**INSTITUTE OF PRINTING TECHNOLOY**

**&**

**GOVERNMENT POLYTECHNIC COLLEGE**

**SHORANUR**

****

**Department Of Computer Engineering**

**2020-2021**

**PROJECT REPORT ON**

**MEDHELPER**

Submitted by:

**MAHESH P** (Reg No. : 19138121) **ANURAG P** (Reg No. : 19138118) **DILSHITH T S** (Reg No. : 19138119) **SUMA M** (Reg No. : 19138125) **SARATH S** (Reg No. : )

GUIDED BY:

**ABDUL KADER K T**

**INSTITUTE OF PRINTING TECHNOLOY & GOVERNMENT POLYTECHNIC COLLEGE**

**SHORANUR, KERALA**

**CERTIFICATE**

This is to certify that the project titled **MED HELPER** that has been presented by **MAHESH P (Reg. No. : 19138121), ANURAG P (Reg. No. : 19138118), DILSHITH T S (Reg. No. : 19138119), SUMA M (Reg. No. : 19138125), SARATH S** **(Reg. No. :18130712),** final year students of COMPUTER ENGINEERING, in partial fulfillment of the requirements for the award of the DIPLOMA IN COMPUTER ENGINEERING under the Directorate of the Technical Education, Kerala, during the year 2021 - 2022

**Guided By Head of Department**

**Internal Examiner External Examiner**

**Place: SHORANUR**

**Date: 06/14/2022**

## ACKNOWLEDGEMENT

First of all I thank the Almighty for showering his infinite blessing throughout my life. I extend my sincere thanks especially to Mrs. **Abdul Kader K T**, head of department for her co- operation and full support .

I would like to express my sincere and propound gratitude to the staff in charge **Mrs. Saani H** and all other supporting staffs for remaining source of inspiration and encouragement throughout to enable me to succeed in the endeavour .

##### MAHESH P (Reg. No. : 19138121) ANURAG P(Reg. No. : 19138118) DILSHITH T S (Reg. No. : 19138119) SUMA M (Reg. No. : 19138125)

##### SARATH S (Reg. No. :18130712)

**COMPUTER ENGINEERING [2021 - 2022]**

**CONTENTS**

[ABSTRACT 5](#_TOC_250005)

1. [INTRODUCTION 6](#_TOC_250004)
2. BEFITS OF PROJECT 7
3. SCOPE OF PROJECT 8
4. APPLICATIONS 9
5. [ORGANISATION OF REPORT 10-11](#_TOC_250003)
6. [TECHNOLOGY TO BE USED 12-15](#_TOC_250002)
7. REQUIREMENT ANALYSIS AND SPECIFICATIONS 16
8. [DATAFLOW DIAGRAM 17](#_TOC_250001)
9. [DETECTION NOTEBOOK 18-24](#_TOC_250000)
10. WORKING 25
11. OUTPUTS 26
12. CONCLUSION 27
13. REFERENCE 28

## ABSTRACT

The **MED-HELPER** is a

# INTRODUCTION

This is an AI and Computer vision based application which can distinguish people if they have worn a mask and if they have worn it correctly or not. The application uses Tensorflow and Pytorch library. The application uses YOLOV5 object detection algorithm and YOLOV5s neural network.

The network has a learning rate of 98% and very low learning loss.

# BENEFITS OF THE PROJECT

The project is very light weight and robust and so it has many features

* can run on almost any device
* light weight so very little computing resource needed
* Great efficiency
* Easy to use

# APPLICATION

The application is mainly used for facemask detection. Hence it can be used in almost any area where there is a chance of people gather together, like shopping malls, theatre, auditoriums etc.

Since the system is so powerful the accuracy is very high and making it highly efficient and accurate

* 1. **SCOPE OF THE PROJECT**

The application of transfer learning on pre-trained models with extensive experimentation over an unbiased dataset resulted in a

highly robust and low-cost system. The identity detection of faces, violating the mask norms further, increases the utility of the system for public benefits.

### ORGANISATION OF REPORT

Based on the outline design of the system requirement in term of input outputprocedures, the technical issues raised during technical feasibility include:

* + Does the necessary technology exists to do what is proposed ?
  + Does the proposed equipment have the technical capacity to hold the datarequired to use in the new system?
  + Adequate responses provided by the proposed system?
  + Is the system flexible enough to facilitate expansion?
  + Is there any technical guarantee of accuracy, reliability, ease of access and data security?
  + The system developers task is to view needed capabilities in light of currently available technology. Or site works hand with high technology. A database has to maintained in order to update and backup data whatever required. To create databases, we use MYSQL server
  + For the acceptance within the organization the following points are important and those are explained according to the topic.
  + Whether the system provides right information to the right place?
  + In current system which is semi computerized with the system the information may be lost in the process of sending one place to other. Thisis mainly due to the human interaction into the process of transferring information’s from one place to another. Whether the new system affect the current user in the system ?
  + The new proposed system will affect the users in the following areas accuracy, efficiency, productivity.

### TECHNOLOGY TO BE USED

#### PyTorch :

**PyTorch** is an open source machine learning library based on the Torch library, used for applications such as computer

vision and natural language processing , primarily developed

by Facebook's AI Research lab (FAIR).[ It is free and open-source software released under the Modified BSD license.

It is a low level API but it has some high level features like

Tensor computation (like NumPy) with strong GPU acceleration Deep neural networks built on a tape-based autograd system



#### TensorFlow :

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Tensorflow is a symbolic math library based on dataflow and differentiable programming.

The API is developed by Google and also has a great community support also

We can build and train model using high level keras API



#### YOLOv5:

YOLOv5 is a family of object detection architectures and models pretrained on the COCO dataset, and represents Ultralytics open- source research into future vision AI methods, incorporating lessons learned and best practices evolved over thousands of hours of

research and development.

#### COLAB:

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary

python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically,

Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs.



#### ANACONDA:

Anaconda is a distribution of the Python and R programming

languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS



### REQUIREMENT ANALYSIS AND SPECIFICATION

**HARDWARE REQUIREMENTS:**

Processor : Intel Core i5

Hard Disk : 256GB (SSD Recommended)

RAM : 4GB

**SOFTWARE REQUIREMENTS:**

Operating System : Windows or Ubuntu Software Tools : Anaconda

Command Prompt or terminal COLAB

TensorFlow Programming Language : Python

### DATAFLOW DIAGRAM

##### Level 0:

FACE MASK DETECTOR

TRAINED MODAL

INPUT

RESULT

**Level 1:**

IMAGE

PRE-PROCESSING

IMAGE

PRE - PROCESSING

GENERATE AND TRAIN FACE MASK

SERIALISE MASK CLASSIFIER

RESULT

Determine “With mask", "No Mask” or

“Improper Mask”

APPLY FACE MASK DETECTOR

FACE MASK DATA

FACES FROM LIVE/IMAGE

/VIDEO

### DETECTION NOTEBOOK

!nvidia-smi

Mon Aug 2 10:38:28 2021

+ +

| NVIDIA-SMI 470.42.01 Driver Version: 460.32.03 CUDA Version: 11.2 |

| + + +

| GPU Name Persistence-M| Bus-Id Disp.A | Volatile Uncorr. ECC |

| Fan Temp Perf Pwr:Usage/Cap| Memory-Usage | GPU-Util Compute M. |

| | | MIG M. |

|===============================+======================+======================|

| 0 Tesla T4 Off | 00000000:00:04.0 Off | 0 |

| N/A 65C P8 11W / 70W | 0MiB / 15109MiB | 0% Default |

| | | N/A |

+ + + +

+ +

| Processes: |

| GPU GI CI PID Type Process name GPU Memory |

| ID ID Usage |

|=============================================================================|

| No running processes found |

+ +

!git clone https://github.com/ultralytics/yolov5 Cloning into 'yolov5'...

remote: Enumerating objects: 8670, done.ote: Counting objects: 100% (384/384), done.ote: Compressing objects: 100% (251/251), done.ote: Total 8670 (delta 235), reused 249 (delta

133), pack-reused 8286

%cd /content/yolov5

!pip install -U -r requirements.txt import torch

from IPython.display import Image

!git clone https://github.com/iAmEthanMai/mask-detection-dataset.git Cloning into 'mask-detection-dataset'...

remote: Enumerating objects: 3502, done.ote: Counting objects: 100% (913/913), done.ote:

Compressing objects: 100% (868/868), done.ote: Total 3502 (delta 24), reused 861 (delta

1. , pack-reused 2589

!python3 train.py --batch 1 --epochs 100 --data /content/mask-detection-dataset/data/dat a.yaml --cfg /content/yolov5/models/yolov5s.yaml --weights '' --device 0

from n params module arguments 0 -1 1 3520 models.common.Focus [3, 32, 3]

1 -1 1 18560 models.common.Conv [32, 64, 3,

2]

2 -1 1 18816 models.common.C3 [64, 64, 1]

3 -1 1 73984 models.common.Conv [64, 128, 3,

2]

4 -1 1 156928 models.common.C3 [128, 128, 3]

5 -1 1 295424 models.common.Conv [128, 256, 3,

2]

6 -1 1 625152 models.common.C3 [256, 256, 3]

7 -1 1 1180672 models.common.Conv [256, 512, 3,

2]

8 -1 1 656896 models.common.SPP [512, 512,

[5, 9, 13]]

9 -1 1 1182720 models.common.C3 [512, 512, 1, False]

10 -1 1 131584 models.common.Conv [512, 256, 1,

1]

* 1. -1 1 0 torch.nn.modules.upsampling.Upsample [None, 2, 'ne arest']
  2. [-1, 6] 1 0 models.common.Concat [1]

13 -1 1 361984 models.common.C3 [512, 256, 1, False]

14 -1 1 33024 models.common.Conv [256, 128, 1,

1]

1. -1 1 0 torch.nn.modules.upsampling.Upsample [None, 2, 'ne arest']
2. [-1, 4] 1 0 models.common.Concat [1]

17 -1 1 90880 models.common.C3 [256, 128, 1, False]

18 -1 1 147712 models.common.Conv [128, 128, 3,

2]

19 [-1, 14] 1 0 models.common.Concat [1]

20 -1 1 296448 models.common.C3 [256, 256, 1, False]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 21 |  | -1 | 1 | 590336 | models.common.Conv | | [256, 256, 3, |
| 2] |  |  |  |  |  | |  |
| 22 | [-1, | 10] | 1 | 0 | models.common.Concat | | [1] |
| 23  False]  24 | [17, 20, | -1  23] | 1  1 | 1182720  21576 | models.common.C3  models.yolo.Detect | | [512, 512, 1,  [3, [[10, 13, |
| 16, 30, | 33, 23], | [30, | 61, | 62, 45, | 59, 119], [116, 90, | 156, | 198, 373, 326]], [128, 256, 5 |
| 12]] |  |  |  |  |  |  |  |

Model Summary: 283 layers, 7068936 parameters, 7068936 gradients, 16.4 GFLOPs

Image sizes 640 train, 640 val Using 0 dataloader workers Logging results to runs/train/exp

Starting training for 100 epochs...

21<00:00, 8.40it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 0/99 | 0.478G | 0.09874 | 0.0612 | 0.033 | 2 |

Class Images 00% 43/43 [00:02<00:00, 20.58it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.00121 | 0.00598 |

0.000131 2.64e-05

18<00:00, 8.65it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 1/99 | 0.508G | 0.09449 | 0.06878 | 0.02862 | 11 |

Class Images 00% 43/43 [00:01<00:00, 23.43it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.000946 | 0.0359 |

0.000114 1.94e-05

17<00:00, 8.85it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 2/99 | 0.508G | 0.08959 | 0.06489 | 0.02437 | 7 |

Class Images 00% 43/43 [00:01<00:00, 25.90it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.341 | 0.0103 |

0.000819 0.000193

18<00:00, 8.69it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 3/99 | 0.508G | 0.08555 | 0.0639 | 0.02003 | 7 |

Class Images 00% 43/43 [00:02<00:00, 18.87it/s]

|  |  |  |  |
| --- | --- | --- | --- |
| Labels | P | R | mAP@.5 mAP@.5:.95: 1 |
| 492 | 0.372 | 0.0368 | 0.0142 0.00405 |

all 85

img\_size

640: 100% 682/682 [01:

17<00:00, 8.81it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 4/99 | 0.64G | 0.0773 | 0.06386 | 0.01739 | 8 |

Class Images 00% 43/43 [00:02<00:00, 16.98it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.729 | 0.0872 |

0.0422 0.0111

17<00:00, 8.81it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 5/99 | 0.753G | 0.07219 | 0.06194 | 0.01667 | 5 |

Class Images 00% 43/43 [00:01<00:00, 23.95it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.768 | 0.0957 |

0.0598 0.0144

16<00:00, 8.94it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 6/99 | 0.753G | 0.06634 | 0.05886 | 0.01613 | 12 |

Class Images 00% 43/43 [00:01<00:00, 24.08it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.762 | 0.146 |

0.0761 0.02

17<00:00, 8.82it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 7/99 | 0.753G | 0.06729 | 0.05539 | 0.01541 | 16 |

Class Images 00% 43/43 [00:01<00:00, 24.24it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.821 | 0.15 |

0.136 0.054

16<00:00, 8.86it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 8/99 | 0.753G | 0.06066 | 0.0562 | 0.01573 | 4 |

Class Images 00% 43/43 [00:01<00:00, 26.63it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.782 | 0.138 |

0.0923 0.0231

16<00:00, 8.89it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 9/99 | 0.753G | 0.05925 | 0.05667 | 0.01462 | 4 |

Class Images 00% 43/43 [00:01<00:00, 24.61it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.842 | 0.162 |

0.163 0.0602

17<00:00, 8.84it/s]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 10/99 | 0.753G | 0.0578 | 0.05279 | 0.01437 | 13 |

Class Images 00% 43/43 [00:01<00:00, 23.24it/s]

all 85

img\_size

640: 100% 682/682 [01:

mAP@.5 mAP@.5:.95: 1

|  |  |  |
| --- | --- | --- |
| Labels | P | R |
| 492 | 0.835 | 0.131 |

0.144 0.0376

.

.

img\_size

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | gpu\_mem | box | obj | cls | labels |
| 99/99 | 0.755G | 0.02518 | 0.03003 | 0.003919 | 7 |

640: 100% 682/682 [01:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 17<00:00, 8.76it/s] |  | | | | | |
| Class | Images | Labels | P | R | mAP@.5 | mAP@.5:.95: 1 |
| 00% 43/43 [00:02<00:00, | 21.39it/s] |  |  |  |  |  |
| all | 85 | 492 | 0.727 | 0.674 | 0.642 | 0.395 |
| 0 | 85 | 92 | 0.642 | 0.587 | 0.592 | 0.277 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 85 | 390 | 0.872 | 0.835 | 0.88 | 0.552 |
| 2 | 85 | 10 | 0.666 | 0.6 | 0.453 | 0.355 |

100 epochs completed in 2.204 hours.

!python3 detect.py --source /content/mask-detection-dataset/data/images/maksssksksss787.p ng --weights /content/yolov5/runs/train/exp/weights/last.pt

detect: weights=['/content/yolov5/runs/train/exp/weights/last.pt'], source=/content/mask- detection-dataset/data/images/maksssksksss787.png, imgsz=640, conf\_thres=0.25, iou\_thres=0.45, max\_det=1000, device=, view\_img=False, save\_txt=False, save\_conf=False, save\_crop=False, nosave=False, classes=None, agnostic\_nms=False, augment=False, visual- ize=False, update=False, project=runs/detect, name=exp, exist\_ok=False, line\_thickness=3, hide\_labels=False, hide\_conf=False, half=False

YOLOv5 🚀 v5.0-339-g53bfcbe torch 1.9.0+cu102 CUDA:0 (Tesla T4, 15109.75MB)

Fusing layers...

Model Summary: 224 layers, 7059304 parameters, 0 gradients, 16.3 GFLOPs

image 1/1 /content/mask-detection-dataset/data/images/maksssksksss787.png: 480x640 1 0, 7 1s, Done. (0.014s)

from google.colab import files files.download('/content/yolov5/runs/train/exp/weights/best.pt')

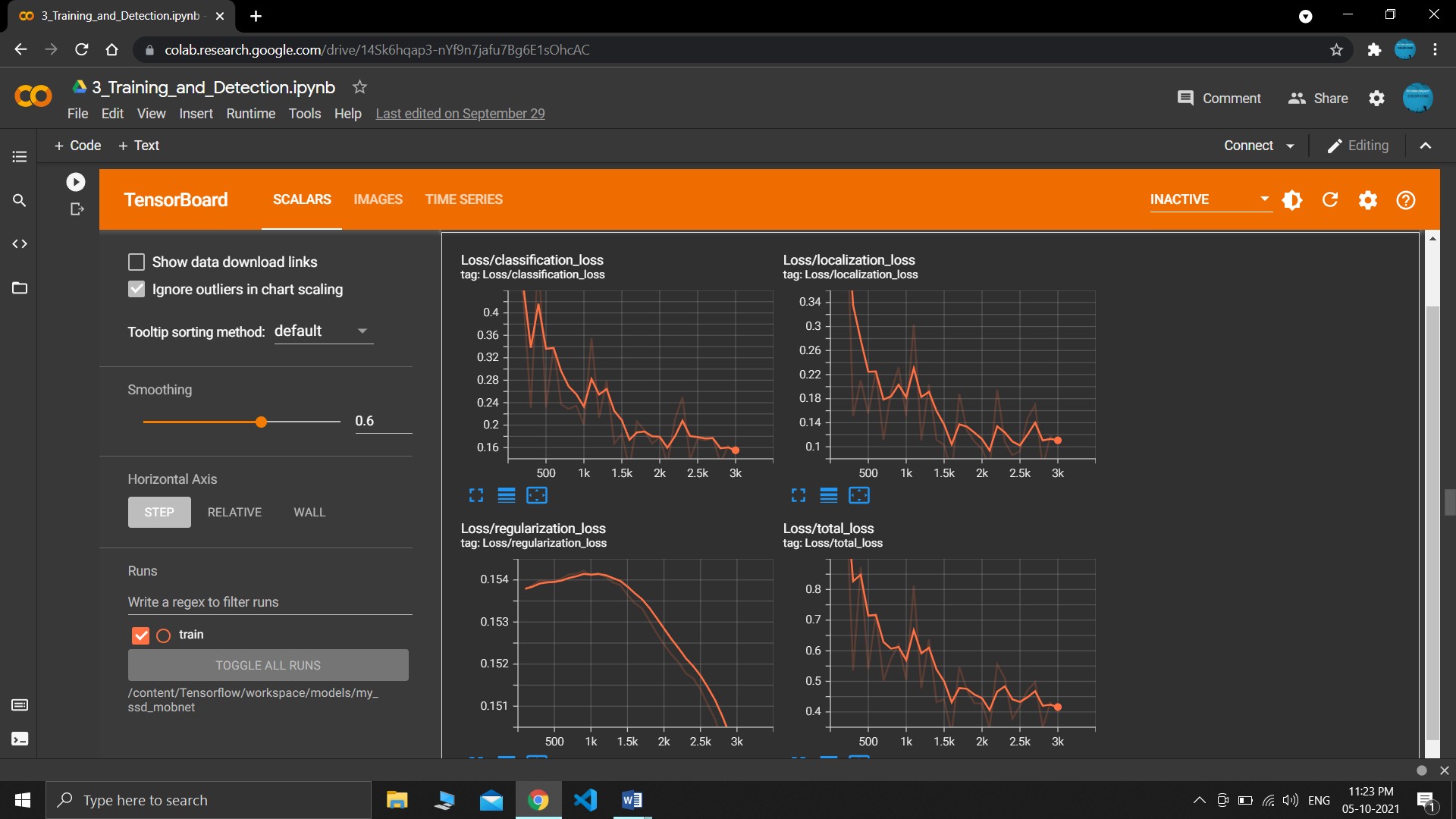
<IPython.core.display.Javascript object>

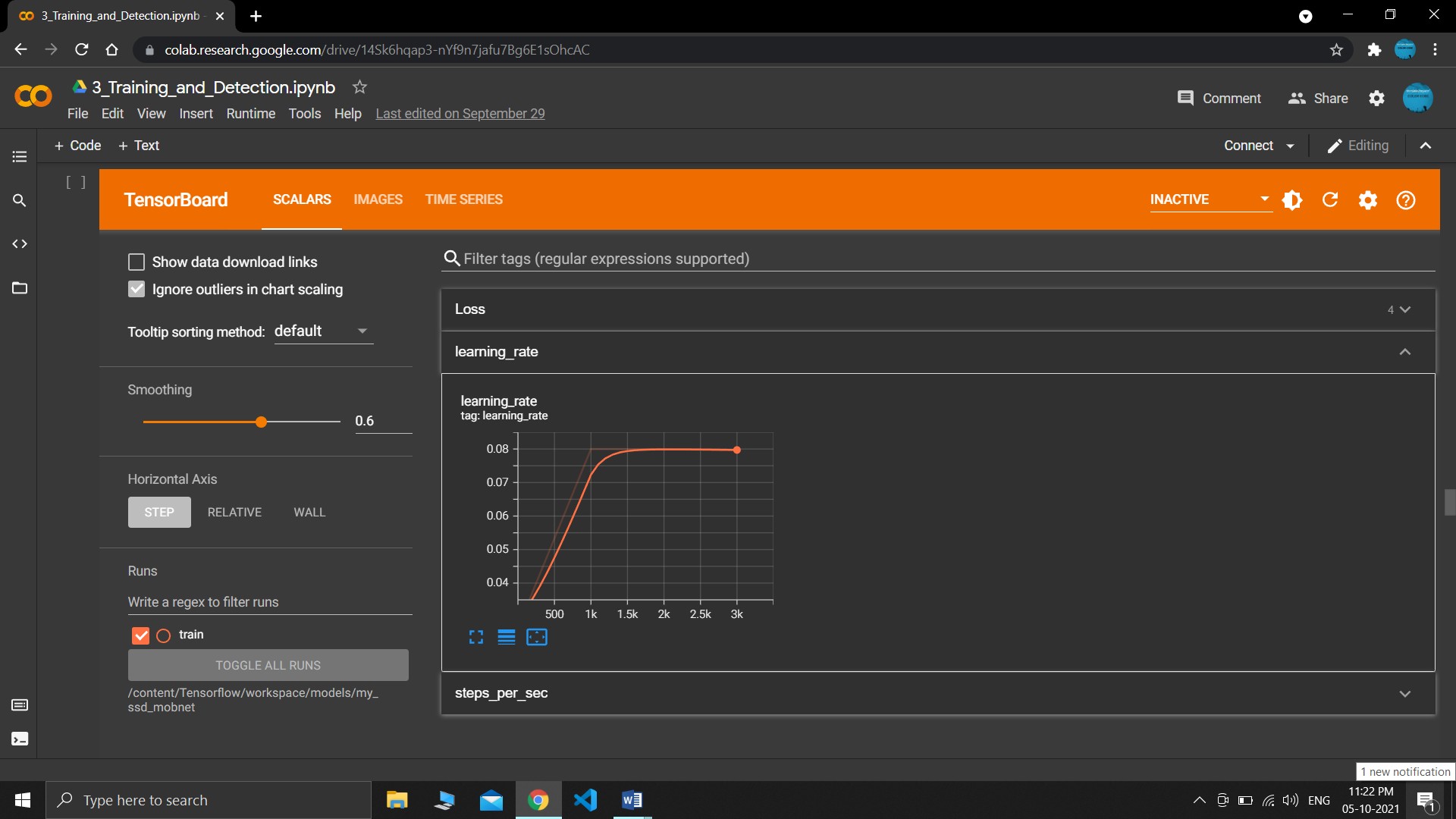
<IPython.core.display.Javascript object>

from google.colab import drive drive.mount('/content/gdrive')

Mounted at /content/gdrive

%cp /content/yolov5/runs/train/exp/weights/best.pt /content/gdrive/My\ Drive





detect: weights=['best.pt'], source=0, imgsz=[640, 640], conf\_thres=0.25, iou\_thres=0.45, max\_det=1000, device=, view\_img=False, save\_txt=False, save\_conf=False, save\_crop=False, nosave=False, classes=None, agnostic\_nms=False, augment=False, visualize=False, up- date=False, project=runs/detect, name=exp, exist\_ok=False, line\_thickness=3, hide\_la- bels=False, hide\_conf=False, half=False

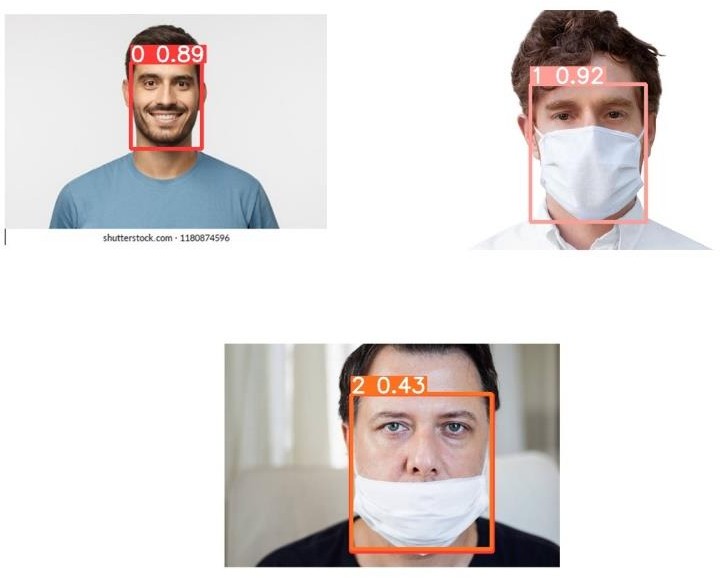
YOLOv5 2021-9-9 torch 1.9.1+cpu CPU

Fusing layers...

Model Summary: 224 layers, 7059304 parameters, 0 gradients, 16.3 GFLOPs 1/1: 0... success (inf frames 640x480 at 30.00 FPS)

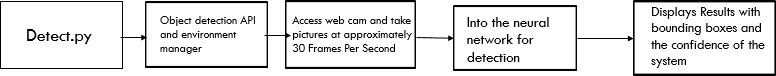
0: 480x640 1 0, Done. (0.880s)

0: 480x640 1 0, Done. (0.614s)

0: 480x640 1 0, Done. (0.518s)

### WORKING OF THE PROJECT

The application uses a custom object trained neural network with fine tuned weights which has a exceptional learning rate and training loss. As the neural network is YOLO architecture the overall performance and detection rate is also great.



As you can see the Neural Network is trained to detect 3 classes, No mask, Mask and Mask Worn improperly.

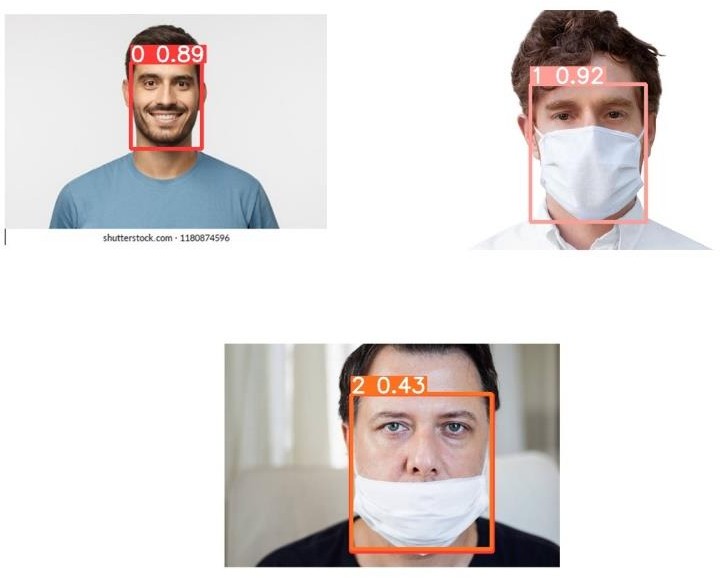
The values in the bounding boxes represent the 3 classes and the confi- dence of the network

The network is capable of identifying the faces with a good accuracy in a crowd and in live view , hence making it fit for deploying in a live feed like a CCTV camera system or other types of monitoring system.

the tensorboard graph of the neural network and in that the graph represents training and validation loss.

The loss starts high but with each epochs the loss is decreasing and the networks starts to learn and improve

### OUTPUT OF THE PROJECT

****



****

* 1. **CONCLUSION**

The face mask detector app is made to run on any de- vices with a great accuracy and efficiency which can help to identify if the people are wearing their mask properly or not.

With the help of the app we can prevent the ongoing pan- demic and save peoples lifes.

### REFERENCES

* [**https://colab.research.google.com/**](https://colab.research.google.com/)
* [**https://www.tensorflow.org/tutorials**](https://www.tensorflow.org/tutorials)
* [**https://pytorch.org/tutorials/**](https://pytorch.org/tutorials/)
* [**https://github.com/ultralytics/yolov5**](https://github.com/ultralytics/yolov5)
* [**https://docs.anaconda.com/anaconda/**](https://docs.anaconda.com/anaconda/)