

Exercise 4.3

Yolov8 Object Detection and Counting on Nvidia Jetson AGX Orin GPU

Overview:

In this lab, students will set up the Nvidia Jetson AGX Orin and familiarize themselves with it. We will set up the environment for AI applications. Furthermore, we will do object detection and counting using the popular Yolov8 model. Please note that the boards have been already flashed with the Jetpack 6 SDK. Some useful wheel files for installing the torch 2.3 and torchvision 1.8 were also downloaded. Follow the steps outlined and ask the TAs and the instructors if you have any difficulty.

Required Software:

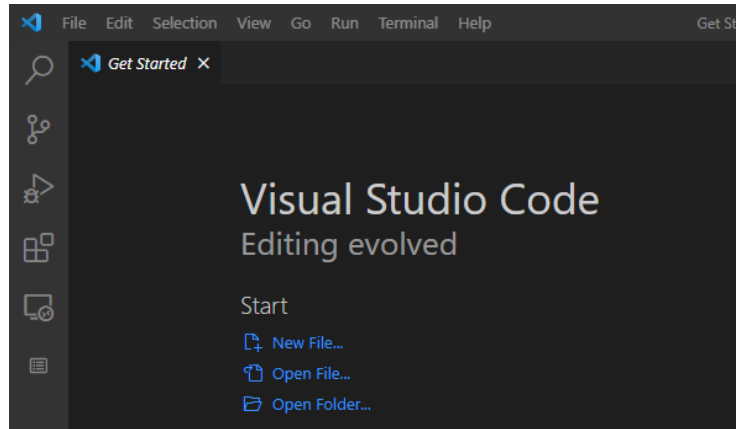
1. Windows/Linux/MAC Host PC
2. Visual Studio Code:
3. Putty

Steps for the Lab Session:

1. Download and install the required software
2. Connect and access the Jetson AGX Orin board
3. Setup the AI environment
4. Performing the Yolov8 object detection Inference
5. Object counting with Yolov8
6. Home Exercise: Face recognition

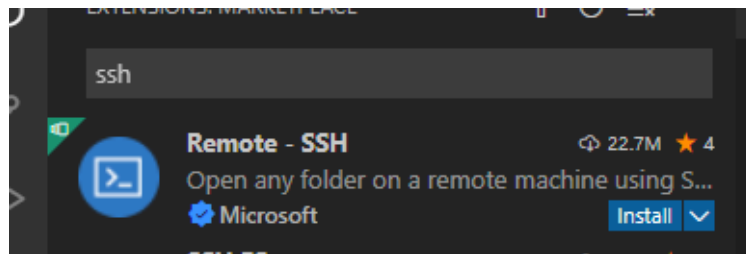
1.0 Download and Installed the Required Software:

1. **Visual Studio Code:**
 - Download and install VS Code: <https://code.visualstudio.com/>
 - Open VS code from the start menu or by searching

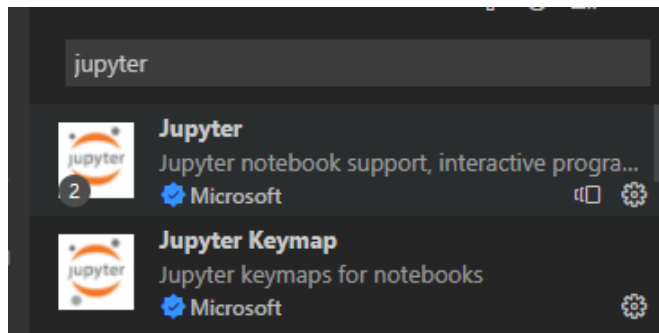


In the VS Code Environment, add the following extensions

1. Python
2. Remote SSH



3. Jupyter extensions



2. Putty:

Download and install Putty: <https://www.putty.org/>

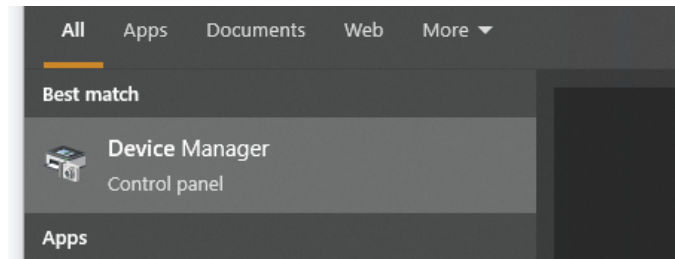
2.0 Connect and access the Jetson AGX Orin board

To connect and access the board there are various ways as follows:

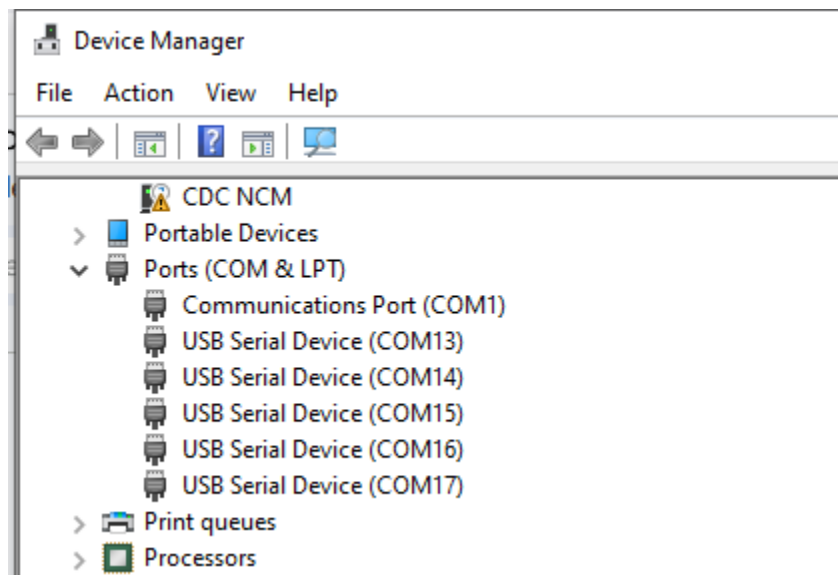
1. Using serial port (Use Putty or other serial tools)
2. SSH using Putty
3. SSH using command prompt
4. SSH in VS Code

2.1 Remote Access to the Board Using Serial Port with Putty

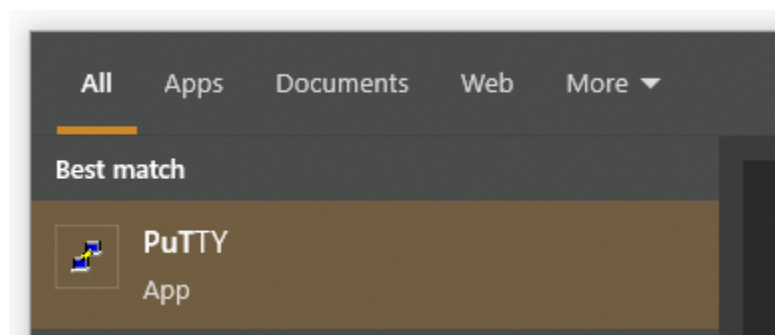
- Open Device Manager



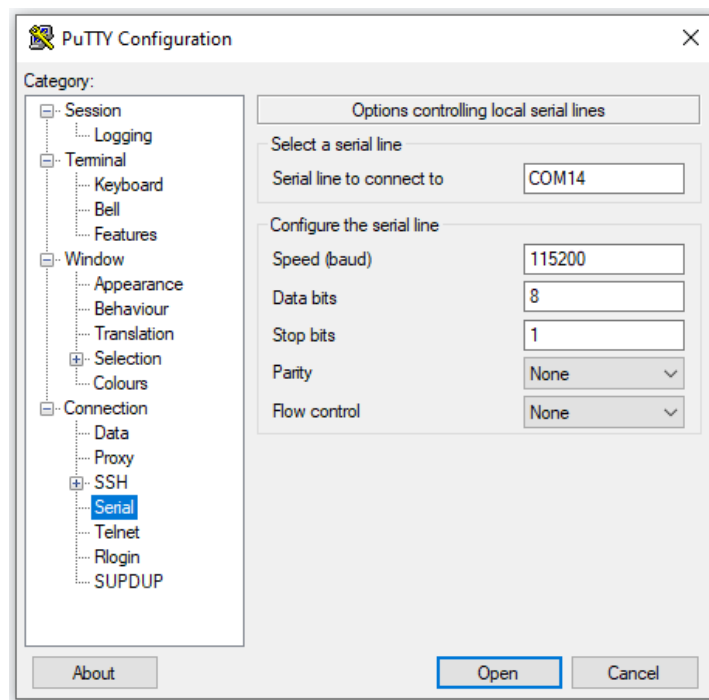
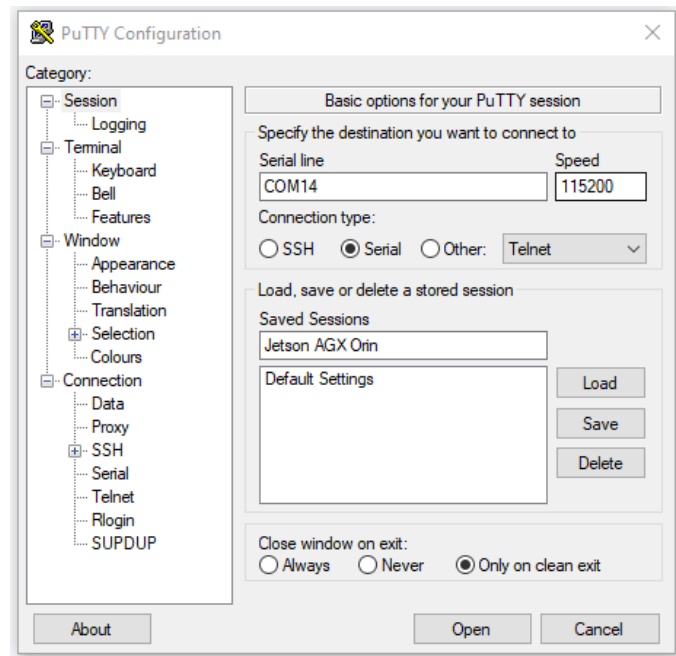
- Check the Port Number of the board under **PORTS**. The board has 4 serial ports, use the first one. Choose the first Port that comes with the board. Here COM14



- Open Putty



- Select the **Serial** option and enter the port number and the Baud rate (115200). You can adjust the setting.



- Click Open. You can refresh the board, or type ls to continue if the board already started.

```
COM17 - PuTTY
ubuntu login: eeuser
Password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.136-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

Expanded Security Maintenance for Applications is not enabled.

252 updates can be applied immediately.
195 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

28 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Last login: Mon Sep  2 15:14:57 HKT 2024 from 192.168.55.100 on pts/0
eeuser@ubuntu:~$
```

- Check the ip address: >> ifconfig

```
COM17 - PuTTY

usb0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether da:f9:05:60:ae:49 txqueuelen 1000 (Ethernet)
    RX packets 12861 bytes 1163929 (1.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2160 bytes 533641 (533.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

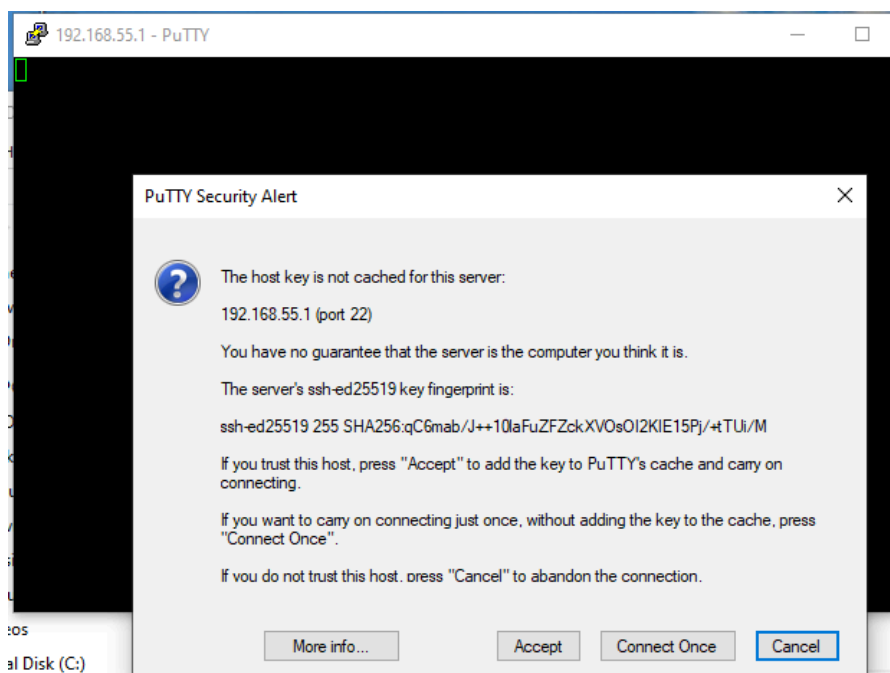
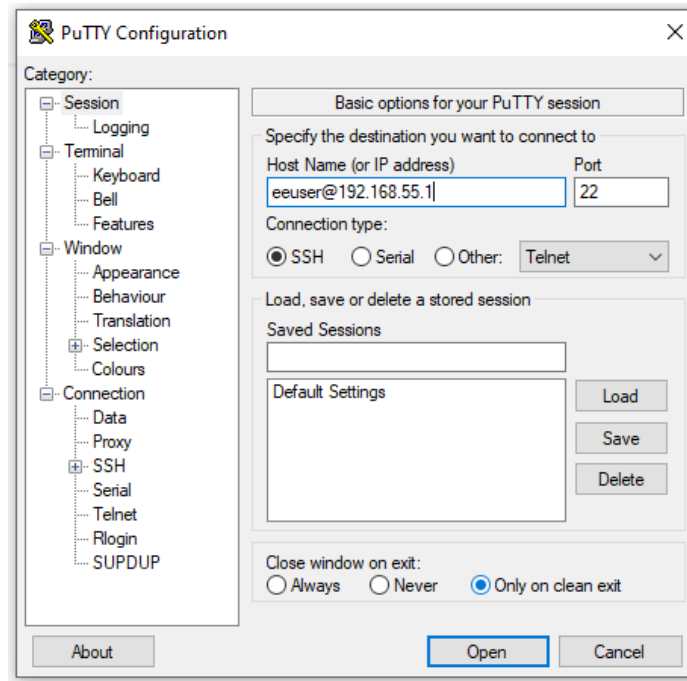
usb1: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether da:f9:05:60:ae:4b txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.28.199.95 netmask 255.255.128.0 broadcast 172.28.199.255
    inet6 fe80::bfa8:e86c:da2f:b024 prefixlen 64 scopeid 0x20:0:0:0
    ether 48:e7:da:41:33:c1 txqueuelen 1000 (Ethernet)
    RX packets 716358 bytes 1049847798 (1.0 GB)
    RX errors 0 dropped 4 overruns 0 frame 0
    TX packets 202075 bytes 21424934 (21.4 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eeuser@ubuntu:~$
```

2.2 SSH using Putty:

- You can alternatively choose SSH in Putty
- Connect the USB cable to the board and the PC
- Enter the hostname@ip_address e.g. eeuser@192.168.55.1
- Accept the key connection request and Click Open



```
eeuser@ubuntu: ~
* Management:      https://landscape.canonical.com
* Support:         https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

Expanded Security Maintenance for Applications is not enabled.

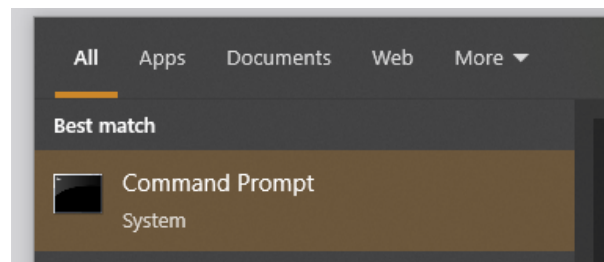
252 updates can be applied immediately.
195 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

28 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Last login: Mon Sep  2 17:13:26 2024 from 192.168.55.100
eeuser@ubuntu:~$
```

2.3 SSH using command prompt

- You can open a command Prompt (type cmd)



- SSH to the board by typing the command:
>> ssh eeuser@192.168.55.1

```
Command Prompt - ssh eeuser@172.28.199.95
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

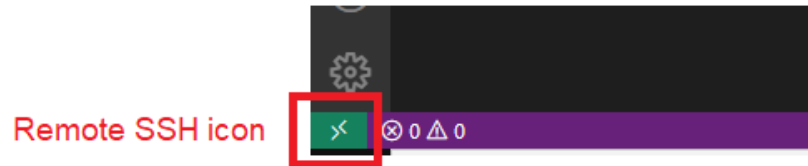
C:\Users\abdul>ssh eeuser@ 172.28.199.95
ssh: connect to host port 22: Connection refused

C:\Users\abdul>ssh eeuser@172.28.199.95
The authenticity of host '172.28.199.95 (172.28.199.95)' can't be established.
ECDSA key fingerprint is SHA256:vxjUW1wGVARpGIcRkfusYFjZvWYRn+yzimCTtCe8xT8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

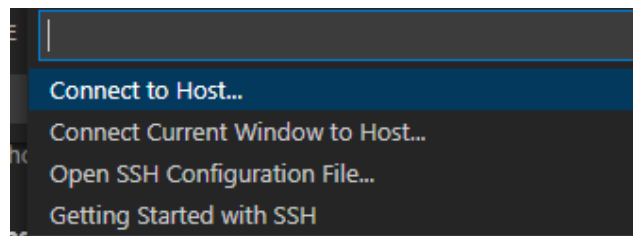
- Accept the connect request and press enter

2.3 SSH using Visual Studio Code

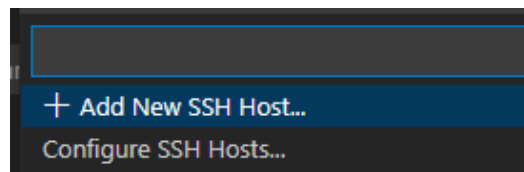
- Open VS Code
- Click at the Remote SSH icon at the left bottom of the VS Code window



- Select **Connect to Host** from the window that open at the top

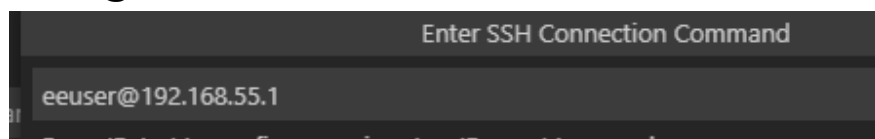


- Choose the **+ Add New SSH Host**

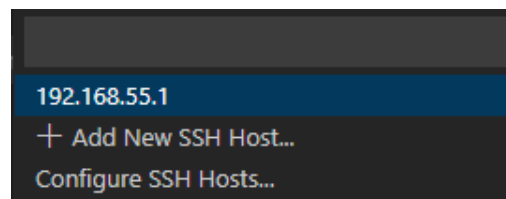


- Enter the ssh connection command

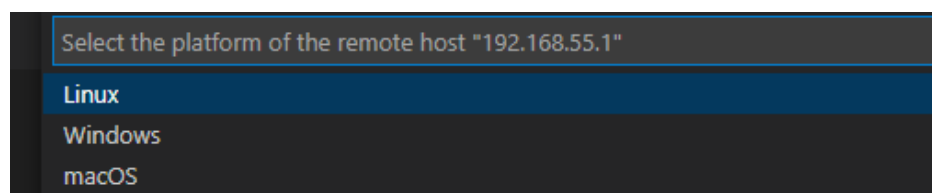
>> eeuser@192.168.55.1



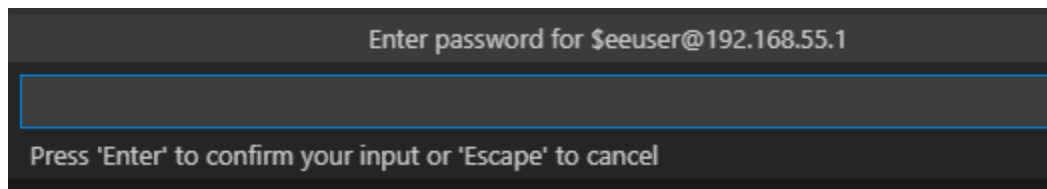
- Click the SSH icon again, now select the saved ip address of the board (192.168.55.1)



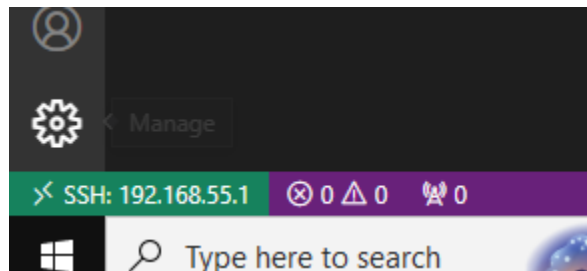
- Select Linux and Continue



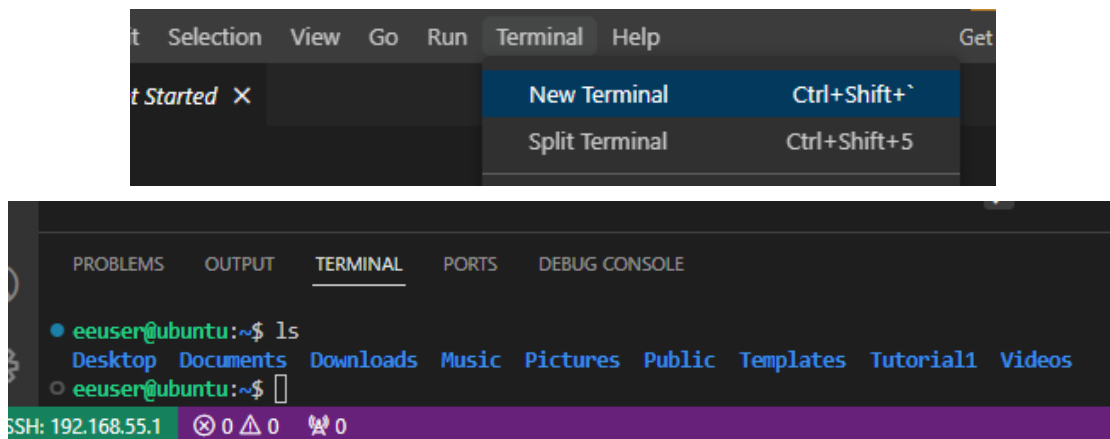
- Enter the Password



- SSH will now be ready



- Open a terminal or a folder on the board



3.0 Setting up the AI environment

Here, we set up the environment for the AI inference. We can alternatively use docker by running the below command. However, in this tutorial we will follow the native installation

Steps involved:

1. Establish and check network connection
2. Check Python, virtual environment, and JeckPack 6 availability
3. Check Tensorrt availability
4. Create a virtual environment
5. Copy tensorrt to the virtual environment package folder
6. Install Ultralytics
7. Install Torch 2.3 and Torchvision 1.8
8. Install Onnxruntime
9. Test the environment

3.1 Establish and check network connection

- Connect the board to either WiFi or share Ethernet from your PC
- Ping Google to ensure internet connection
 - >> ping www.google.com
- Press ctrl+C to stop

```
eeuser@ubuntu:~$ ping www.google.com
PING www.google.com (142.250.76.4) 56(84) bytes of data.
64 bytes from nchkg-a-in-f4.1e100.net (142.250.76.4): icmp_seq=1 ttl=59 time=4.49 ms
64 bytes from nchkg-a-in-f4.1e100.net (142.250.76.4): icmp_seq=2 ttl=59 time=4.91 ms
64 bytes from nchkg-a-in-f4.1e100.net (142.250.76.4): icmp_seq=3 ttl=59 time=17.0 ms
^C
--- www.google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 4.489/59.770/169.915/77.884 ms
eeuser@ubuntu:~$
```

3.2 Check Python, virtual environment, and JeckPack 6 availability

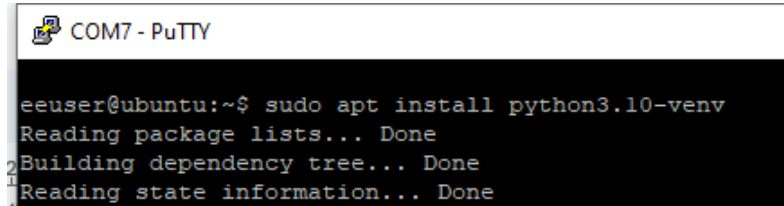
- Check if python has been installed
 - >> python --version
- If Python is not installed, you will need to install it or download and install it from the official Python website.
 - >> sudo apt-get update
 - >> sudo apt install python3.10
- Check if the virtual environment has been installed

>> virtualenv --version

- If Python environment is not installed, you will need to install it

>> sudo apt-get update

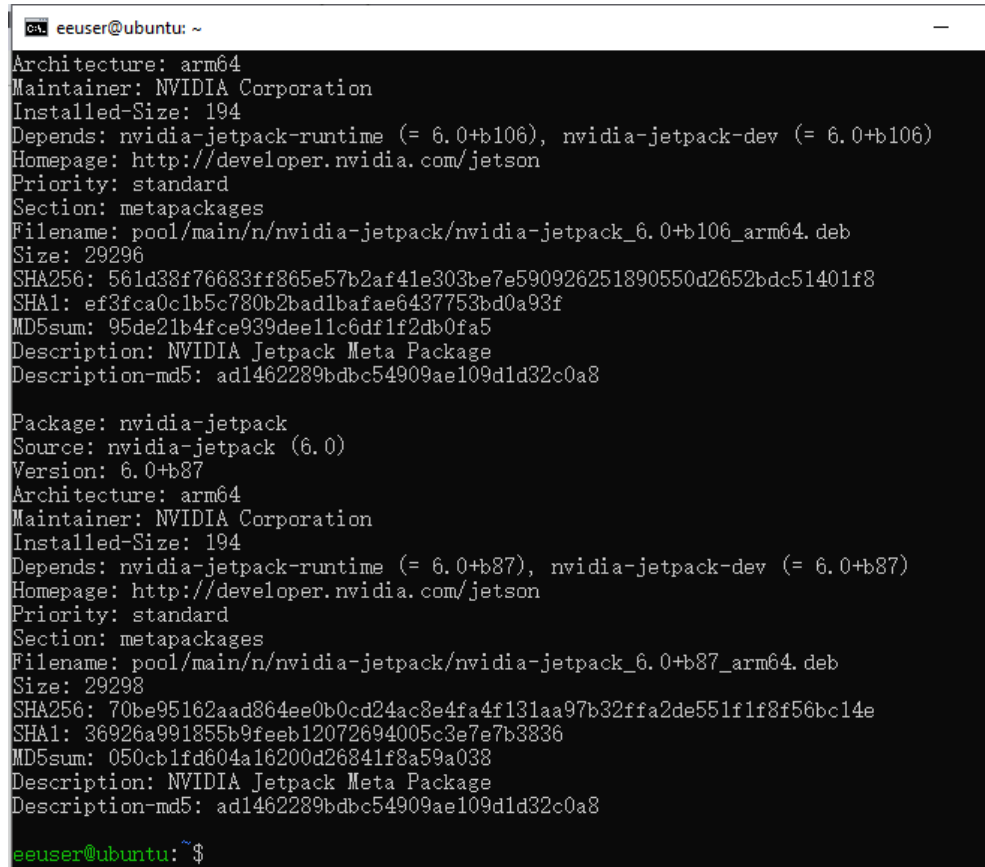
>> sudo apt install python3.10-venv



```
COM7 - PuTTY
eeuser@ubuntu:~$ sudo apt install python3.10-venv
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

- Check if Jetpack 6 has been successfully installed
- Check the Jetpack version

>> sudo apt-cache show nvidia-jetpack



```
eeuser@ubuntu: ~
Architecture: arm64
Maintainer: NVIDIA Corporation
Installed-Size: 194
Depends: nvidia-jetpack-runtime (= 6.0+b106), nvidia-jetpack-dev (= 6.0+b106)
Homepage: http://developer.nvidia.com/jetson
Priority: standard
Section: metapackages
Filename: pool/main/n/nvidia-jetpack/nvidia-jetpack_6.0+b106_arm64.deb
Size: 29296
SHA256: 561d38f76683ff865e57b2af41e303be7e590926251890550d2652bdc51401f8
SHA1: ef3fca0c1b5c780b2bad1bafae6437753bd0a93f
MD5sum: 95de21b4fce939dee11c6d1f2db0fa5
Description: NVIDIA Jetpack Meta Package
Description-md5: ad1462289bdbc54909ae109d1d32c0a8

Package: nvidia-jetpack
Source: nvidia-jetpack (6.0)
Version: 6.0+b87
Architecture: arm64
Maintainer: NVIDIA Corporation
Installed-Size: 194
Depends: nvidia-jetpack-runtime (= 6.0+b87), nvidia-jetpack-dev (= 6.0+b87)
Homepage: http://developer.nvidia.com/jetson
Priority: standard
Section: metapackages
Filename: pool/main/n/nvidia-jetpack/nvidia-jetpack_6.0+b87_arm64.deb
Size: 29298
SHA256: 70be95162aad864ee0b0cd24ac8e4fa4f131aa97b32ffa2de551f1f8f56bc14e
SHA1: 36926a991855b9feeb12072694005c3e7e7b3836
MD5sum: 050cb1fd604a16200d26841f8a59a038
Description: NVIDIA Jetpack Meta Package
Description-md5: ad1462289bdbc54909ae109d1d32c0a8

eeuser@ubuntu:~$
```

3.3 Check Tensorrt availability

- Check Tensorrt installation directory
>> pip show tensorrt
- Check Tensorrt version

```
>> python3.10
```

```
import tensorrt as trt
print(trt.__version__)
```

If Tensorrt is not found, you may need to call the attention of our TA. The TA will either install the Tensorrt dependency or reinstall the jetpack again by running these commands:

Install tensorrt dependency

```
>> sudo apt install python3-libnvinfer
```

Re-install Jetpack 6

```
>> sudo apt-get update
```

```
>> sudo apt install jetpack
```

3.4 Create a virtual environment

- Open a terminal and navigate to the directory where you want to create your virtual environment. Run the following command to create a new virtual environment named ultralytics-env:

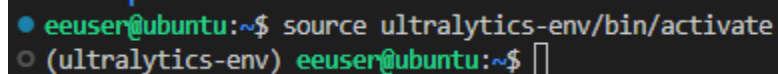
```
>> python3 -m venv ultralytics-env
```

3.4.1 Activate the Virtual Environment

- Activate the virtual environment using the following command:

```
>> source ultralytics-env/bin/activate
```

- After activation, your terminal prompt should change to indicate that you are now working within the virtual environment.

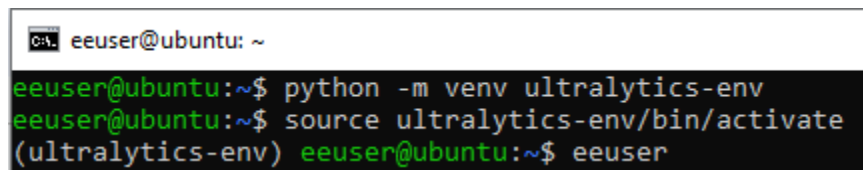


```
eeuser@ubuntu:~$ source ultralytics-env/bin/activate
(ultralytics-env) eeuser@ubuntu:~$
```

3.4.2 Deactivate the Virtual Environment

- Once you are done working in the virtual environment at the end of the lab or if you want to troubleshoot, you can deactivate it by running:

```
>> deactivate
```



```
eeuser@ubuntu:~$ python -m venv ultralytics-env
eeuser@ubuntu:~$ source ultralytics-env/bin/activate
(ultralytics-env) eeuser@ubuntu:~$ deactivate
eeuser@ubuntu:~$
```

3.5 Copy tensorrt to the virtual environment package directory

Since you want to use tensorrt and the tensorrt is not installed in the virtual environment, you need to copy the tensorrt installation directory to your Python environment's packages directory.

- To find the tensorrt installation directory, run this command outside the environment:
 >> pip which tensorrt
- Usually, the tensorrt will be installed in /usr/bin/python3.10/dist-packages
- To find the Python environment's packages directory, run this command inside the environment:
 >> pip show pip

```
(ultralytics-env) eeuser@ubuntu:~$ pip show pip
Name: pip
Version: 24.2
Summary: The PyPA recommended tool for installing Python packages.
Home-page: https://pip.pypa.io/
Author:
Author-email: The pip developers <distutils-sig@python.org>
License: MIT
Location: /home/eeuser/ultralytics-env/lib/python3.10/site-packages
Requires:
Required-by:
```

- Copy the tensorrt from its global installation directory to the environment's packages directory.

```
>> sudo cp -r /usr/lib/python3.10/dist-packages/tensorrt
/home/eeuser/ultralytics-env/lib/python3.10/site-packages
```

```
● (ultralytics-env) eeuser@ubuntu:~$ sudo cp -r /usr/lib/python3.10/dist-packages/tensorrt /home/
eeuser/ultralytics-env/lib/python3.10/site-packages
[sudo] password for eeuser:
● (ultralytics-env) eeuser@ubuntu:~$ python3.10
```

- **Check the tensorrt version in the environment to verify its availability.**

```
>> python3.10
import tensorrt as trt
print(trt.__version__)
```

```
● (ultralytics-env) eeuser@ubuntu:~$ python3.10
Python 3.10.12 (main, Sep 11 2024, 15:47:36) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorrt as trt
>>> exit()
○ (ultralytics-env) eeuser@ubuntu:~$
```

3.6 Install Ultralytics Package

Now we will install Ultralytics package on the Jetson with optional dependencies so that we can export the PyTorch models to other different formats. We will mainly focus on [NVIDIA TensorRT exports](#) because TensorRT will make sure we can get the maximum performance out of the Jetson devices.

- Update packages list, install pip and upgrade to latest

```
>> sudo apt update
```

```
(ultralitics-env) eeuser@ubuntu:~$ sudo apt update
[sudo] password for eeuser:
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 https://repo.download.nvidia.com/jetson/common r36.3 InRelease
Hit:3 https://repo.download.nvidia.com/jetson/t234 r36.3 InRelease
Hit:4 https://repo.download.nvidia.com/jetson/ffmpeg r36.3 InRelease
Hit:5 http://ports.ubuntu.com/ubuntu-ports jammy InRelease
Get:6 http://ports.ubuntu.com/ubuntu-ports jammy-updates InRelease [128 kB]
Hit:7 http://ports.ubuntu.com/ubuntu-ports jammy-backports InRelease
Get:8 http://ports.ubuntu.com/ubuntu-ports jammy-security InRelease [129 kB]
Get:9 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 Packages [1,738 kB]
Fetched 1,995 kB in 3s (604 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
247 packages can be upgraded. Run 'apt list --upgradable' to see them.
(ultralitics-env) eeuser@ubuntu:~$
```

```
>> sudo apt install python3-pip -y
```

```
(ultralitics-env) eeuser@ubuntu: ~$ sudo apt install python3-pip -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3-pip is already the newest version (22.0.2+dfsg-1ubuntu0.4).
The following packages were automatically installed and are no longer required:
  gdal-data libaec0 libarmadillo10 libarpac2 libavcodec-dev libavformat-dev libavutil-dev
  libblsscl1 libcfitsio9 libcharls2 libdc1394-dev libdeflate-dev libdouble-conversion3
  libexif-dev libfreexl1 libfyba0 libgdal30 libgdc-m-dev libgdc-m3.0 libgeos-c1v5
  libgeos3.10.2 libgeotiff5 libgl2ps1.4 libglew2.2 libgphoto2-dev libhdf4-0-alt
  libhdf5-103-1 libhdf5-hl-100 libheif1 libilmbase-dev libjpeg-dev libjpeg-turbo8-dev
  libjpeg8-dev libkmlbase1 libkml-dom1 libkml-engine1 libleft5 libminizip1
  libmysqlclient21 libnetcdf19 libodbc2 libodbcinst2 libogdi4.1 libopencv-calib3d4.5d
  libopencv-contrib4.5d libopencv-dnn4.5d libopencv-features2d4.5d libopencv-flann4.5d
  libopencv-highgui4.5d libopencv-imgcodecs4.5d libopencv-imgproc4.5d libopencv-ml4.5d
  libopencv-objectdetect4.5d libopencv-photo4.5d libopencv-shape4.5d libopencv-stitching4.5d
  libopencv-superres4.5d libopencv-video4.5d libopencv-videoio4.5d libopencv-videostab4.5d
  libopencv-viz4.5d libopenexr-dev libpng-dev libpq5 libproj22 libraw1394-dev librfttopo1
  libsocket++1 libspatialite7 libsuperlu5 libswresample-dev libswscale-dev libs2z
  libtbb-dev libtesseract4 libtiff-dev libtiffxx5 liburiparser1 libvtk9.1 libxerces-c3.2
  mysql-common proj-data unixodbc-common
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 247 not upgraded.
(ultralitics-env) eeuser@ubuntu: ~$
```

```
>> pip install -U pip
```

```
ca. eeuser@ubuntu: ~  
(ultralytics-env) eeuser@ubuntu:~$ pip install -U pip  
Requirement already satisfied: pip in ./ultralytics-env/lib/python3.10/s  
Collecting pip  
  Downloading pip-24.2-py3-none-any.whl (1.8 MB)  
1.8/1.8 MB 9.8 MB/s eta 0:  
Installing collected packages: pip  
  Attempting uninstall: pip  
    Found existing installation: pip 22.0.2  
    Uninstalling pip-22.0.2:  
      Successfully uninstalled pip-22.0.2  
Successfully installed pip-24.2  
(ultralytics-env) eeuser@ubuntu:~$
```

- Install `ultralitics` pip package with optional dependencies (Around 5 mins)

```
>> pip install ultralytics[export]
```

```
eeuser@ubuntu: ~$ pip install ultralytics[export]
Collecting ultralytics[export]
  Downloading ultralytics-8.2.86-py3-none-any.whl.metadata (41 kB)
Collecting numpy<2.0.0,>=1.23.0 (from ultralytics[export])
  Downloading numpy-1.26.4-cp310-cp310-manylinux_2_17_aarch64.manylinux2014_aarch64.manylinux2019_aarch64.whl (17.5 MB)
Collecting matplotlib>=3.3.0 (from ultralytics[export])
  Downloading matplotlib-3.9.2-cp310-cp310-manylinux_2_17_aarch64.manylinux2014_aarch64.manylinux2019_aarch64.whl (11.1 MB)
```

```
onnx-1.16.2 opencv-python-4.10.0.84 openvino-2024.3.0 openvino-telemetry-2024.1.0
-3.3.0 optax-0.2.1 orbox-checkpoint-0.6.1 packaging-23.2 pandas-2.2.2 pillow-10.4.0
3.20.3 psutil-6.0.0 py-cpuinfo-9.0.0 pyaml-24.7.0 pyasn1-0.6.0 pyasn1-modules-0.4.0
2.18.0 pyparsing-3.1.4 python-dateutil-2.9.0.post0 pytz-2024.1 pyyaml-6.0.2 request
requests-oauthlib-2.0.0 rich-13.8.0 rsa-4.9 scipy-1.14.1 seaborn-0.13.2 six-1.16.0 s
2 tensorflow-2.15.2 tensorflow-data-server-0.7.2 tensorflow-2.15.1 tensorflow-cpu
1 tensorflow-decision-forests-1.8.1 tensorflow-estimator-2.15.0 tensorflow-hub-0.16
low-io-gcs-filestore-0.37.1 tensorflowjs-4.20.0 tensorflowstore-0.1.64 termcolor-2.4.0
2.15.1 toolz-0.12.1 torch-2.4.0 torchvision-0.19.0 tqdm-4.66.5 typing-extensions-4
a-2024.1 ultralytics-8.2.86 ultralytics-thop-2.0.6 urllib3-2.2.2 werkzeug-3.0.4 wh
wrap-1.14.1 wurlitizer-3.1.1 zipp-3.20.1
(ultralytics-env) eeuser@ubuntu:~$
```

- **Verify Ultralytics Installation**

To verify that Ultralytics has been installed correctly, you can run the following command in the Python interpreter:

```
>> python3.10
import ultralytics
print(ultralytics.__version__)
```

If there are no errors and the version prints successfully, the installation was successful.

```
(ultralytics-env) eeuser@ubuntu:~$ python3.10
Python 3.10.12 (main, Sep 11 2024, 15:47:36) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import ultralytics
>>> print(ultralytics.__version__)
8.2.97
>>> exit()
(ultralytics-env) eeuser@ubuntu:~$
```

- Reboot the device

>> sudo reboot

```
eeuser@ubuntu:~$ sudo reboot
Connection to 192.168.8.104 closed by remote host.
Connection to 192.168.8.104 closed.

C:\Users\abdul>
```

3.7 Install PyTorch 2.3.0 and Torchvision 0.18

The above ultralytics installation will install Torch and Torchvision. However, these 2 packages installed via pip are not compatible to run on Jetson platform which is based on ARM64 architecture. Therefore, we need to manually install a pre-built PyTorch pip wheel and compile/install Torchvision from source.

- Activate the virtual environment using the following command:

>>source ultralytics-env/bin/activate

- Install dependencies

>> sudo apt-get install libopenmpi-dev libopenblas-base libomp-dev -y

```
C:\ eeuser@ubuntu: ~
eeuser@ubuntu:~$ source ultralytics-env/bin/activate
(ultralytics-env) eeuser@ubuntu:~$ sudo apt-get install libopenmpi-dev libopenblas
-base libomp-dev -y
[sudo] password for eeuser:
Reading package lists... Done
```



```

Unpacking libomp-dev:arm64 (1:14.0-55~exp2) ...
Setting up libomp5-14:arm64 (1:14.0.0-1ubuntu1.1) ...
Setting up libopenblas-base:arm64 (0.3.20+ds-1) ...
Setting up libllvm14:arm64 (1:14.0.0-1ubuntu1.1) ...
Setting up libomp-14-dev (1:14.0.0-1ubuntu1.1) ...
Setting up libomp-dev:arm64 (1:14.0-55~exp2) ...
Processing triggers for libc-bin (2.35-0ubuntu3.6) ...
(ultralytics-env) eeuser@ubuntu:~$

```

Note: The wheel files have already been downloaded and placed in the `~/Tutorial1` folder.

– Install Torch 2.3.0

```
>> pip install ~/Tutorial1/torch-2.3.0-cp310-cp310-linux_aarch64.whl
```

- If you would like to install the torch directly from its repo, use the below command instead of the above. If you already have run the above, please skip this

```
>> pip install
```

https://github.com/ultralytics/assets/releases/download/v0.0.0/torch-2.3.0-cp310-cp310-linux_aarch64.whl

```

eeuser@ubuntu: ~
(ultralytics-env) eeuser@ubuntu:~$ pip install https://github.com/ultralytics/assets/releases/download/v0.0.0/torch-2.3.0-cp310-cp310-linux_aarch64.whl
Collecting torch==2.3.0
  Downloading https://github.com/ultralytics/assets/releases/download/v0.0.0/torch-2.3.0-cp310-cp310-linux_aarch64.whl (226.5 MB)
    226.5/226.5 MB 2.4 MB/s eta 0:00:00
Requirement already satisfied: filelock in ./ultralytics-env/lib/python3.10/site-packages (from torch==2.3.0) (3.15.4)

```

```

Installing collected packages: torch
Attempting uninstall: torch
  Found existing installation: torch 2.4.0
  Uninstalling torch-2.4.0:
    Successfully uninstalled torch-2.4.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.
torchvision 0.19.0 requires torch==2.4.0, but you have torch 2.3.0 which is incompatible.
Successfully installed torch-2.3.0
(ultralytics-env) eeuser@ubuntu:~$

```

– Install Torchvision 0.18

```
>> pip install ~/Tutorial1/torchvision-0.18.0a0+6043bc2-cp310-cp310-linux_aarch64.whl
```

- If you would like to install the torch directly from its repo, use the below command instead of the above. If you already have run the above, please skip this.

```
>> pip install
```

https://github.com/ultralytics/assets/releases/download/v0.0.0/torchvision-0.18.0a0+6043bc2-cp310-cp310-linux_aarch64.whl

```
eeuser@ubuntu: ~
(ultralytics-env) eeuser@ubuntu:~$ pip install https://github.com/ultralytics/assets/releases/download/v0.0.0/torchvision-0.18.0a0+6043bc2-cp310-cp310-linux_aarch64.whl
Collecting torchvision==0.18.0a0+6043bc2
  Downloading https://github.com/ultralytics/assets/releases/download/v0.0.0/torchvision-0.18.0a0+6043bc2-cp310-cp310-linux_aarch64.whl (1.4 MB)
    1.4/1.4 MB 6.1 MB/s eta 0:00:00
Requirement already satisfied: numpy in ./ultralytics-env/lib/python3.10/site-pack
Attempting uninstall: torchvision
  Found existing installation: torchvision 0.19.0
  Uninstalling torchvision-0.19.0:
    Successfully uninstalled torchvision-0.19.0
Successfully installed torchvision-0.18.0a0+6043bc2
(ultralytics-env) eeuser@ubuntu:~$
```

3.8 Install onnxruntime-gpu

The [onnxruntime-gpu](#) package hosted in PyPI does not have `aarch64` binaries for the Jetson. So we need to manually install this package. This package is needed for some of the exports.

All different `onnxruntime-gpu` packages corresponding to different JetPack and Python versions are listed [here](#). However, here we will download and install `onnxruntime-gpu 1.18.0` with `Python3.10` support.

Note: The `onnxruntime-gpu` wheel file has already been downloaded and placed in the `~/Tutorial1` folder. Hence the below command is not needed. However if you prefer to download it directly online use the below command:

```
>> wget https://nvidia.box.com/shared/static/48dtuob7meiw6ebgfsfqakc9vse62sg4.whl
-O onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl
```

```
eeuser@ubuntu: ~
(ultralytics-env) eeuser@ubuntu:~$ wget https://nvidia.box.com/shared/static/48dtuob7meiw6ebgfsfqakc9vse62sg4.whl -O onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl
--2024-09-03 10:31:38-- https://nvidia.box.com/shared/static/48dtuob7meiw6ebgfsfqakc9vse62sg4.whl
Resolving nvidia.box.com (nvidia.box.com)... 74.112.186.157
```

```

HTTP request sent, awaiting response... 200 OK
Length: 73862239 (70M) [application/octet-stream]
Saving to: 'onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl'

onnxruntime_gpu-1.18 100%[=====] 70.44M 13.4MB/s in 5.7s

2024-09-03 10:31:46 (12.4 MB/s) - 'onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl' saved [73862239/73862239]

(ultralitics-env) eeuser@ubuntu:~$

```

– Install the downloaded onnxruntime_gpu

```
>> pip install ~/Tutorial1/onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl
```

```

eeuser@ubuntu: ~
(ultralitics-env) eeuser@ubuntu:~$ pip install onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl
Processing ./onnxruntime_gpu-1.18.0-cp310-cp310-linux_aarch64.whl
Collecting coloredlogs (from onnxruntime-gpu==1.18.0)
  Downloading coloredlogs-15.0.1-py2.py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: flatbuffers in ./ultralitics-env/lib/python3.10/site-packages (from onnxruntime-gpu==1.18.0) (24.3.25)

```

Note: `onnxruntime-gpu` will automatically revert back the numpy version to latest. So we need to reinstall numpy to 1.26.4 to fix an issue by executing:

```

Installing collected packages: numpy, humanfriendly, coloredlogs, onnxruntime-gpu
Attempting uninstall: numpy
  Found existing installation: numpy 1.23.5
  Uninstalling numpy-1.23.5:
    Successfully uninstalled numpy-1.23.5
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.
openvino 2024.3.0 requires numpy<2.0.0,>=1.16.6, but you have numpy 2.1.0 which is incompatible.
tensorflow-cpu-aws 2.15.1 requires numpy<2.0.0,>=1.23.5, but you have numpy 2.1.0 which is incompatible.
ultralitics 8.2.86 requires numpy<2.0.0,>=1.23.0, but you have numpy 2.1.0 which is incompatible.
Successfully installed coloredlogs-15.0.1 humanfriendly-10.0 numpy-2.1.0 onnxruntime-gpu-1.18.0
(ultralitics-env) eeuser@ubuntu:~$

```

```
>> pip install numpy==1.26.4
```

```
(ultralitics-env) eeuser@ubuntu:~$ pip install numpy==1.26.4
Collecting numpy==1.26.4
  Using cached numpy-1.26.4-cp310-cp310-manylinux_2_17_aarch64.manylinux2014_aarch64.whl.metadata (62 kB)
Downloading numpy-1.26.4-cp310-cp310-manylinux_2_17_aarch64.manylinux2014_aarch64.whl (14.2 MB)
----- 14.2/14.2 MB 22.8 MB/s eta 0:00:00
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.23.5
    Uninstalling numpy-1.23.5:
      Successfully uninstalled numpy-1.23.5
  Successfully installed numpy-1.26.4
(ultralitics-env) eeuser@ubuntu:~$
```

4.0 Performing the YOLOv8 object detection Inference

Out of all the model export formats supported by Ultralytics, TensorRT delivers the best inference performance when working with NVIDIA Jetson devices and our recommendation is to use TensorRT with Jetson.

4.1 Convert Model to TensorRT and Run Inference

The YOLOv8n model in PyTorch format is converted to TensorRT to run inference with the exported model. The code is provided as below for the export and the inference.

- **Using Python:**

```
from ultralytics import YOLO

# Load a YOLOv8n PyTorch model
model = YOLO("yolov8n.pt")

# Export the model
model.export(format="engine") # creates 'yolov8n.engine'

# Load the exported TensorRT model
trt_model = YOLO("yolov8n.engine")

# Run inference
results = trt_model("https://ultralitics.com/images/bus.jpg")
```

- **Using Cli:**

```
# Export a YOLOv8n PyTorch model to TensorRT format
yolo export model=yolov8n.pt format=engine # creates 'yolov8n.engine'

# Run inference with the exported model
yolo predict model=yolov8n.engine source='https://ultralitics.com/images/bus.jpg'
```

4.2 How to run the python code?

We recommend the second Option i.e. Using Jupyter

1. Create a python script using vi and run it as below code:

```
>> mkdir My-Yolo
```

Create a new Python script

```
>> touch my_yolov8.py
```

Copy and paste the python code

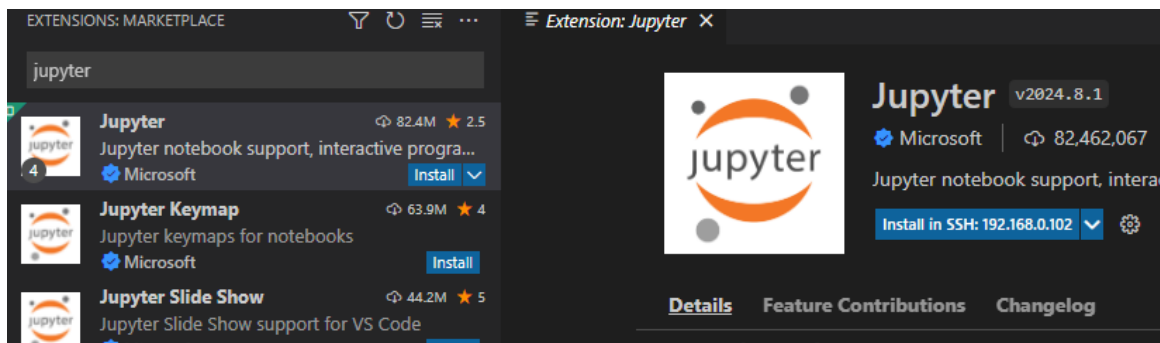
```
>> vi my_yolov8 ( save the file. Press esc and type :w to save; :wq to quit)
```

Execute the code

```
>> python ./my_yolov8
```

2. Using the Jupyter on the board

- Install Jupyter on the the board
- SSH to the board, search Jupyter from the extension
- **Click Install in SSH**



After this, you can create a new Jupyter File, copy and paste the code and then run to do the inference.

4.3 Yolo Benchmarking

Run the following to benchmark the Yolo

Python:

```
from ultralytics import YOLO

# Load a YOLOv8n PyTorch model

model = YOLO("yolov8n.pt")

# Benchmark YOLOv8n speed and accuracy on the COCO8 dataset for all all export
formats

results = model.benchmarks(data="coco8.yaml", imgsz=640)
```

Cli:

```
# Benchmark YOLOv8n speed and accuracy on the COCO8 dataset for all all export formats

>> yolo benchmark model=yolov8n.pt data=coco8.yaml imgsz=640
```

5.0 YOLOv8 object Counting Inference

In this section, students are familiarized with using online tutorials.

Follow either one of the following online tutorials and perform the object counting

1. <https://medium.com/@BasicAI-Inc/yolo-object-counting-step-by-step-guide-yolo-v8-8bdda5cad970>
2. <https://docs.ultralytics.com/guides/object-counting/#real-world-applications>

6.0 Home Exercise: YOLOv8 face Recognition Inference

Study and perform face recognition inference .