# Implementation of EfficientDet Model on Jetson Nano

## Objective:

Implement the **.pth** file, trained using an EfficientDet model which detects positive and negative fringes of a signal, on NVIDIA Jetson Nano.

**The entire procedure is expected to take up to 7 hours.

## Components Installed:

1. git
2. htop
3. Python 3
4. pyyaml
5. tqdm
6. Cython
7. NumPy
8. build essential packages
9. cycler
10. kiwisolver
11. pyparsing
12. python date utilities
13. libfree
14. matplotlib
15. gfortran:
16. libopenblas
17. liblapack date utilities
18. libjpeg date utilities
19. pillow
20. pycocotools
21. tensorboard
22. tensorboardX
23. webcolors
24. Pytorch
25. torchvision

## Procedure:

1) Insert your microSD card into your computer using an SD Card reader.
2) Format your microSD Card using SD Card Formatter.
3) Download the Jetson Nano Developer Kit SD Card image.
4) Burn the image onto the microSD card using balenaEtcher.
5) Remove the SD card from your computer and insert it into your Jetson Nano.
6) Setup your jetson nano by attaching a keyboard, mouse, display unit, Ethernet cable (can also be done using usb tethering), and power supply.
7) Follow the ALL the steps given here to complete the booting process.
8) Transfer the .pth file of your machine learning model to the Jetson Nano.
9) Move your test images to folder **/home/jetson/Yet-Another-EfficientDet-Pytorch/test .**
10) Make a new folder named **weights** and copy the .pth model file into this folder.
11) Make the following changes in **/home/jetson/Yet-Another-EfficientDet-Pytorch/efficientdet_test.py:**
    - On Line 20: img_path = ‘test/[image_name].[image_type]’
    - On Line 24: obj_list = [‘positive_fringe’,’negative_fringe’]
    - On Line 51: model.load_state_dict(torch.load(f'weights/[model_filename].pth', map_location='cpu'))
    - On Line 91: cv2.imwrite(f'test/[output_filename]', imgs[i])
12) Right click your mouse and click on “Open Terminal” to open the Terminal.
13) Intsall *git* by typing the command:

    ```
    sudo apt-get install git
    ```
14) Clone the GitHub repository given on this link.
15) Install *htop* using:
    - `sudo apt-get update`
    - `sudo apt-get install htop`
16) Install *Python3 pip* using the command:

    ```
    sudo apt install python3-pip
    ```
17) Install *pyyaml* by using the command:

    ```
    pip3 install –U PyYAML==5.3.1
    ```
18) Install *tqdm* by using:

```
pip3 install tqdm
```

19) Install *Cython* by using the command:

```
pip3 install cython
```

20) Install *NumPy* by using the command:

```
pip3 install –U numpy==1.19.5
```

21) Install *build essential packages* by using the command:

```
sudo apt install build-essential libssl-dev libffi-dev python3-dev
```

22) Install *cycler* by using the command:

```
pip3 install cycler==0.10
```

23) Install *kiwisolver* by using the command:

```
pip3 install kiwisolver==1.3.1
```

24) Install *pyparsing* by using the command:

```
pip3 install pyparsing==2.4.7
```

25) Install *python date utilities* by using the command:

```
pip3 install python-dateutil==2.8.2
```

You can try to skipping instructions from 21-30 and 32 and directly install pycocotools after libjpeg. Not tested. If error occurs, follow all the stated instructions

26) Install *libfree date utilities* by using the command:

```
sudo apt install libfreetype6-dev
```

27) Install *matplotlib date utilities* by using the command:

```
pip3 install  --no-deps matplotlib==3.2.2
```

28) Install *gfortran date utilities* by using the command:

```
sudo apt install gfortran
```

29) Install *libopenblas date utilities* by using the command:

```
sudo apt install libopenblas-dev
```

30) Install *liblapack date utilities* by using the command:

```
sudo apt install liblapack-dev
```

31) Install *libjpeg date utilities* by using the command:

```
sudo apt install libjpeg-dev
```

32) Install *Pillow* by using the command:

```
pip3 install pillow==8.3.2
```

33) Install *pycocotools* by using the command:

```
pip3 install pycocotools
```

34) Install *tensorboard* by using the command:

```
pip3 install tensorboard
```

35) Install *tensorboardX* by using the command:

```
pip3 install tensorboardX
```

36) Install *webcolors* by using the command:

```
pip3 install webcolors
```

37) Install *Pytorch* by using the following commands one by one:

-
  ```
  wget https://nvidia.box.com/shared/static/p57jwntv436lfrd78inwl7iml6p13fzh.whl -O torch-1.9.0-cp36-cp36m-linux_aarch64.whl
  ```

-
  ```
  pip3 install typing-extensions-4.1.1
  ```

-
  ```
  pip3 install torch-1.9.0-cp36-cp36m-linux_aarch64.whl
  ```

38) Install *torchvision* by using the following commands one by one:

-
  ```
  sudo apt-get install libjpeg-dev zlib1g-dev libpython3-dev libavcodec-dev libavformat-dev libswscale-dev
  ```

-
  ```
  git clone --branch v0.9.0 https://github.com/pytorch/vision torchvision
  ```

-
  ```
  sudo apt-get install libopenblas-base libopenmpi-dev
  ```

-
  ```
  cd torchvision
  ```

-
  ```
  export BUILD_VERSION=0.9.0
  ```

-
  ```
  python3 setup.py install --user
  ```

39) Leave the director by using the command:

```
cd ~
```

40) We are now ready to run our model. To monitor our RAM and swap usage during the model implementation, open a new terminal window and open htop by typing:

```
htop
```

htop is not necessary but recommended.

41) On the previous terminal, type the following commands:

-
  ```
  cd Yet-Another-EfficientDet-Pytorch
  ```

-
  ```
  python3 efficientdet_test.py
  ```