PRESENTATION

Face Recognition

From Raspberry pi

GROUP 1

Topics

- Component
- Library
- Roboflow
- Train model
- Result
- Problems
- Member

Component

List:

- Rasberry pi 4
- Camera Module V2
- USB-C Power Supply
- Sd Card
- Mouse
- Monitor Display
- Keyboard
- Neural Compute Stick

Library

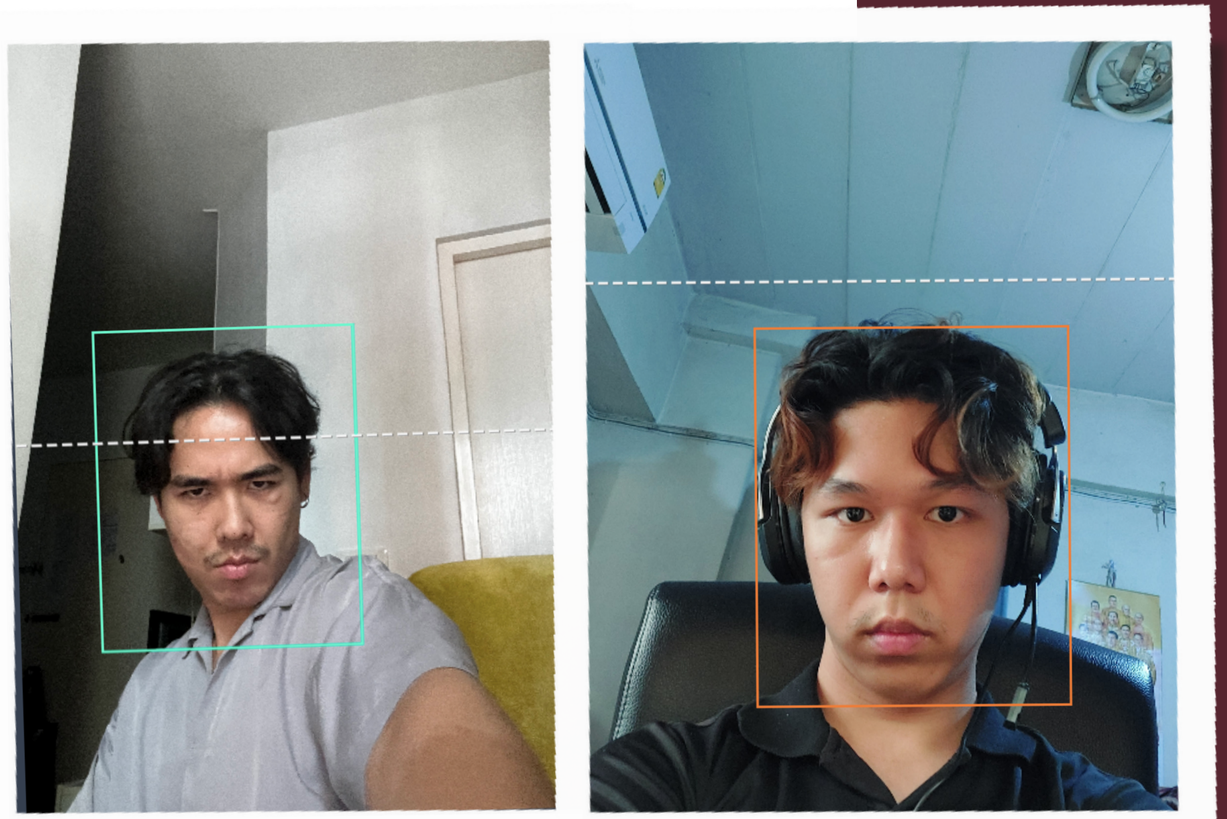
- snapd(cmake)
- python3
- opencv
- numpy
- torch
- utils

RoboFlow

- RoboFlow is use to create labels and detect faces by using images. Also it can increase the data in the dataset in a short time(Additional function)

Steps:

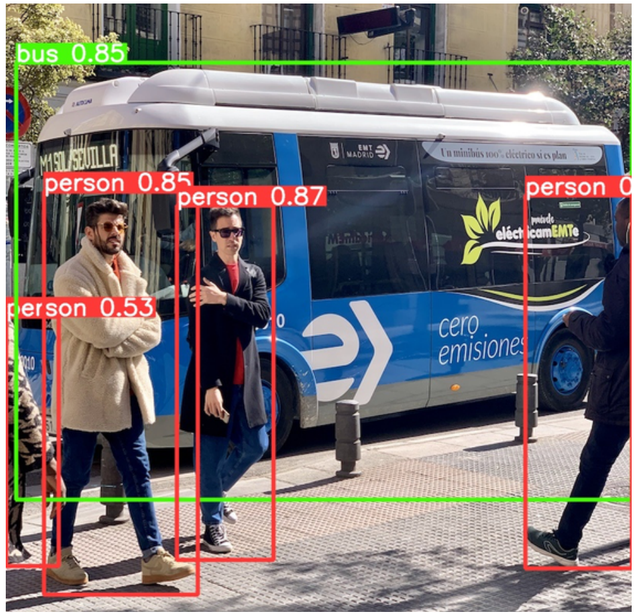
1. Create a workspace
2. Upload the data(photos) into our work space.
3. Cropping a person face in the image.
4. Set all of the cropped image to train.
5. The RoboFlow will duplicate the image making the accuracy increase.



Train model

```
[ ]  !git clone https://github.com/ultralytics/yolov5  # clone
     %cd yolov5
     %pip install -qr requirements.txt  # install

     import torch
     import utils
     display = utils.notebook_init()  # checks

     YOLOv5 🚀 v7.0-172-gc3c1304 Python-3.10.11 torch-2.0.1+cu118 CUDA:0 (Tesla T4, 15102MiB)
     Setup complete ✅ (2 CPUs, 12.7 GB RAM, 23.3/78.2 GB disk)

[ ]  !python detect.py --weights yolov5s.pt --img 640 --conf 0.25 --source data/images

     detect: weights=['yolov5s.pt'], source=data/images, data=data/coco128.yaml, imgsz=[640, 640], co
     YOLOv5 🚀 v7.0-172-gc3c1304 Python-3.10.11 torch-2.0.1+cu118 CUDA:0 (Tesla T4, 15102MiB)

     Downloading https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt to yolov5s.p
     100% 14.1M/14.1M [00:00<00:00, 324MB/s]

     Fusing layers...
     YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients
     image 1/2 /content/yolov5/data/images/bus.jpg: 640x480 4 persons, 1 bus, 67.7ms
     image 2/2 /content/yolov5/data/images/zidane.jpg: 384x640 2 persons, 2 ties, 41.7ms
     Speed: 0.5ms pre-process, 54.7ms inference, 131.6ms NMS per image at shape (1, 3, 640, 640)
     Results saved to runs/detect/exp
```

Firstly, we go to goolge colab. Using yolov5 to train our model by install everything that requirement (PyTorch, numpy, opencv, matplotlib, etc.) Next we create detect.py to test the detection images and result will be show up on runs/detect/exp



Train model

coco.yaml

```
train: ./content/drive/MyDrive/Human
val: /content/drive/MyDrive/Human

nc: 5
names: ['Kaow', 'Mean', 'Phone', 'Saung', 'Tho']
```

Next we go to the file, click yolov5 --> data --> coco.yaml. To train and valid our dataset from roboflow and classify our member group.

Train model

yolov5s.yaml

```yaml
# YOLOv5 🚀 by Ultralytics, AGPL-3.0 license

# Parameters
nc: 5  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, C3, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 6, C3, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, C3, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 3, C3, [1024]],
   [-1, 1, SPPF, [1024, 5]],  # 9
  ]

# YOLOv5 v6.0 head
head:
  [[-1, 1, Conv, [512, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 6], 1, Concat, [1]],  # cat backbone P4
   [-1, 3, C3, [512, False]],  # 13

   [-1, 1, Conv, [256, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 4], 1, Concat, [1]],  # cat backbone P3
   [-1, 3, C3, [256, False]],  # 17 (P3/8-small)

   [-1, 1, Conv, [256, 3, 2]],
   [[-1, 14], 1, Concat, [1]],  # cat head P4
   [-1, 3, C3, [512, False]],  # 20 (P4/16-medium)

   [-1, 1, Conv, [512, 3, 2]],
   [[-1, 10], 1, Concat, [1]],  # cat head P5
   [-1, 3, C3, [1024, False]],  # 23 (P5/32-large)

   [[17, 20, 23], 1, Detect, [nc, anchors]],  # Detect(P3, P4, P5)
  ]
```

Next we go to the file again, click yolov5 --> model --> yolov5s.yaml. This is the classification(CNN) that use in our model and we set the number of classes to be 5 because our classes have 5 classes.

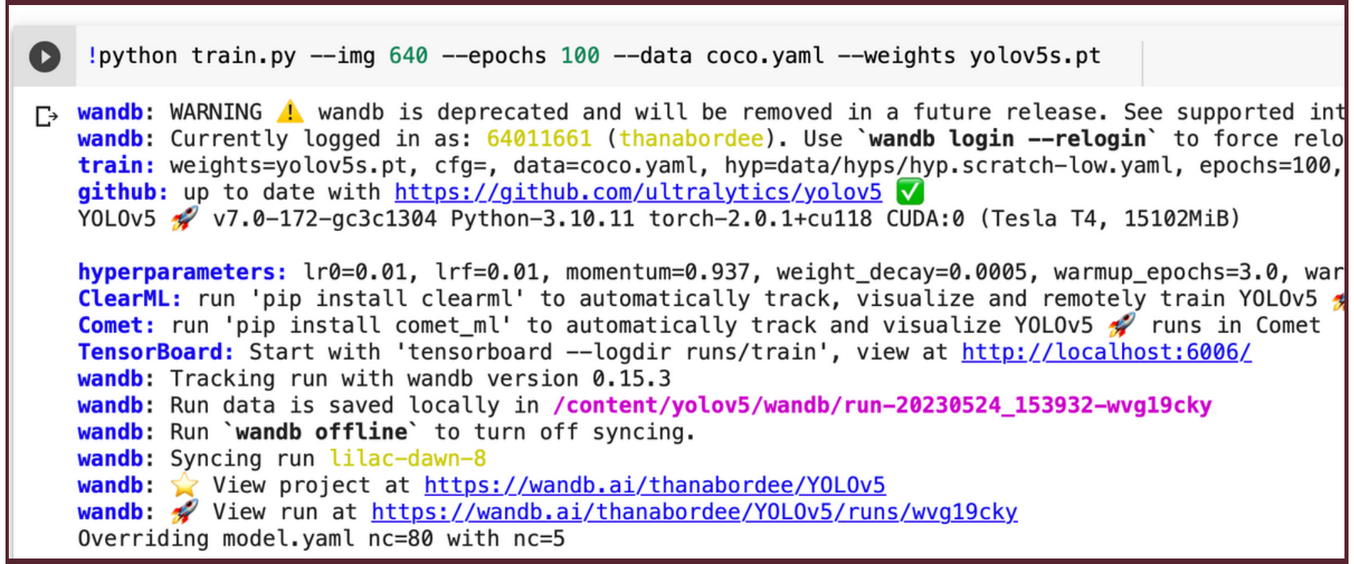
Train model

```
[ ]  # Weights & Biases (optoinal)
     %pip install -q wandb
     import wandb
     wandb.login()
```

```
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
True
```

Before we train, we need to install Weights and Biases by using this code

Train model



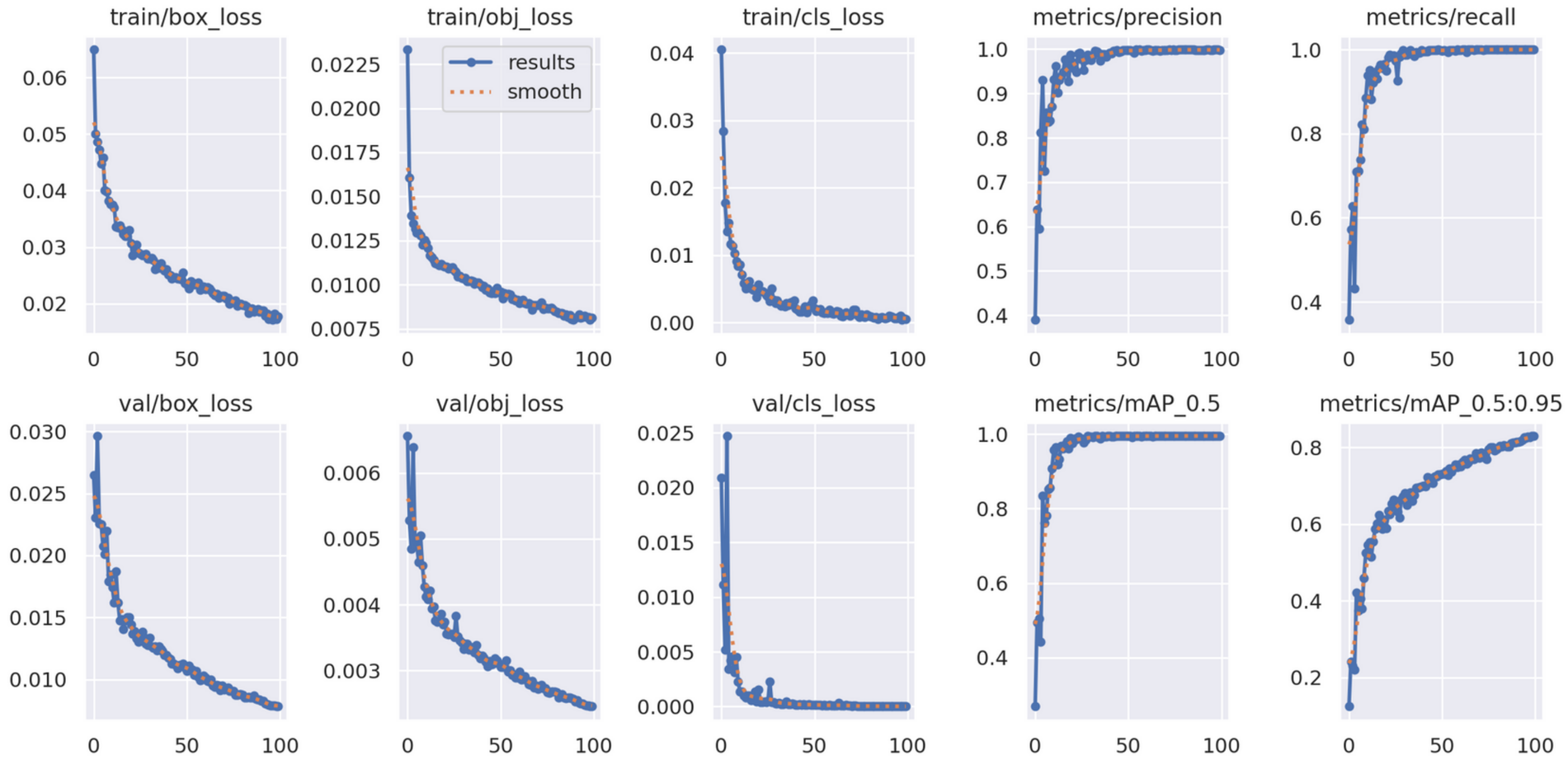
Now the essential part is we train our model by setting img. pixel(640x640) and epoch 100 times. The data will get from coco.yaml and yolov5s.pt that we set from before. We use 2568 images to train and 2754 images to valid.

Train model

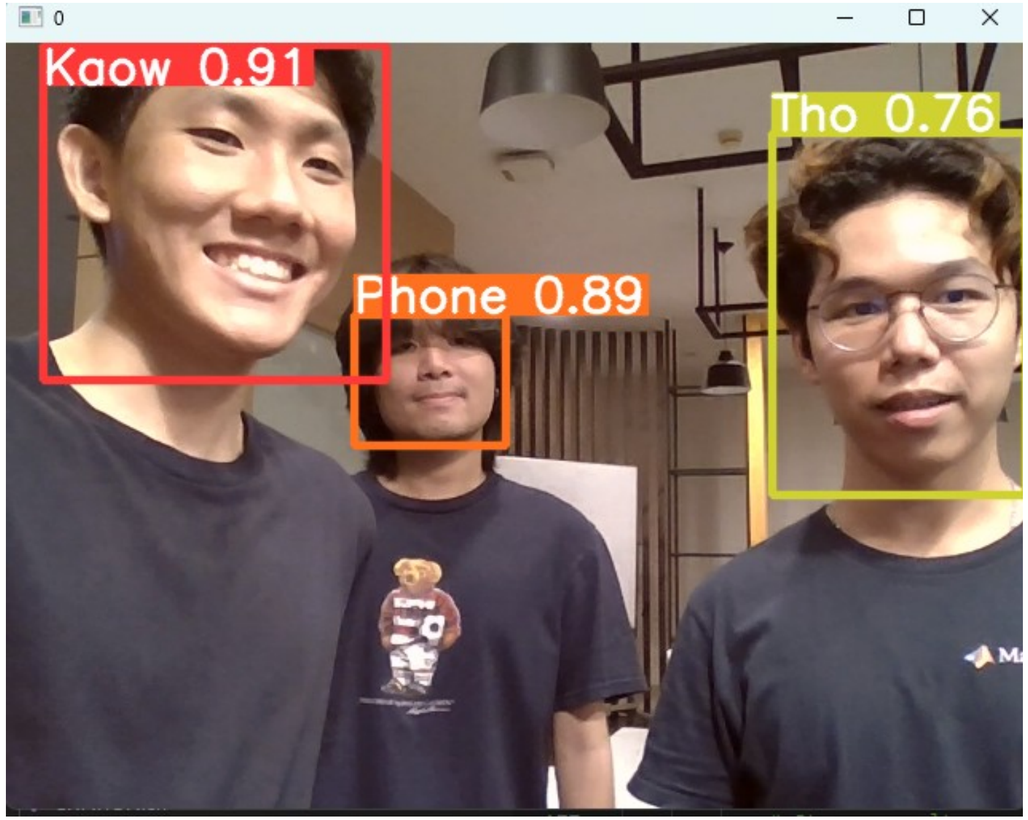
```
Run summary:
           best/epoch 98
         best/mAP_0.5 0.995
    best/mAP_0.5:0.95 0.82967
       best/precision 0.99863
          best/recall 0.99956
      metrics/mAP_0.5 0.995
 metrics/mAP_0.5:0.95 0.82951
    metrics/precision 0.99863
       metrics/recall 0.99957
       train/box_loss 0.01771
       train/cls_loss 0.00053
       train/obj_loss 0.00812
         val/box_loss 0.00782
         val/cls_loss 1e-05
         val/obj_loss 0.00246
                x/lr0 0.0003
                x/lr1 0.0003
                x/lr2 0.0003
```

Finally, we get the best/precision is **0.99863** which very good values that we expected. and we got the best epoch at **98**. Therefore, we use the best epoch to detect our model by download file best.pt and use in python code(this file will show in **runs/train/exp/weights/best.pt**)

Train model


train/box_loss
train/obj_loss
results
smooth
train/cls_loss
metrics/precision
metrics/recall
val/box_loss
val/obj_loss
val/cls_loss
metrics/mAP_0.5
metrics/mAP_0.5:0.95

Result

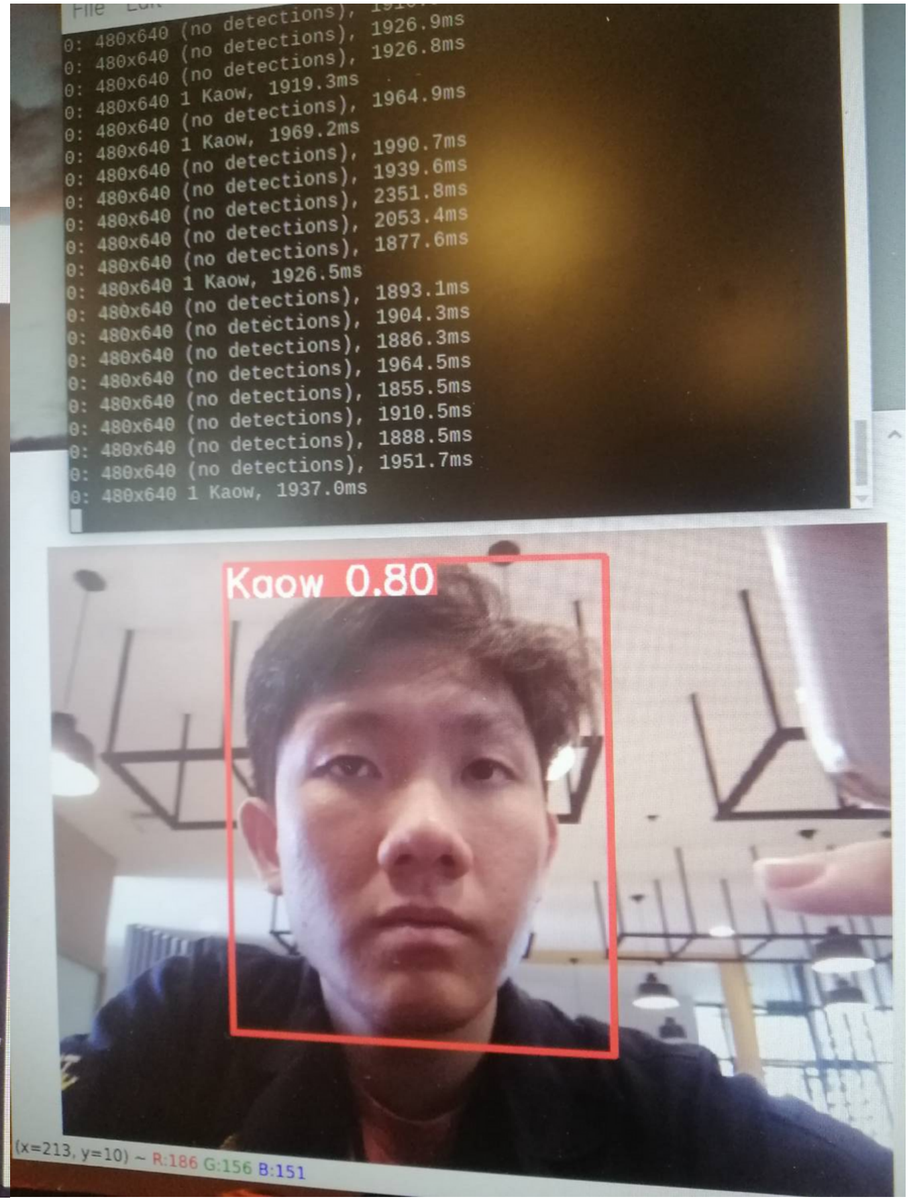


The trained model predicts correct in only some condition with high confident score. But it is sensitive to light and back ground, and get the wrong prediction in some condition.

We believe that the label which is too large and included too much environment such as background, hair, and parts of shirt color make the model worse. So, we should only label eyes, nose, and mouse.

Result

```
File  Edit  Tabs  Help
Make targets are:
  make help - shows this message
  make install - Installs the ncsdk.
  make examples - makes the ncsdk examples.
  make api - installs only the api. Ideal for RPi setup.
  make uninstall - uninstalls the ncsdk.
  make clean - removes targets and intermediate files.
group1@raspberrypi:~/ncsdk $ make install
./uninstall-opencv.sh
OpenCV uninstall starting
[sudo] password for group1:
Reading installer configuration variables from ./ncsdk.conf
WARNING: Package(s) not found: opencv-python
OpenCV uninstall finished

make install starting.
./install.sh
Movidius Neural Compute Toolkit Installation
Your current combination of Linux distribution and distribution v
ersion is not officially supported! Error on line 55.  Will exit
make: *** [Makefile:48: install] Error 1
group1@raspberrypi:~/ncsdk $
```



Problem

1. We use a little data to train (around 400 images). It can work but precision only has 0.6-0.7 which has not high precision. It can detect only if we near camera and has a lot of error
2. We also use raspberry pi OS (32 bit) but it has a lot of error when install library and a lot of bug will occur.
3. We stuck a long time with training model in google colab. it took around 4-5 hours to train the big dataset. Also we got limit using GPU in google colab that the first we didn't know that has a limit per day.
4. The camera is not stable when we put our training model into raspberry pi. we got only 0.5 FPS - 1 FPS. and if we use neural compute stick, it just help a little bit to increase FPS(8 FPS).
5. We can install install openvino which make us can see the NCSDK1 in usb port. But we cannot install mvnc and integrate the neural computer with the raspberry pi
6. The model shows wrong detection for some conditions.

Member

1. 64011661 Thanabordee Charoenjai
2. 64110091 Atiwit Chomchalao
3. 64011463 Napol Uranwate
4. 64011740 Saung Hnin Phyu
5. 64011529 Pattana Teekatananon

END OF PRESENTATION

THANK YOU