### Google Coral Board Setup

1. **Requirements:**

**Note:** Do not power the board or connect any cables until instructed to do so.

* 1. Before you begin, collect the following hardware:
  2. A host computer running Linux (recommended), Mac, or Windows 10
  3. Python 3 installed
  4. One microSD card with at least 8 GB capacity, and an adapter to connect it to your host computer
  5. One USB-C power supply (2-3 A / 5 V), such as a phone charger
  6. One USB-C to USB-A cable (to connect to your computer)
  7. An available Wi-Fi connection (or Ethernet cable)

1. **Flash the board:** Follow this [link](https://coral.ai/docs/dev-board/get-started/#flash-the-board)
2. I**nstall MDT (on Host Computer):** Linux/Mac:

Open a terminal and execute the following commands:

***$ python3 -m pip install --user mendel-development-tool***

***$ echo 'export PATH="$******PATH:$HOME******/.local/bin"' >> ~/.bash\_profile***

***$ source ~/.bash\_profile***

Windows:

***$ python3 -m pip install --user mendel-development-tool***

***$ echo "alias mdt='winpty mdt'" >> ~/.bash\_profile***

***$ source ~/.bash\_profile***

1. **Connect to the board's shell via MDT**

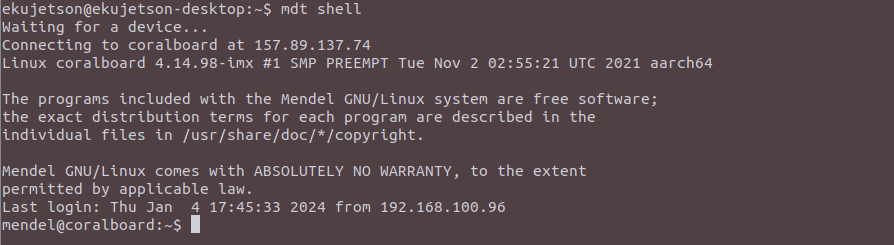
For macOS 10.15 (Catalina) and later, USB-based MDT connections are not supported. Instead, MDT functions over the local network.

1. Connect the board to your computer using a USB-C cable to the board's "OTG" port.
2. On your host computer terminal, verify that MDT detects your board:

$ mdt devices



1. Initiate the device shell using MDT

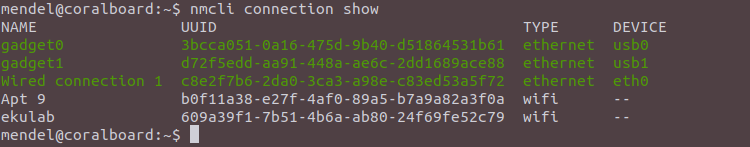


5. **Connect to the internet**

To connect the Coral board to the internet, you can use Ethernet or Wi-Fi depending on your setup.

Follow this [link](https://coral.ai/docs/dev-board/get-started/#connect-internet) for detailed information.

Verify the connection: $ nmcli connection show



6. Update the Mendal software and dependencies:

$ sudo apt-get update  
  
$ sudo apt-get dist-upgrade

7. **Run the Demo App:**

* Dev Board Terminal:

***$ edgetpu\_demo –stream***

If connected via MDT over USB On your desktop (connected to the Dev Board via USB), open **http://192.168.100.2:4664** in a web browser.

This should display a video playing in your browser. The demo uses a pre-recorded video to highlight real-time object detection by the MobileNet model running on the Dev Board.

* If you have a monitor attached to the Dev Board, you can see the demo directly on that screen.

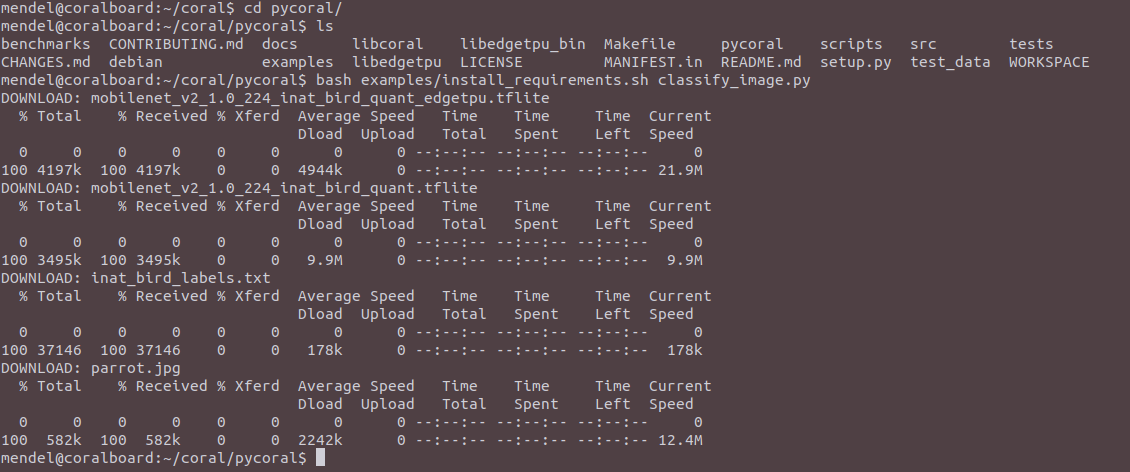
***$ edgetpu\_demo –device***

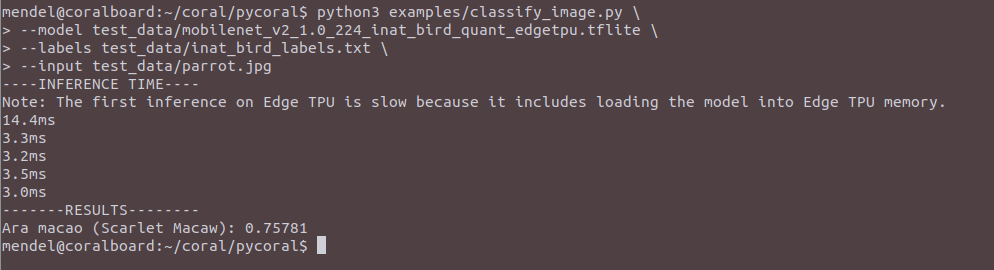
This displays the demo directly on the connected monitor.

**8. Run a model using the PyCoral API:**

Performing an inference on the Edge TPU using the TensorFlow Lite API (assisted by the PyCoral API)





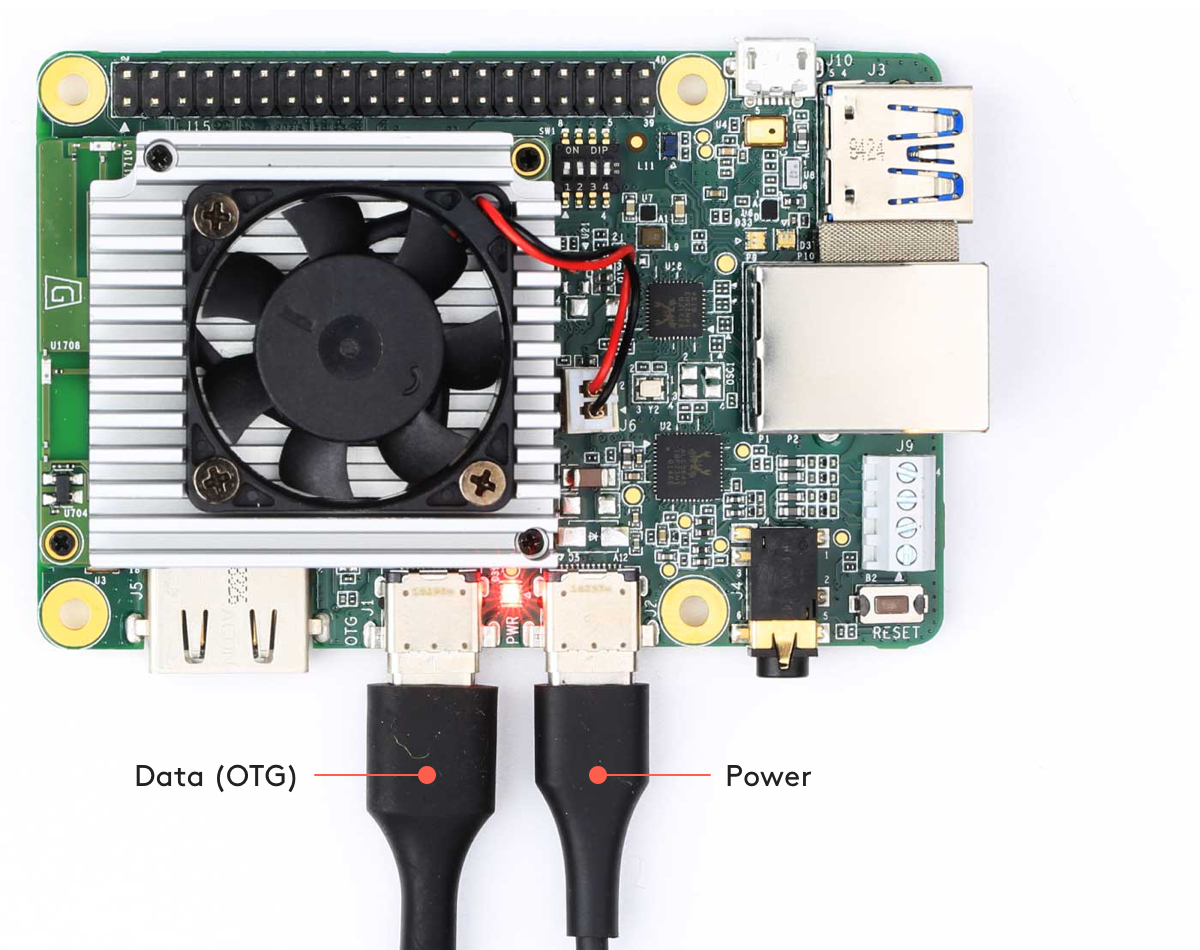


To demonstrate varying inference speeds, the example repeats the same inference five times. It prints the time to perform each inference and then the top classification result (the label ID/name and the confidence score, from 0 to 1.0).

**MacOS Connection Setup**

**Method 1:**

1. Connect over USB OTG - You can also connect to the serial console without the micro-B USB cable, but only if your host computer is Linux or Mac, and your board is fully booted up.



1. Run this command in the macOs Terminal

prasannareddy@Prasi ~ % ls /dev/cu.usbmodemcoralboard1

/dev/cu.usbmodemcoralboard1

1. Connect to the device shown using a serial console program such as screen as follows:

prasannareddy@Prasi ~ % screen /dev/cu.usbmodemcoralboard1 115200

1. The default username and password are both "mendel". When you're done, kill the screen session by pressing CTRL+A, K, and then Y to confirm.

Method 2:

1. Install the mendel-development tool

prasannareddy@Prasi ~ % pip install mendel-development-tool

1. Export the mdt path to the .bash\_profile

prasannareddy@Prasi ~ % export PATH="/Users/prasannareddy/Library/Python/3.10/bin:$PATH"

1. After editing the file, apply the changes with source ~/.bash\_profile.
2. Now, when you enter the command *‘mdt shell*’ you can connect to the coral board.

Note: Make sure your macOS network and coral board network should be the same.

**Upgrade the python version from 3.7 to 3.9 in Debian Systems:**

[**https://linuxize.com/post/how-to-install-python-3-9-on-debian-10/**](https://linuxize.com/post/how-to-install-python-3-9-on-debian-10/)

1. **sudo apt install build-essential zlib1g-dev libncurses5-dev libgdbm-dev libnss3-dev libssl-dev libsqlite3-dev libreadline-dev libffi-dev curl libbz2-dev**
2. Download the latest tar file from the python website

**$ wget** [**https://www.python.org/ftp/python/3.9.1/Python-3.9.1.tgz**](https://www.python.org/ftp/python/3.9.1/Python-3.9.1.tgz)

1. Untar the file

**tar -xf Python-3.9.1.tgz**

1. Navigate to the directory and execute the configure script:

**./configure --enable-optimizations**

The --enable-optimizations option optimizes the Python binary by running multiple tests. This makes the build process slower.

The script runs several checks to make sure all the dependencies on your system are present:

1. Start the Python 3.9 build process:

**make -j 2**

1. When the build process is complete, install the Python binaries by typing:

**sudo make altinstall**

We’re using altinstall instead of install because later we will overwrite the default system python3 binary.

1. Check the python version using: **python3.9 --version**