

TensorFlow

Install

Learn

API

Resources

Community

Why TensorFlow

Install TensorFlow

Packages

Additional setup

Build from source

Language bindings

pip

Docker

GPU support

Problems

Linux / macOS

Windows

Raspberry Pi

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

6.5 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	Open link in new window
cu101/torch-1.6.0%2Bcu101-cp38-cp38-win_amd64.whl	Open link in new window
cu101/torch-1.7.0%2Bcu101-cp36-cp36m-linux_x86_64.whl	Open link in incognito window
cu101/torch-1.7.0%2Bcu101-cp37-cp37m-linux_x86_64.whl	Send link to iPhone
cu101/torch-1.7.0%2Bcu101-cp38-cp38-linux_x86_64.whl	Save link as...
cu101/torch-1.7.1%2Bcu101-cp36-cp36m-linux_x86_64.whl	Copy link address
cu101/torch-1.7.1%2Bcu101-cp36-cp36m-win_amd64.whl	Use迅雷下载
cu101/torch-1.7.1%2Bcu101-cp37-cp37m-linux_x86_64.whl	Inspect
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	

- Open link in new tab
- Open link in new window
- Open link in incognito window
- Send link to iPhone 8
- Save link as...
- Copy link address
- 使用迅雷下载
- Inspect Ctrl+Shift+I

安装后 python 编译器输入


```
import torch
```

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

返回 True 证明成功