**Yahboom Jetson Nano B01 Sub – Ubuntu 20.04**

**Jetpack 4.6.5, jtop, TensorRT 8.2.6, VPI,**

**opencv 4.5, pytorch 1.13, torchvision 0.14.0, onnxruntime-gpu 설치 설명서**

작성자: 정도윤([rabbitsun2@gmail.com](mailto:rabbitsun2@gmail.com))

작성일자: 2024-09-13(금)

**목차**

**1. Ubuntu 20.04에서 GCC, G++ 9버전에서 8버전으로 하향하기(CUDA 10.2) ………………… 3**

**2. Ubuntu 20.04에서 NVIDIA jetson APT 주석 풀어주기 ……………………………………… 4**

**3. Ubuntu 20.04에서 NVIDIA Jetpack 4.6.5 설치하기(Ubuntu 20.04, Jetson Nano만 가능) … 5**

**4. Ubuntu 20.04에서 .bashrc 설치하기 ……………………………………………………………… 6**

**5. Ubuntu 20.04에서 TensorRT 설치하기 ……………………………………………………………… 7**

**6. Ubuntu 20.04에서 NVIDIA VPI 설치하기 ………………………………………………………… 9**

**7. Ubuntu 20.04에서 OpenCV 4.5(opencv-python 4.5) 수동 설치 ……………………… 11**

**8. Ubuntu 20.04에서 Pytorch 1.13 설치하기(Q-engineeering) …………………………… 19**

**9. Ubuntu 20.04에서 torchvision 설치하기(수동 컴파일) ……………………………………… 21**

**10. Ubuntu 20.04에서 onnxruntime-gpu 설치하기 ……………………………………………… 22**

**1. Ubuntu 20.04에서 GCC, G++ 9버전에서 8버전으로 하향하기(CUDA 10.2)**

# install gcc and g++ version 8

$ sudo apt-get install gcc-8 g++-8

# setup the gcc selector

$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 9

$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-8 8

# setup the g++ selector

$ sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-9 9

$ sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-8 8

# if you want to make a selection use these commands

$ sudo update-alternatives --config gcc

$ sudo update-alternatives --config g++

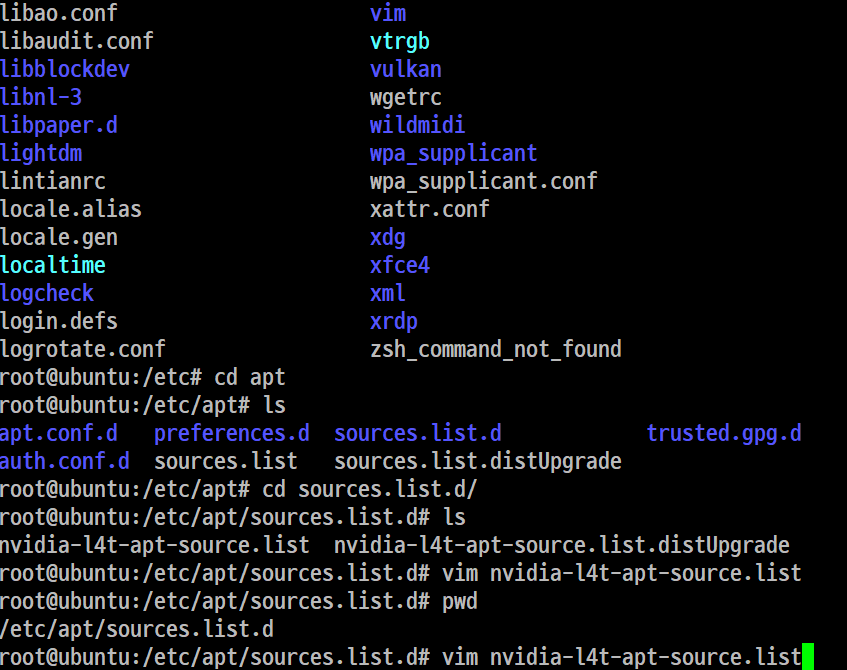
gcc, g++을 8버전으로 바꿔야 CUDA 10.2에서 사용할 수 있음.

**2. Ubuntu 20.04에서 NVIDIA Jetson APT 주석 풀어주기**

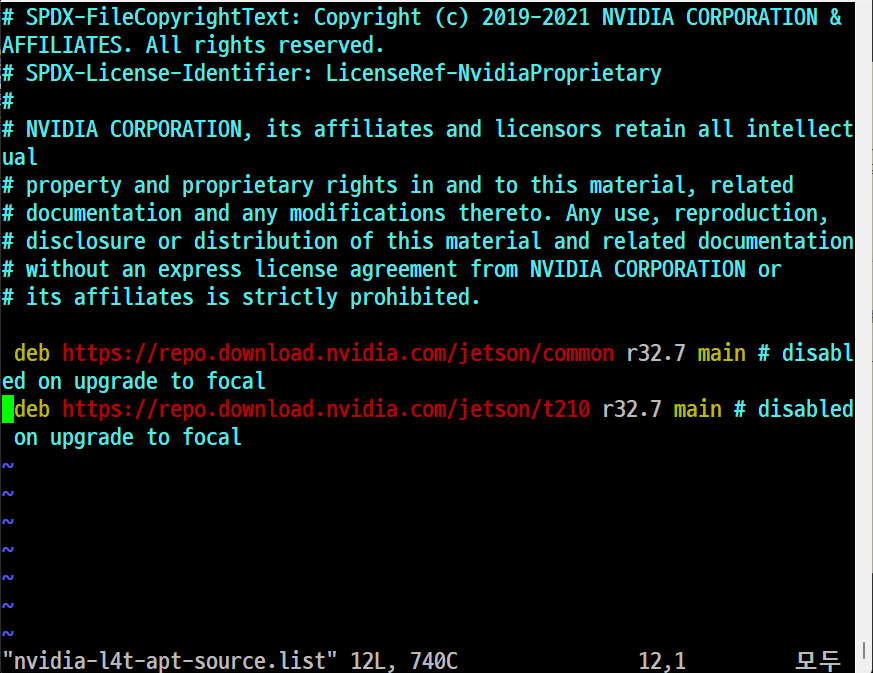
jetpack 4.5를 비롯한 vpi를 설치하려면 NVIDIA Jetson common APT의 주석을 풀어주어야 한다.

cd /etc/apt/sources.list.d

vim nvidia-l4t-apt-source.list



**그림 1.** nvidia-l4t-apt-source.list 파일 수정하기



**그림 2.** nvidia-l4t-apt-source.list 파일 수정하기

deb 주석을 두 개를 풀어준다.

**3. Ubuntu 20.04에서 NVIDIA Jetpack 4.6.5 설치하기(Ubuntu 20.04, Jetson Nano만 가능)**

apt search nvidia-jetpack

apt-get install nvidia-jetpack

이 명령어로 설치가 가능하다.

실험 결과로 Jetson Nano에서는 NVIDIA CUDA 사이트에 접속해서 별도로 CUDA 10.2를 설치하는 것이 아니다.

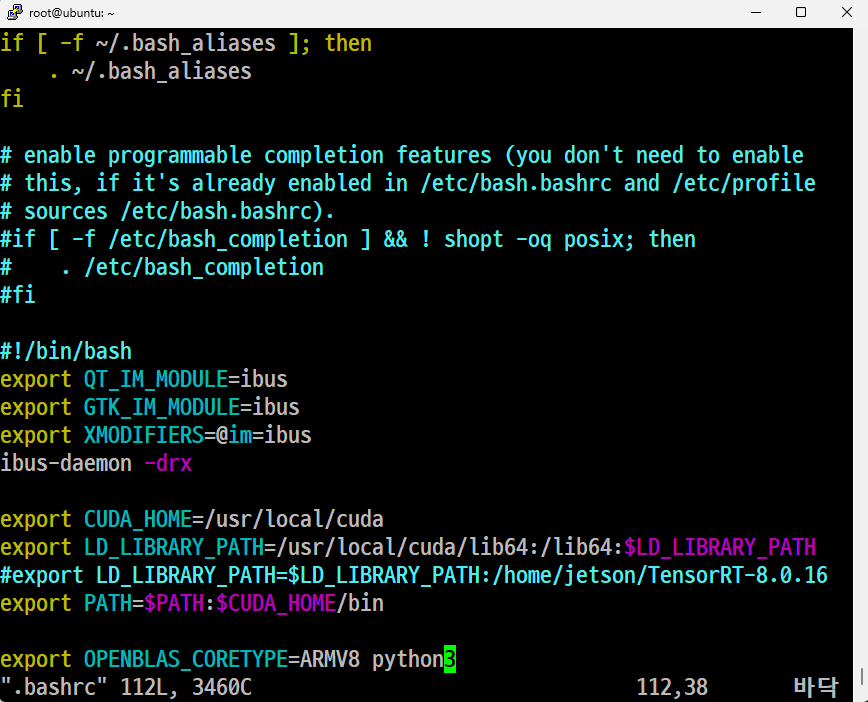
**4. Ubuntu 20.04에서 .bashrc 설치하기**

Jetpack 4.6.5를 설치하면 CUDA 10.2가 /usr/local/cuda-10.2에 설치된다.

경로를 지정해줘야 한다.

그리고 Python 3.8도 ARMv8 프로세서를 사용한다고 지정을 해줘야 한다.

|  |
| --- |
| **.bashrc 추가해야 할 내용** |
| #!/bin/bash  export QT\_IM\_MODULE=ibus  export GTK\_IM\_MODULE=ibus  export XMODIFIERS=@im=ibus  ibus-daemon -drx  export CUDA\_HOME=/usr/local/cuda  export LD\_LIBRARY\_PATH=/usr/local/cuda/lib64:/lib64:$LD\_LIBRARY\_PATH  #export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/home/jetson/TensorRT-8.0.16  export PATH=$PATH:$CUDA\_HOME/bin  export OPENBLAS\_CORETYPE=ARMV8 python3 |



**그림 3.** .bashrc 파일 수정하기

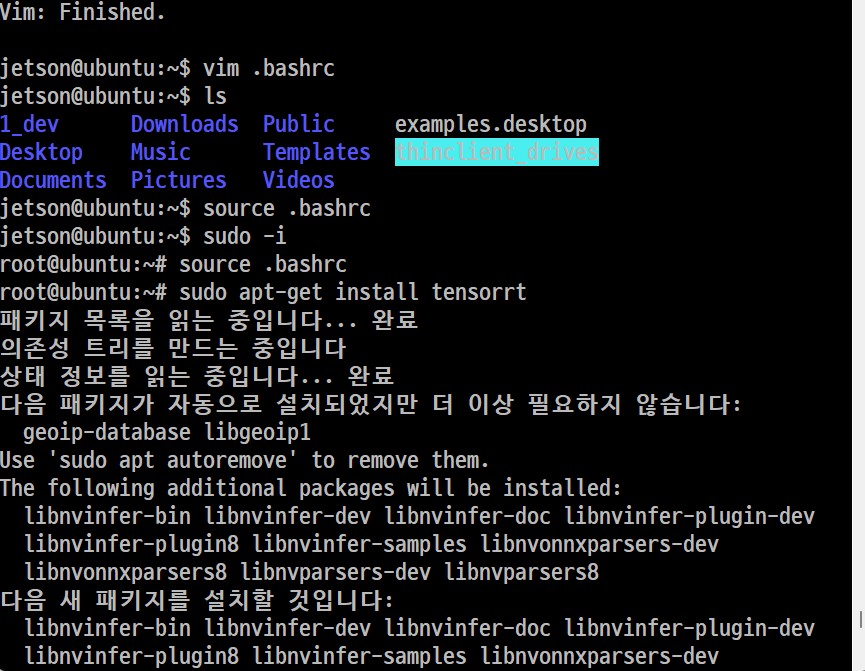
**5. Ubuntu 20.04에서 TensorRT 설치하기**

수동 컴파일을 하지 말고, apt-get으로 제공되는 TensorRT로 설치하면 된다.

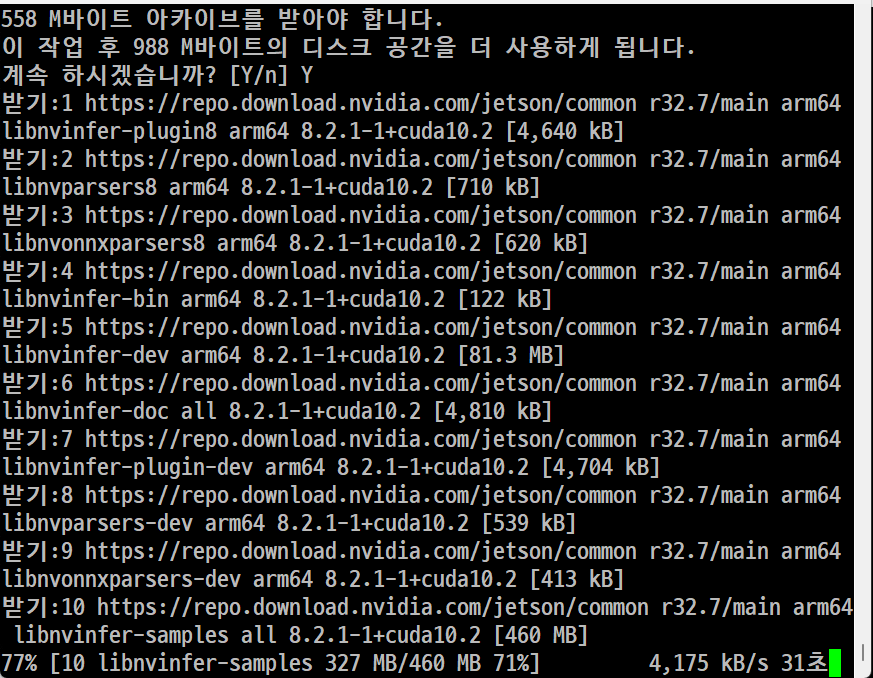
NVIDIA TensorRT 공식사이트에서 제공하는 tar 파일은 x86-64를 제공하므로 Jetson Nano에서 호환되지 않는다.

sudo apt-get install tensorrt

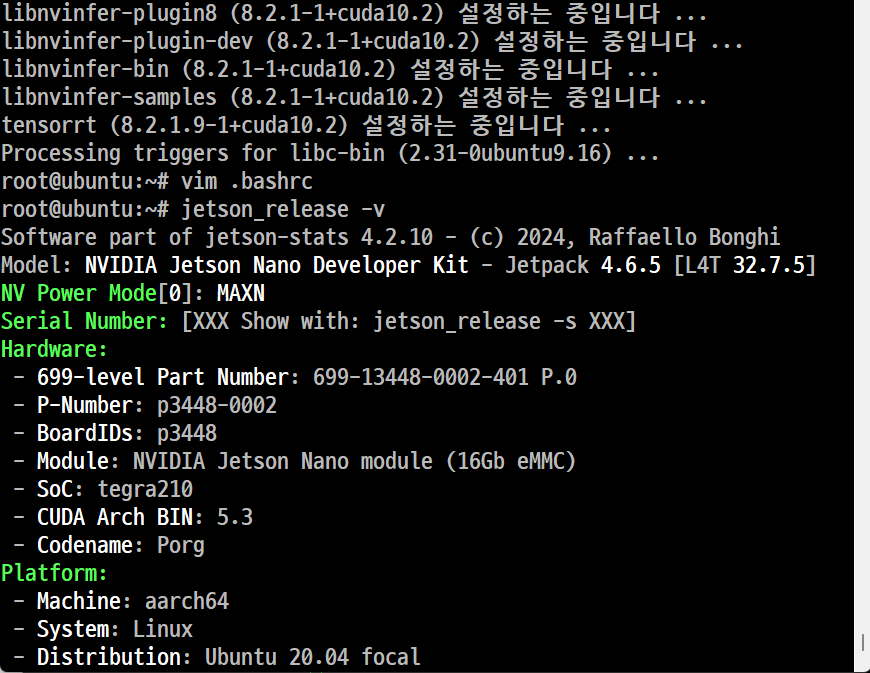
jetson\_release -v



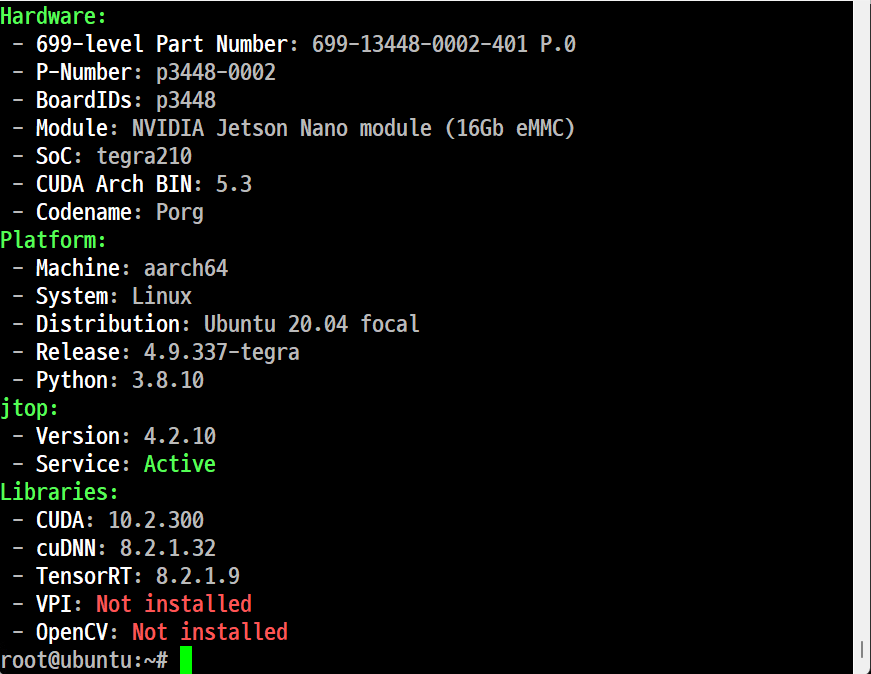
**그림 4.** TensorRT 설치하기



**그림 5.** TensorRT 설치하기



**그림 6.** TensorRT 설치하기

****

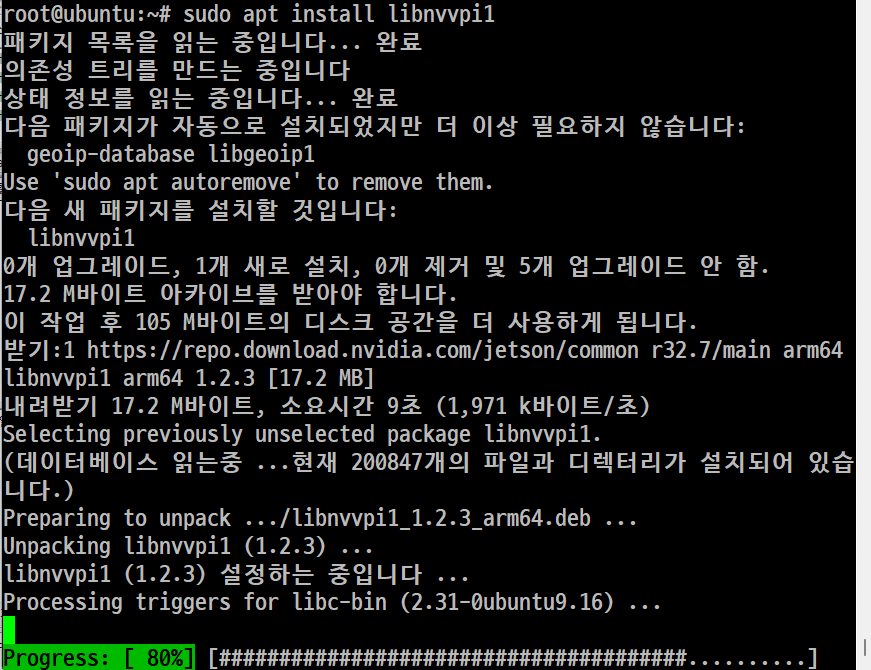
**그림 7.** TensorRT 설치하기

**6. Ubuntu 20.04에서 NVIDIA VPI 설치하기**

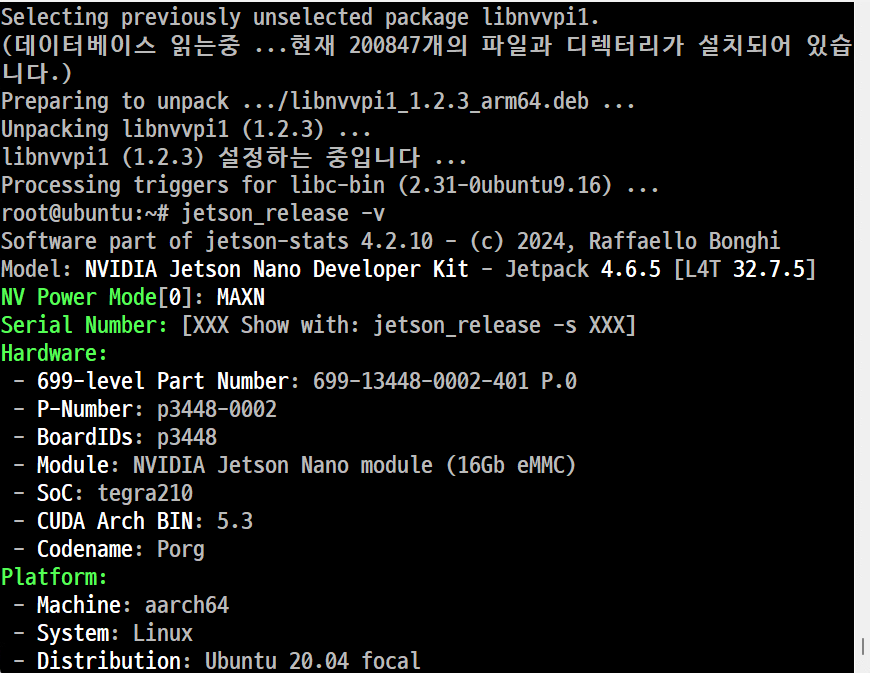
수동 컴파일을 하지 말고, apt-get으로 제공되는 libnvvpi1으로 설치하면 된다.

sudo apt install libnvvpi1

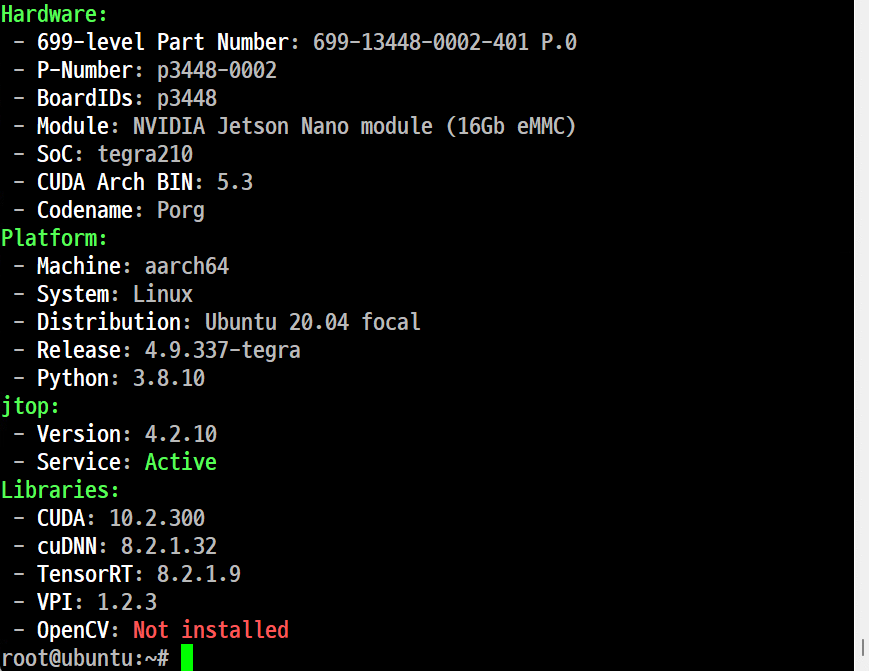
sudo jetson\_release -v



**그림 8.** NVIDIA-VPI 설치하기



**그림 9.** NVIDIA-VPI 설치하기



**그림 10.** NVIDIA-VPI 설치하기

**7. Ubuntu 20.04에서 OpenCV 4.5(opencv-python 4.5) 수동 설치**

수동 컴파일로 OpenCV를 설치해야 aarch64에서 동작한다.

**[1. 가상 메모리, 스왑 공간]**

# 이미 했다면 안해도 됨

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install nano

sudo apt-get install dphys-swapfile

# /sbin/dphys-swapfile 파일 open

sudo nano /sbin/dphys-swapfile

## Swap파일의 값이 다음과 같도록 값을 추가하거나, 파일 내 주석을 해제

# CONF\_SWAPSIZE=4096

# CONF\_SWAPFACTOR=2

# CONF\_MAXSWAP=4096

# 값을 수정 후 [Ctrl] + [X], [y], [Enter]를 눌러 저장하고 닫으면 됨.

# /etc/dphys-swapfile 파일 open

sudo nano /etc/dphys-swapfile

## Swap파일의 값이 다음과 같도록 값을 추가하거나, 파일 내 주석을 해제

# CONF\_SWAPSIZE=4096

# CONF\_SWAPFACTOR=2

# CONF\_MAXSWAP=4096

# 값을 수정 후 [Ctrl] + [X], [y], [Enter]를 눌러 저장하고 닫으면 됨.

# 재부팅

sudo reboot

**[2. 기존에 설치 되어 있는 CUDA 버전 확인 후 삭제]**

# 기존 cuda 버전 확인

pkg-config --modversion opencv

# 기존에 설치 된 cuda가 있다면 버전이 뜰 것.

# 설치 된 cuda가 없다면 'No package 'opencv' found' 출력 됨.

# 기존에 설치 된 cuda 버전 및 의존 패키지 삭제

sudo apt-get remove libopencv\*

sudo apt-get autoremove

sudo find /usr/local -name "\*opencv\*" -exec rm {} \;

**[3. 추가 패키지 설치]**

sudo apt-get update

# sudo apt-get upgrade -> 이건 자주 안하는 걸 권장한다 함..

sudo apt-get install -y build-essential cmake git unzip pkg-config

sudo apt-get install -y libjpeg-dev libpng-dev libtiff-dev

sudo apt-get install -y libavcodec-dev libavformat-dev libswscale-dev

sudo apt-get install -y libgtk2.0-dev libcanberra-gtk\*

sudo apt-get install -y python3-dev python3-numpy python3-pip

sudo apt-get install -y python3-dev python3-numpy libtbb2 libtbb-dev

sudo apt-get install -y libxvidcore-dev libx264-dev libgtk-3-dev

sudo apt-get install -y libtbb2 libtbb-dev libdc1394-22-dev

sudo apt-get install -y libv4l-dev v4l-utils

sudo apt-get install -y libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev

sudo apt-get install -y libavresample-dev libvorbis-dev libxine2-dev

sudo apt-get install -y libfaac-dev libmp3lame-dev libtheora-dev

sudo apt-get install -y libopencore-amrnb-dev libopencore-amrwb-dev

sudo apt-get install -y libopenblas-dev libatlas-base-dev libblas-dev

sudo apt-get install -y liblapack-dev libeigen3-dev gfortran

sudo apt-get install -y libhdf5-dev protobuf-compiler

sudo apt-get install -y libprotobuf-dev libgoogle-glog-dev libgflags-dev

sudo apt install ccache

**[4. opencv zip file 다운 및 압축 해제]**

wget -O opencv.zip https://github.com/opencv/opencv/archive/4.5.0.zip

wget -O opencv\_contrib.zip https://github.com/opencv/opencv\_contrib/archive/4.5.0.zip

unzip opencv.zip

unzip opencv\_contrib.zip

**[5. build 폴더 생성]**

# 압축 해제 후 생성 된 opencv-4.5.0 폴더 안으로 이동

cd opencv-4.5.0

# build 폴더 생성 후 이동

mkdir build

cd build

**[6. 컴파일러 설치]**

sudo apt-get install g++-9 (CUDA 10.2을 설치하면 9버전은 사용하면 안 된다.)

**[7. cmake로 빌드하기] (1시간 30분 소요)**

cmake -D CMAKE\_BUILD\_TYPE=RELEASE \

-D CMAKE\_INSTALL\_PREFIX=/usr/local \

-D OPENCV\_EXTRA\_MODULES\_PATH=/root/opencv\_contrib-4.5.0/modules \

-D EIGEN\_INCLUDE\_PATH=/usr/include/eigen3 \

-D WITH\_OPENCLAMDBLAS=OFF \

-D WITH\_GTK=OFF \

-D WITH\_OPENCL=OFF \

-D WITH\_CUDA=ON \

-D CUDA\_FAST\_MATH=ON \

-D OPENCV\_DNN\_CUDA=ON \

-D CUDA\_ARCH\_PTX="" \

-D WITH\_CUDNN=ON \

-D WITH\_CUBLAS=ON \

-D ENABLE\_FAST\_MATH=ON \

-D ENABLE\_NEON=ON \

-D WITH\_QT=OFF \

-D WITH\_OPENMP=ON \

-D WITH\_OPENGL=ON \

-D BUILD\_JASPER=ON \

-D BUILD\_TIFF=ON \

-D WITH\_FFMPEG=ON \

-D WITH\_GSTREAMER=ON \

-D WITH\_TBB=ON \

-D BUILD\_TBB=ON \

-D WITH\_EIGEN=ON \

-D WITH\_V4L=ON \

-D WITH\_LIBV4L=ON \

-D WITH\_VTK=ON \

-D BUILD\_PYTHON=ON \

-D OPENCV\_ENABLE\_NONFREE=ON \

-D INSTALL\_PYTHON\_EXAMPLES=OFF \

-D BUILD\_NEW\_PYTHON\_SUPPORT=ON \

-D OPENCV\_GENERATE\_PKGCONFIG=ON \

-D BUILD\_OPENCV\_WORLD=ON \

-D BUILD\_PERF\_TESTS=OFF \

-D BUILD\_TESTS=OFF \

-D BUILD\_OPENCV\_PYTHON\_TESTS=OFF \

-D BUILD\_JAVA=OFF \

-D INSTALL\_TESTS=OFF \

-D INSTALL\_C\_EXAMPLES=OFF \

-D BUILD\_EXAMPLES=OFF ..

**[8. 컴파일과 설치]**

$ make -j4

$ sudo rm -rf /usr/include/opencv4/opencv2

$ sudo make install

**[9. /usr/local/lib 경로가 출력되는지 확인해야 한다.]**

$ cat /etc/ld.so.conf.d/\*

만약 /usr/local/lib 출력되지 않으면 아래 명령어로 경로를 추가해 줍니다.

$ sudo sh -c 'echo '/usr/local/lib' > /etc/ld.so.conf.d/opencv.conf'

- 공유 라이브러리 정보를 갱신하여 경로 등이 적용되게 합니다.

$ sudo ldconfig

**[10. jetson-release 확인(OpenCV 설치확인)]**

jetson\_release –v

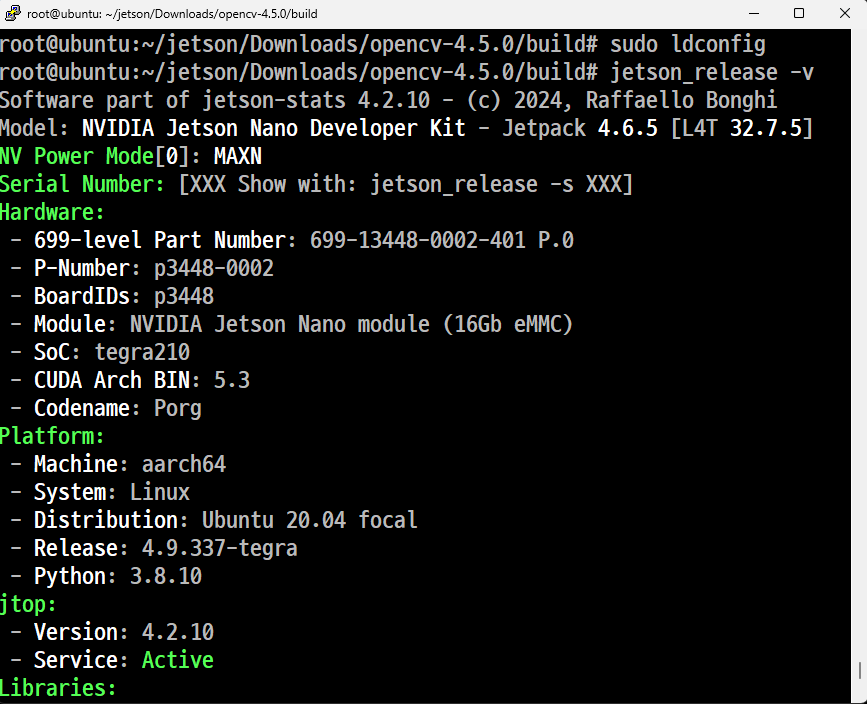
**[11. OpenCV-Python에서 버전 확인하기]**

pip install로 opencv-python을 설치하면 안 된다.

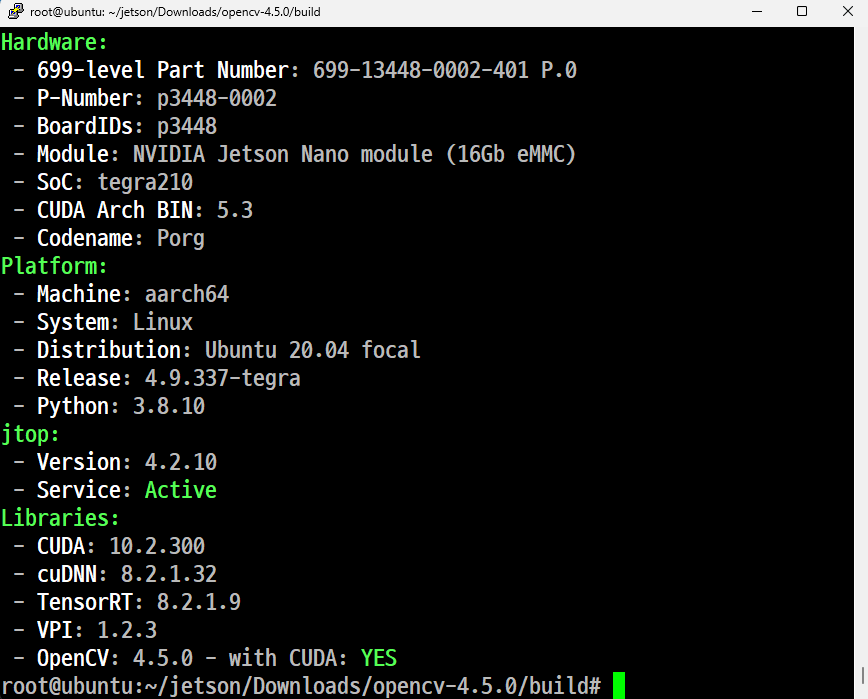
python3

import cv2

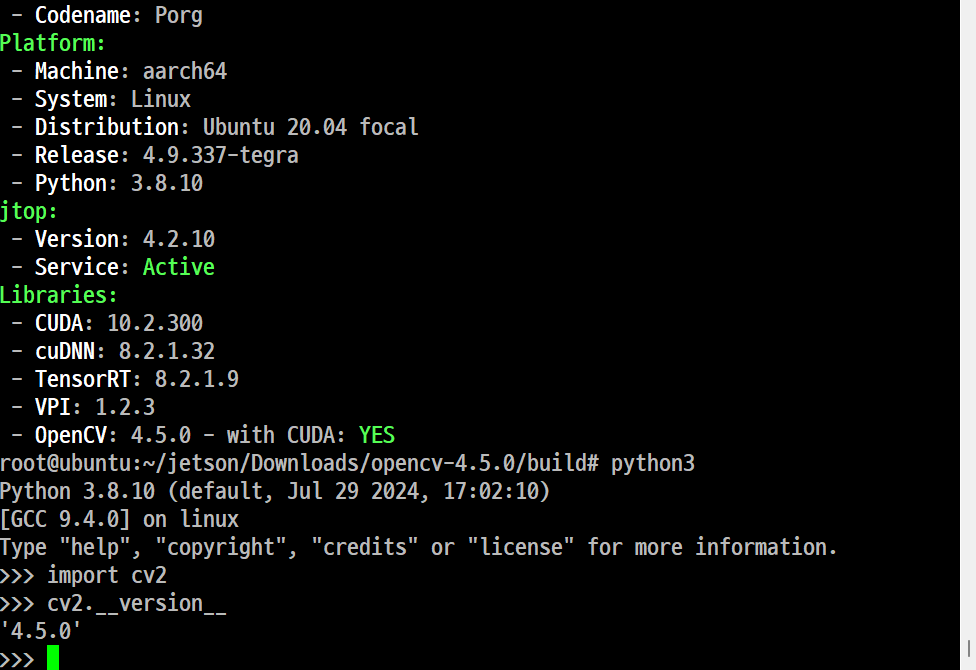
cv2.\_\_version\_\_



**그림 11.** jetson\_release로 OpenCV with CUDA 설치 확인



**그림 12.** jetson\_release로 OpenCV with CUDA 설치 확인



**그림 13.** jetson\_release로 OpenCV with CUDA 설치 확인

**8. Ubuntu 20.04에서 Pytorch 1.13 설치하기(Q-engineeering)**

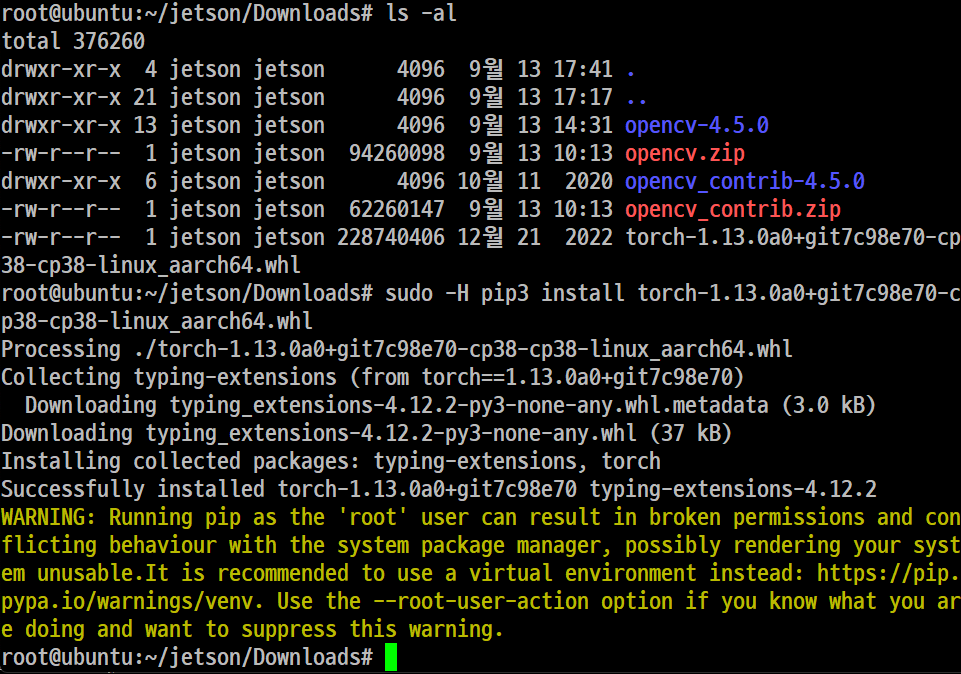
Yahboom Jetson Nano B01 Sub, NVIDIA Jetson Nano에서 우분투 20.04를 설치하면 해당 명령어는 정상적으로 동작한다.

|  |
| --- |
| # install the dependencies (if not already onboard)  $ sudo apt-get install python3-pip libjpeg-dev libopenblas-dev libopenmpi-dev libomp-dev  $ sudo -H pip3 install future  $ sudo pip3 install -U --user wheel mock pillow  $ sudo -H pip3 install testresources  # above 58.3.0 you get version issues  $ sudo -H pip3 install setuptools==58.3.0  $ sudo -H pip3 install Cython  # install gdown to download from Google drive  $ sudo -H pip3 install gdown  # download the wheel  $ gdown https://drive.google.com/uc?id=1e9FDGt2zGS5C5Pms7wzHYRb0HuupngK1  # install PyTorch 1.13.0  $ sudo -H pip3 install torch-1.13.0a0+git7c98e70-cp38-cp38-linux\_aarch64.whl  # clean up  $ rm torch-1.13.0a0+git7c98e70-cp38-cp38-linux\_aarch64.whl |

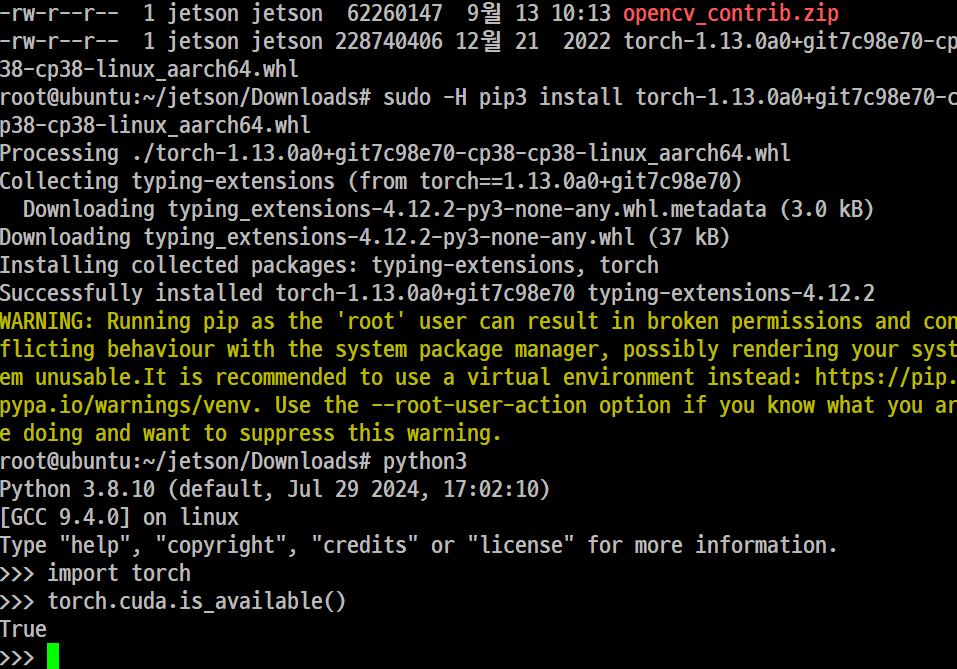
python3

import torch

torch.cuda.is\_available()



**그림 14.** pytorch 설치하기



**그림 15.** pytorch 설치 후 CUDA 확인하기

**9. Ubuntu 20.04에서 torchvision 설치하기(수동 컴파일)**

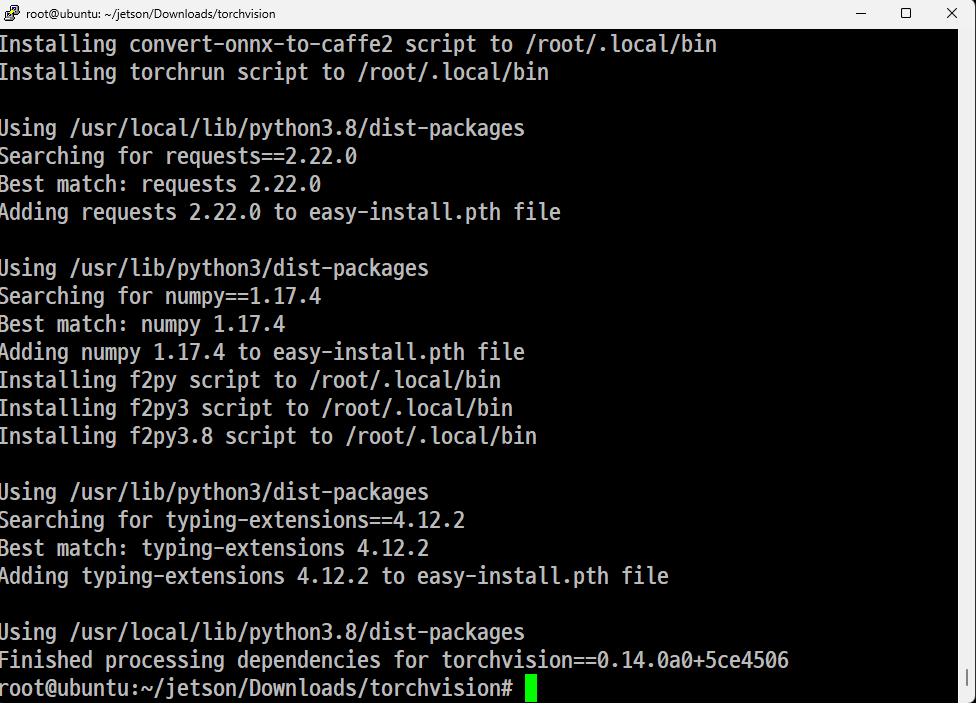
torchvision을 pip로 설치하면 aarch64(ARM64)에서 호환이 되지 않으므로 수동으로 컴파일로 설치해야 한다.

|  |
| --- |
| [torchvision]  sudo apt install -y libjpeg-dev zlib1g-dev  git clone https://github.com/pytorch/vision torchvision  cd torchvision  git checkout v0.16.2  python3 setup.py install --user |

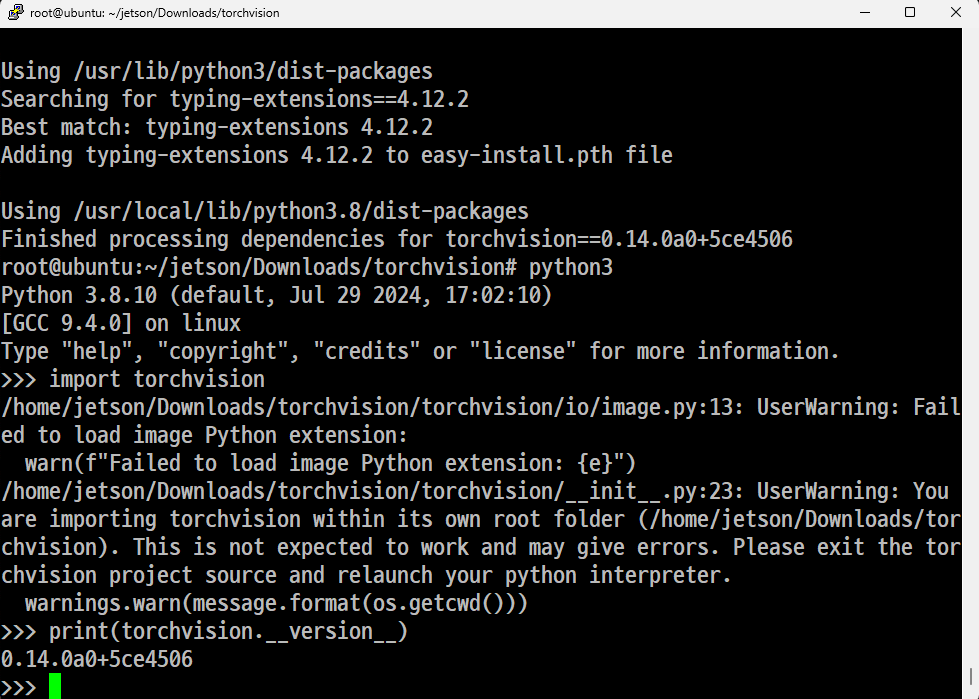
python3

import torchvision

print(torchvision.\_\_version\_\_)



**그림 16.** torchvision 설치하기

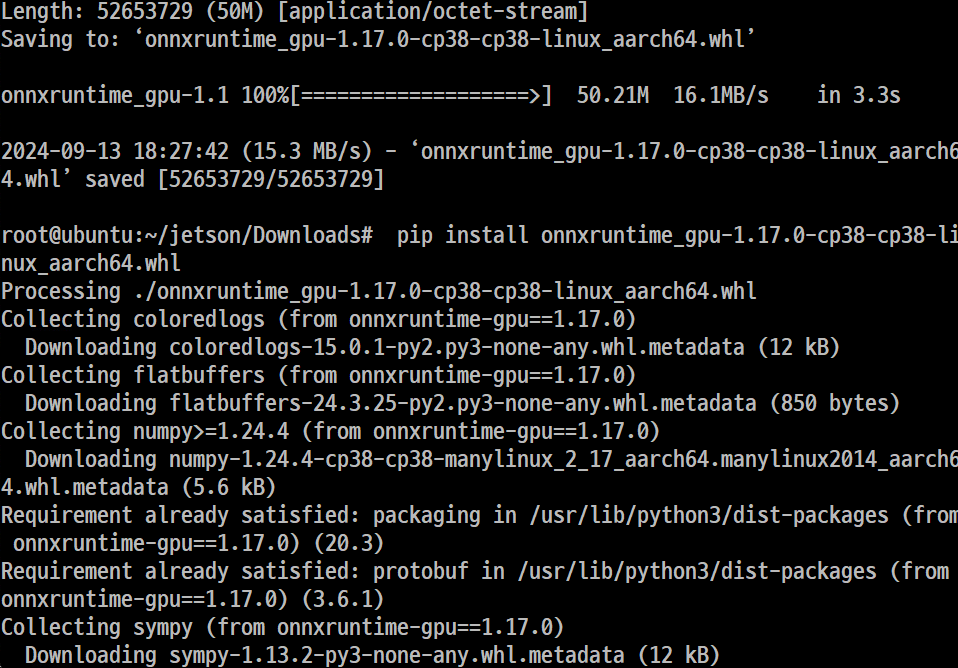


**그림 17.** torchvision 설치하기

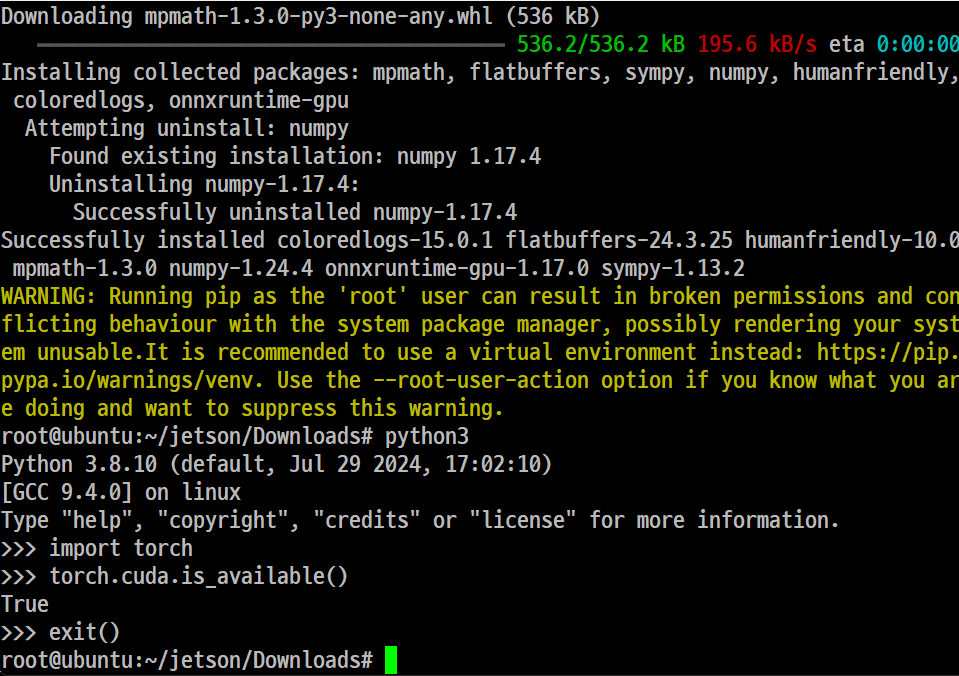
**10. Ubuntu 20.04에서 onnxruntime-gpu 설치하기**

pip로 nvidia의 box에서 제공하는 onnxruntime으로 설치해야 한다.

|  |
| --- |
| [onnxruntime-gpu] (30분)  wget https://nvidia.box.com/shared/static/zostg6agm00fb6t5uisw51qi6kpcuwzd.whl -O onnxruntime\_gpu-1.17.0-cp38-cp38-linux\_aarch64.whl  pip install onnxruntime\_gpu-1.17.0-cp38-cp38-linux\_aarch64.whl |



**그림 18.** onnxruntime\_gpu 설치하기



**그림 19.** onnxruntime\_gpu 설치하기