MINI PROJECT

(2020-21)

Mask Detector

FINAL REPORT

Institute of Engineering & Technology

SUBMITTED BY:-

Ashutosh Kumar(181500149)

Krishan(181500328)

Nitin Goyal(181500432)

Priyansh Agrawal(181500510)

Vinay Kumar Sharma(181500794)

Supervised By
Mr. Mandeep Singh

DECLARATION

We declare that the project report is based on our own work carried out during the course of our study under the supervision of Mr. Mandeep singh. We assert the statements made and conclusions drawn are an outcome of my research work. The work contained in the report is original and has been done by me under the general supervision of my supervisor. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University of India or abroad. We have followed the guidelines provided by the university in writing the synopsis. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.

ACKNOWLEDGEMENT

Our sincere gratitude and thanks towards our project guide Mr. Mandeep Singh. It was only with his backing and support that We could start the project. He provided us all sorts of help and corrected us if ever seemed to make mistakes. We have no such words to express my gratitude. We acknowledge my dearest parents for being such a nice source of encouragement and moral support that helped me tremendously in this aspect. We also declare to the best of my knowledge and belief that the Project Work has not been submitted anywhere else.

Contents

Introduction

Information results displayed via the LCD are mask detector, which means that the person is wearing a mask. The system can detect when the person is not using a mask. Information results displayed via the LCD are no mask, which means that the person is not wearing a mask.

Face Mask Detection Platform uses Artificial Network to recognize is a user is not wearing a mask. The app can be connected to any existing or new IP cameras to detect people without a mask. App user can also add faces and phone numbers to send them an alert in case they are not wearing a mask. If the camera capture an unrecognized face, a notification can be sent out to the administrator.

USE OF THE PROJECT

Using Face Mask Detection System, Hospitals can monitor if their staff is wearing masks during their shift or not. If any health worker is found without a mask, they will receive a notification with a reminder to wear a mask.

Before the coronavirus pandemic, facial-recognition algorithms failed to identify 20-50% of images of people wearing face masks, according to a report from the National Institute of Standards and Technology. But by the end of 2020, it reported a vast improvement in accuracy

1 Project Title: MASK DETECTOR

INTRODUCTION: The Project is named as "Mask Detector". It is named this because in this we have actually used the Pretrained Model of Transfer Learning to Train our Model and then it will help us to do the Mask Detec on.Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. To deal with the lack of data, we make use of a technique called transfer-learning. This reduces the need for data related to the specific task we are dealing with.

2 Problem Statement

In this pandemic situation it is very necessary to ensure that people are following the guideline issued by the government to prevent the spread of Coronavirus. One of them being wearing masks in public places to reduce the transmission by preventing the virus from ge ng into from your nose and mouth. So, this project could be a great ini a ve to ensure that a person is wearing mask and also establishing the safety of others in public places.

3 Project Description

The project consists of Pre-trained model of Transfer Learning that is VGG16 orMobileNetV2, Dataset which contain Mask and Without mask class images and haarcascade file which uses live webcam to easily identify whether a personis Wearing mask or not.

Firstly, we collected the dataset of individuals and divided them into two classes i.e. with mask or without mask. then we imported all the required libraries and pretrained model which we used in this project and a er that we convert our images in 224*224 frame because our pretrained model only works on this specified image size. And then we used some layers and ac va on func on by which we epoch our trained dataset and made our model work with be er accuracy. Then a er that we use this trained model by loading it and then we wrote our image predic on code by which we predicted whether the person is wearing mask or not. We can also by load the same model in video predic on code by which we can access our live webcam with the help of haar cascade filewhere we used live webcam which helped us to detectwhether the person is wearing mask or not through live real me video processing.

4 Objective

This project will help us to monitor the implementa on of COVID-19 guidelines issued by the government in public places. It uses pre trained transfer machine learning model and

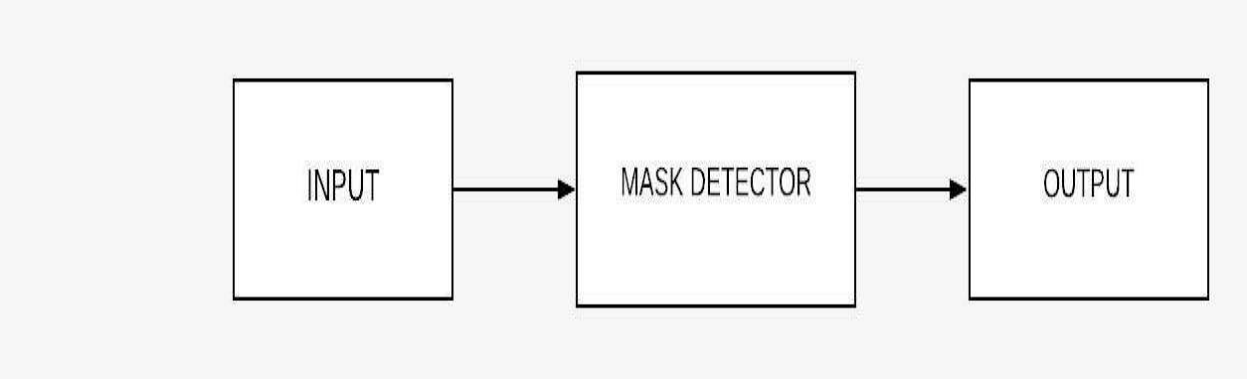
classifica on algorithm to iden fy whether a person is wearing mask or not. This project will also help in ensuring the safety of the individuals in public places.

5.Data modules->

The Project Modules are as follows:

1. **Dataset – It contain a two type of classes which name are with mask or without mask images. By using this dataset we can train our model.**
2. **Classification –by using our train model when we open a camera it will detect/classify our model in two categories which are labeled as mask or without mask.**

Work Flow Diagram



6 Implementation on Methodology

1. First we install the required library according to our project.

2. **Then we check our dataset which contain mask or without mask images.**

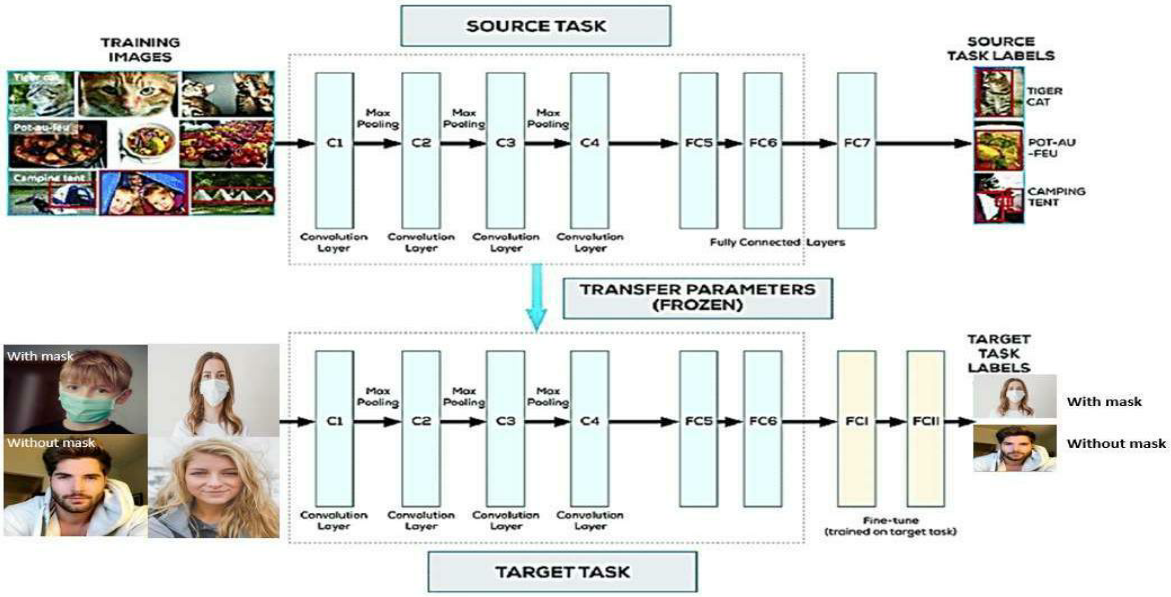
3. **Then we process our data like data preprocessing and pass our data according to our project.**

4. **Then we trained our dataset by using some deep learning algorithm in which we use some layers and ac va on func on.**

5. **Then a er that we check our model accuracy and save our model.**

6. **Then we use our model and implement on real me camera.**

7. **Then finally we got our final result it detects the mask or without mask.**


SOURCE TASK
TRAINING IMAGES
SOURCE TASK LABELS
C1
Max Pooling
C2
Max Pooling
C3
Max Pooling
C4
FC5
FC6
FC7
TIGER CAT
POT-AU -FEU
CAMPING TENT
Convolution Layer
Convolution Layer
Convolution Layer
Convolution Layer
Fully Connected Layers
TRANSFER PARAMETERS (FROZEN)
With mask
Without mask
TARGET TASK LABELS
FCI
FCII
With mask
Without mask
Fine-tune (trained on target task)
TARGET TASK

7 Technologies to be used

So ware Pla orm:

a) **Front-end**

UI with built in camera which captures the live video to detect whether an individual is wearing mask or without mask.

b) **Back-end:**

Pre-trained model on the basis of transfer learning and classifica on algorithm which is trained to detect whether the person is wearing mask or not.

Hardware Pla orm

RAM: at least 4GB,

Hard Disk: 500 GB,

OS: Windows 10 or Linux,

Editor: Jupyter Notebook or Spyder

Browser: Google chrome

8 Advantages of this Project

Mask Detector.

Limits the Spread of coronavirus.

Ensures safety of people in public places.

Helps government to regulate COVID-19 guidelines.

9 Future Scope and further enhancement of the Project

With further improvements these types of models could be integrated with CCTV or other types cameras to detect and iden fy people without masks. With the prevailing worldwide situa on due to COVID-19 pandemic, these types of systems would be very suppor ve for many kinds of ins tu ons around the worldit easier to deploy the model to embedded systems (Raspberry Pi, Google Coral, etc.).This system can therefore be used in real- me applica ons which require face-mask detec on for safety purposes due to the outbreak of Covid-19. This project can be integrated with embedded systems for applica on in airports, railway sta ons, offices, schools, and public places to ensure that public safety guidelines are followedProject Repository Loca on

Features of the project

Automatically Send Alert

Send alert to the faces which are recognized, also set the rate of sending the alerts and detection of faces.

face mask detection
Multi-Channel Recognition

Attach multiple cameras in a few minutes and enable all the cameras to access the AI capability of recognizing faces.

face mask detection
No new hardware to install

The system can work on any existing RTSP camera without the installation of any new cameras. Most of the hospitals and airports have IP cameras installed and RTSP-enabled.

Use cases

Airports

The Face Mask Detection System can be used at airports to detect travelers without masks. Face data of travelers can be captured in the system at the entrance. If a traveler is found to be without a face mask, their picture is sent to the airport authorities so that they could take quick action. If the person's face is already stored, like the face of an Airport worker, it can send the alert to the worker's phone directly.

Hospitals

Using Face Mask Detection System, Hospitals can monitor if their staff is wearing masks during their shift or not. If any health worker is found without a mask, they will receive a notification with a reminder to wear a mask. Also, if quarantine people who are required to wear a mask, the system can keep an eye and detect if the mask is present or not and send notification automatically or report to the authorities.

Offices

The Face Mask Detection System can be used at office premises to detect if employees are maintaining safety standards at work. It monitors employees without masks and sends them a reminder to wear a mask. The reports can be downloaded or sent an email at the end of the day to capture people who are not complying with the regulations or the requirements.

References

Website:

1. h ps://analy csindiamag.com/a-prac cal-guide-to-implement-transfer-learning-in-tensorflow/

2. h ps://docs.opencv.org/2.4/modules/highgui/doc/ reading_and_wri ng_images_and_video.html

3. h ps://docs.opencv.org/3.4/d7/d8b/tutorial_py_face_detec on.html

4. h ps://youtu.be/Ax6P93r32KU

5. h ps://github.com/chandrikadeb7/Face-Mask-Detec on

6. h ps://tryolabs.com/blog/2020/07/09/face-mask-detecon-in-street-camera-v ideo-streams-using-ai-behind-the-curtain/

7. h ps://developer.nvidia.com/blog/implemen ng-a-real-me-ai-based-face-mask-detector-applica on-for-covid-19/

3.1 Books:

1. h p://web.stanford.edu/class/cs20si/lectures/march9guestlecture.pdf

2. h ps://arxiv.org/abs/1801.04381

3. h ps://www.pdfdrive.com/tensorflow-books.html

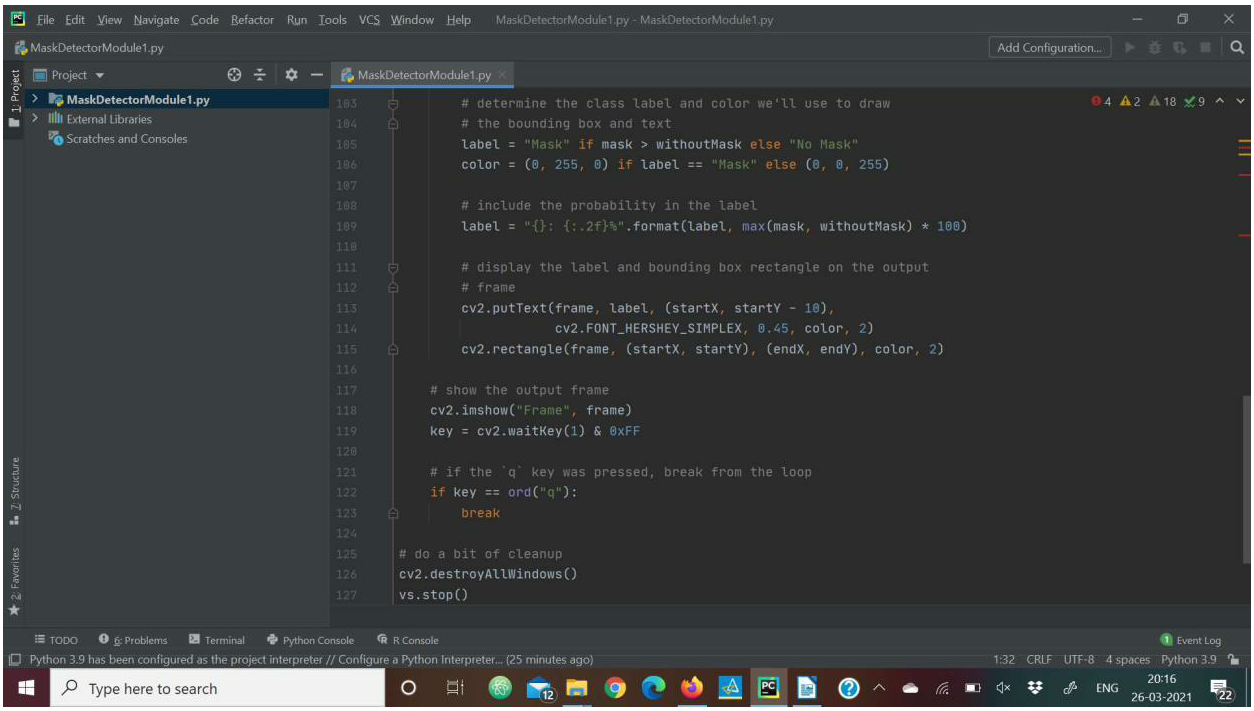
4. h p://uu.diva-portal.org/smash/get/diva2:601707/FULLTEXT01.pdf

5. h ps://www.researchgate.net/publica on/

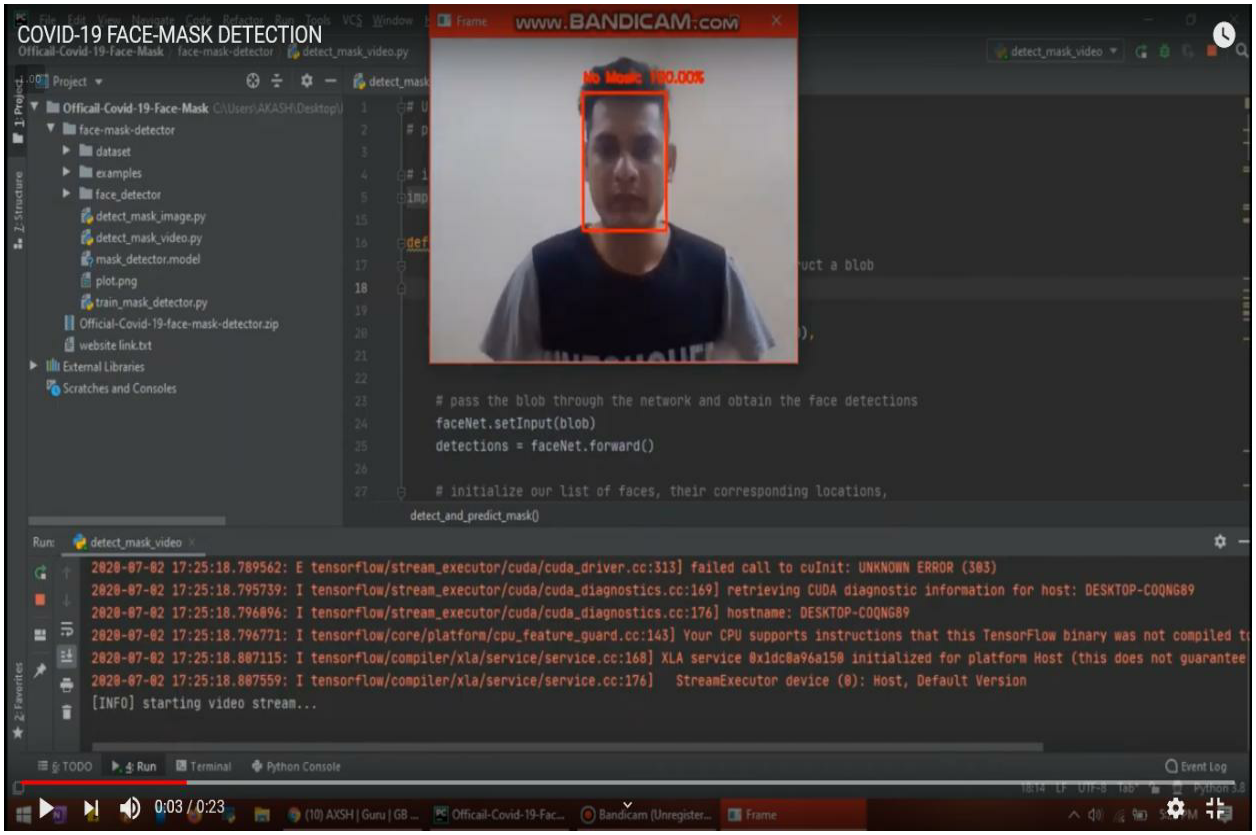
329557776_A_Haar_Classifier_Based_Call_Number_Detec on_and_Coun ng_Method_for_Libr ary_Books

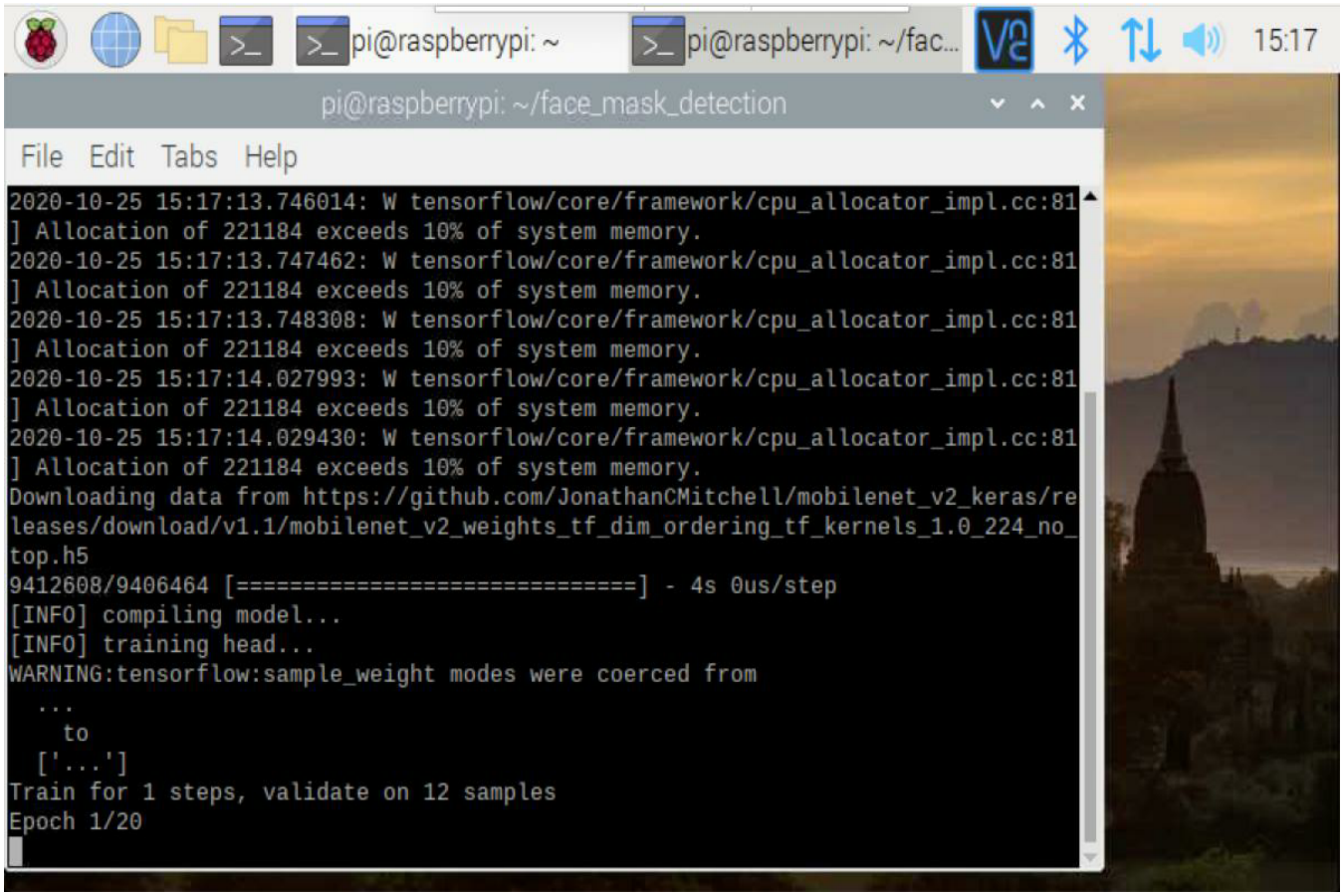
6. h ps://www.researchgate.net/publica on/227173993_The_Performance_of_the_Haar_Cascade_Classifiers_Applied_to_the_Face_and_Eyes_Detec on
7. h ps://ieeexplore.ieee.org/document/8888092
8. h ps://link.springer.com/chapter/10.1007/978-3-540-73007-1_85

Screen Shots


File Edit View Navigate Code Refactor Run Tools VCS Window Help MaskDetectorModule1.py - MaskDetectorModule1.py
MaskDetectorModule1.py
Add Configuration...
Project
MaskDetectorModule1.py
External Libraries
Scratches and Consoles
MaskDetectorModule1.py
# determine the class label and color we'll use to draw
# the bounding box and text
label = "Mask" if mask > withoutMask else "No Mask"
color = (0, 255, 0) if label == "Mask" else (0, 0, 255)
# include the probability in the label
label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
# display the label and bounding box rectangle on the output
# frame
cv2.putText(frame, label, (startX, startY - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
# show the output frame
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
# if the `q` key was pressed, break from the loop
if key == ord("q"):
break
# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
TODO
6: Problems
Terminal
Python Console
R Console
Event Log
Python 3.9 has been configured as the project interpreter // Configure a Python Interpreter... (25 minutes ago)
1:32 CRLF UTF-8 4 spaces Python 3.9
Type here to search
ENG
20:16
26-03-2021

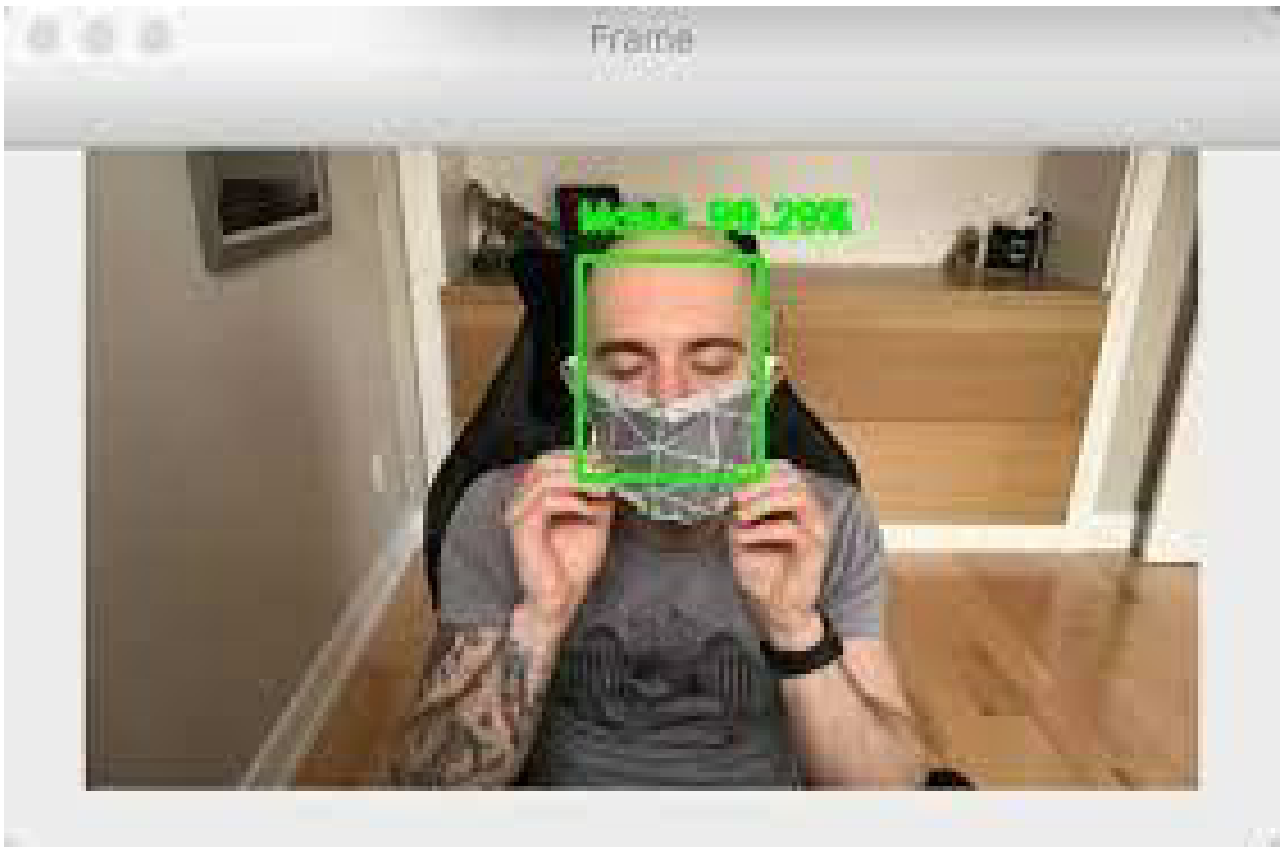

mask

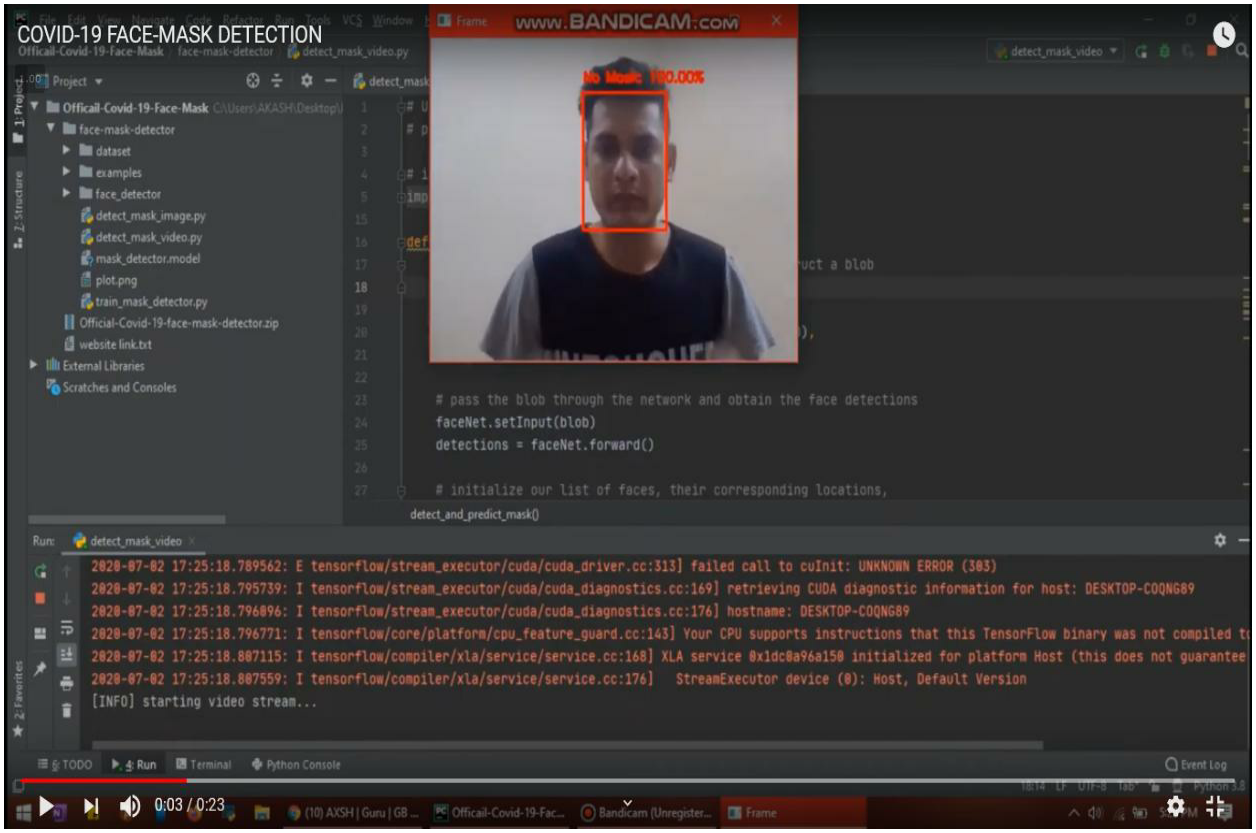

COVID-19 FACE-MASK DETECTION
www.BANDICAM.com
No Mask: 100.00%
Officail-Covid-19-Face-Mask
face_mask_detector
dataset
examples
face_detector
detect_mask_image.py
detect_mask_video.py
mask_detector.model
plot.png
train_mask_detector.py
Official-Covid-19-face-mask-detector.zip
website link.txt
External Libraries
Scratches and Consoles
# pass the blob through the network and obtain the face detections
faceNet.setInput(blob)
detections = faceNet.forward()
# initialize our list of faces, their corresponding locations,
detect_and_predict_mask()
Run: detect_mask_video
2020-07-02 17:25:18.789562: E tensorflow/stream_executor/cuda/cuda_driver.cc:313] failed call to cuInit: UNKNOWN ERROR (303)
2020-07-02 17:25:18.795739: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-COQNG89
2020-07-02 17:25:18.796096: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-COQNG89
2020-07-02 17:25:18.796771: I tensorflow/core/platform/cpu_feature_guard.cc:143] Your CPU supports instructions that this TensorFlow binary was not compiled t
2020-07-02 17:25:18.807115: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x1dc8a96a150 initialized for platform Host (this does not guarantee
2020-07-02 17:25:18.807559: I tensorflow/compiler/xla/service/service.cc:176] StreamExecutor device (0): Host, Default Version
[INFO] starting video stream...
TODO
Run
Terminal
Python Console
Event Log
0:03 / 0:23


pi@raspberrypi: ~
pi@raspberrypi: ~/fac...
15:17
pi@raspberrypi: ~/face_mask_detection
File Edit Tabs Help
2020-10-25 15:17:13.746014: W tensorflow/core/framework/cpu_allocator_impl.cc:81
] Allocation of 221184 exceeds 10% of system memory.
2020-10-25 15:17:13.747462: W tensorflow/core/framework/cpu_allocator_impl.cc:81
] Allocation of 221184 exceeds 10% of system memory.
2020-10-25 15:17:13.748308: W tensorflow/core/framework/cpu_allocator_impl.cc:81
] Allocation of 221184 exceeds 10% of system memory.
2020-10-25 15:17:14.027993: W tensorflow/core/framework/cpu_allocator_impl.cc:81
] Allocation of 221184 exceeds 10% of system memory.
2020-10-25 15:17:14.029430: W tensorflow/core/framework/cpu_allocator_impl.cc:81
] Allocation of 221184 exceeds 10% of system memory.
Downloading data from https://github.com/JonathanCMitchell/mobilenet_v2_keras/re
leases/download/v1.1/mobilenet_v2_weights_tf_dim_ordering_tf_kernels_1.0_224_no_
top.h5
9412608/9406464 [==============================] - 4s 0us/step
[INFO] compiling model...
[INFO] training head...
WARNING:tensorflow:sample_weight modes were coerced from
...
to
['...']
Train for 1 steps, validate on 12 samples
Epoch 1/20

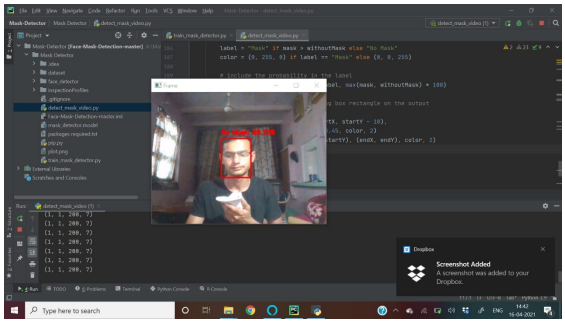
```
COVID-19: Face Mask Detector with OpenCV,
Keras/TensorFlow, and Deep Learning
$ tree --dirsfirst --filelimit 10
.
├── dataset
│   ├── with_mask [690 entries]
│   └── without_mask [686 entries]
├── examples
│   ├── example_01.png
│   ├── example_02.png
│   └── example_03.png
├── face_detector
│   ├── deploy.prototxt
│   └── res10_300x300_ssd_iter_140000.caffemodel
├── detect_mask_image.py
├── detect_mask_video.py
├── mask_detector.model
├── plot.png
└── train_mask_detector.py

5 directories, 10 files
```

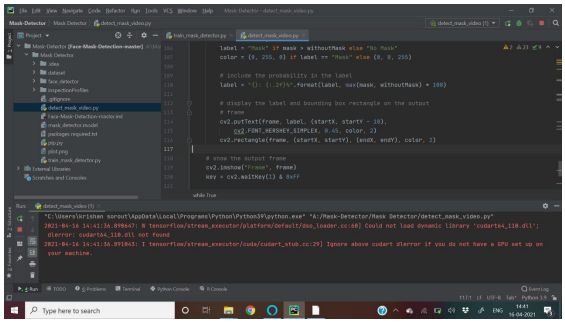

L2C
FaceMask Detection
LIFE2CODING

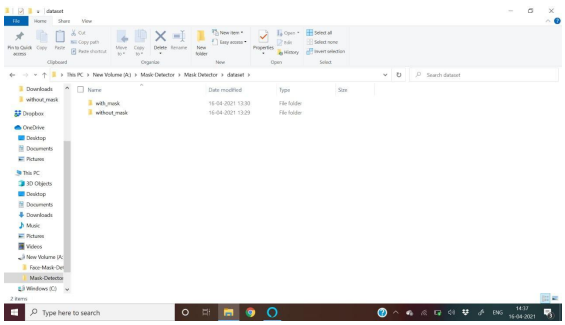

Frame

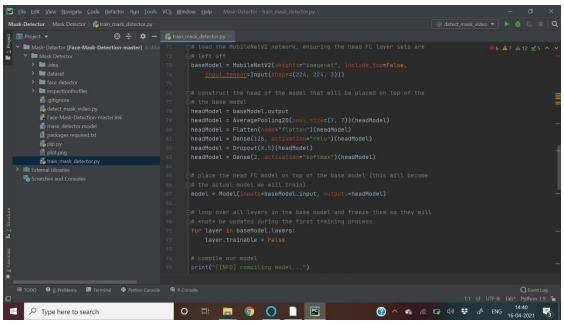

COVID-19 FACE-MASK DETECTION
www.BANDICAM.com
No Mask: 100.00%
Official-Covid-19-Face-Mask
face-mask-detector
dataset
examples
face_detector
detect_mask_image.py
detect_mask_video.py
mask_detector.model
plot.png
train_mask_detector.py
Official-Covid-19-face-mask-detector.zip
website link.txt
External Libraries
Scratches and Consoles
# pass the blob through the network and obtain the face detections
faceNet.setInput(blob)
detections = faceNet.forward()
# initialize our list of faces, their corresponding locations,
detect_and_predict_mask()
Run: detect_mask_video
2020-07-02 17:25:18.789562: E tensorflow/stream_executor/cuda/cuda_driver.cc:313] failed call to cuInit: UNKNOWN ERROR (303)
2020-07-02 17:25:18.795739: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: DESKTOP-COQNG89
2020-07-02 17:25:18.796096: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: DESKTOP-COQNG89
2020-07-02 17:25:18.796771: I tensorflow/core/platform/cpu_feature_guard.cc:143] Your CPU supports instructions that this TensorFlow binary was not compiled t
2020-07-02 17:25:18.807115: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x1dc0a96a150 initialized for platform Host (this does not guarantee
2020-07-02 17:25:18.807559: I tensorflow/compiler/xla/service/service.cc:176] StreamExecutor device (0): Host, Default Version
[INFO] starting video stream...
TODO
Run
Terminal
Python Console
Event Log
0:03 / 0:23

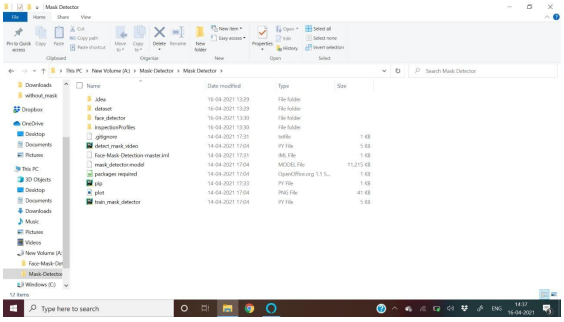

FACE-MASK DETECTION
USING PYTHON AND TENSORFLOW
python
TensorFlow
Always

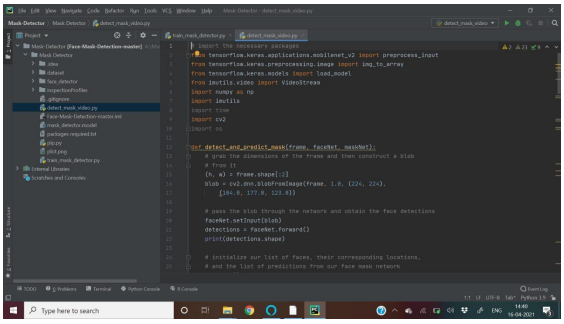

Dropbox
Screenshot Added
A screenshot was added to your Dropbox.

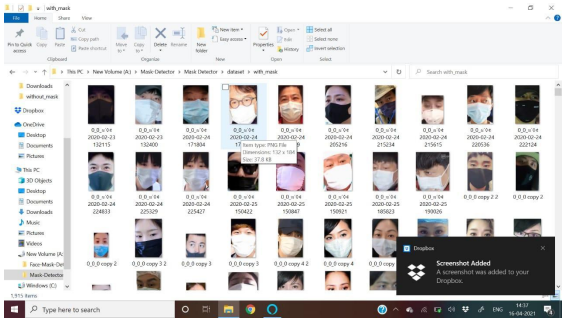


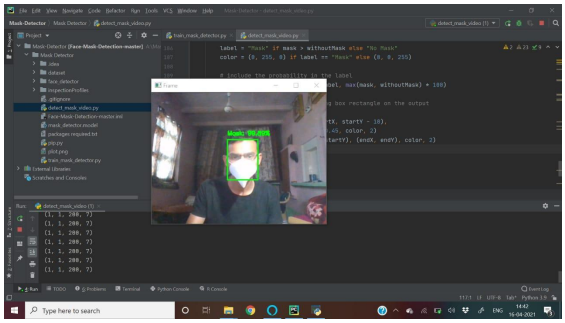


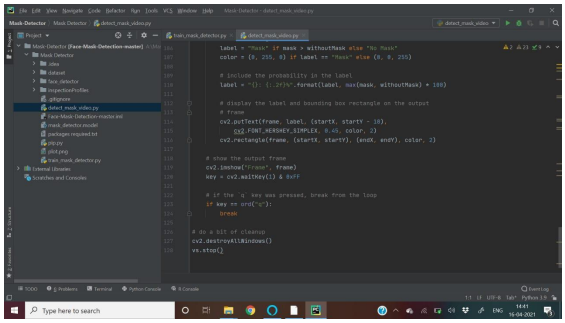


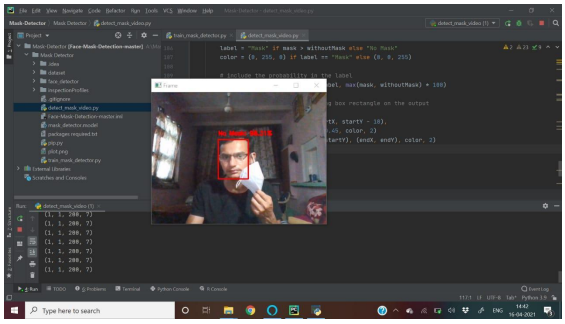


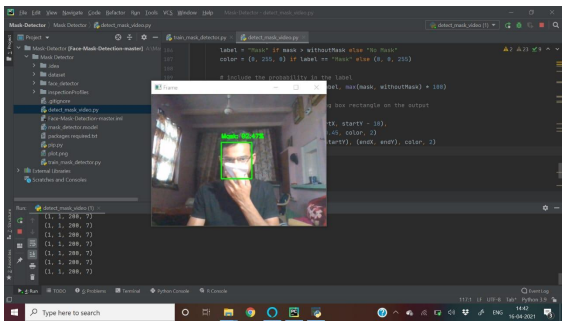


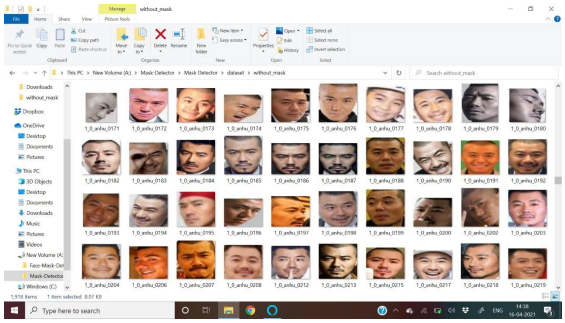


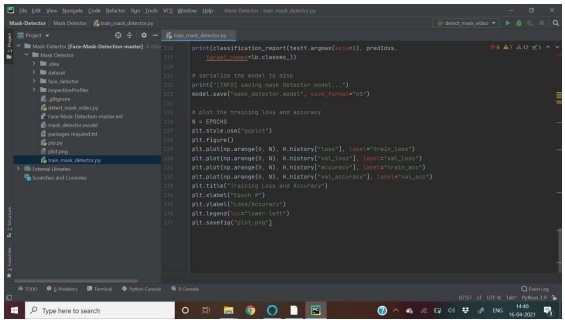


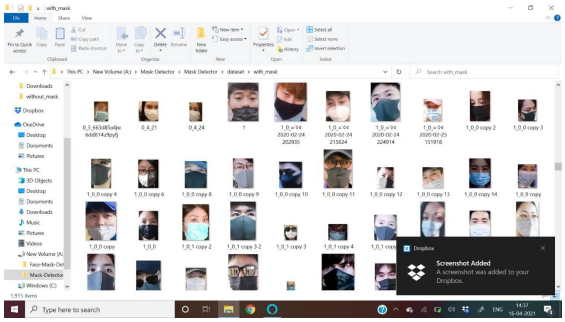


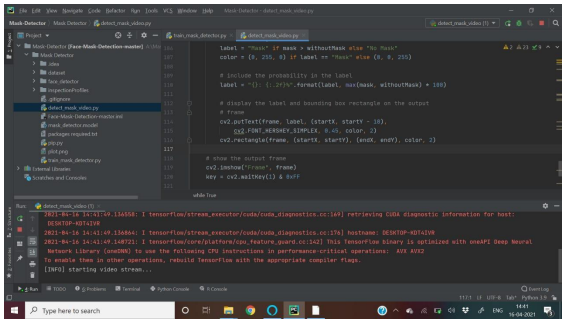


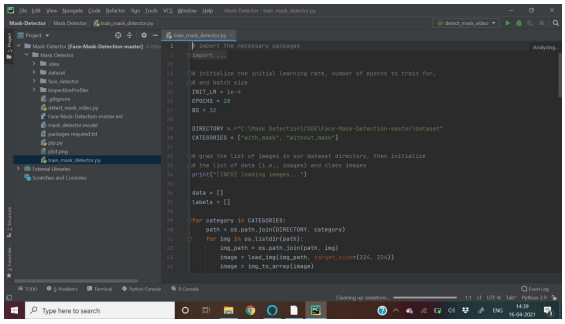


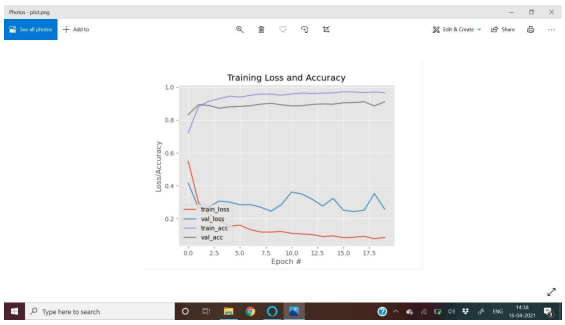


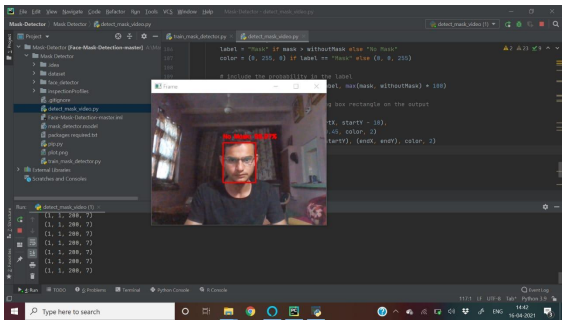



with_mask
Dropbox
Screenshot Added
A screenshot was added to your Dropbox.
Type here to search

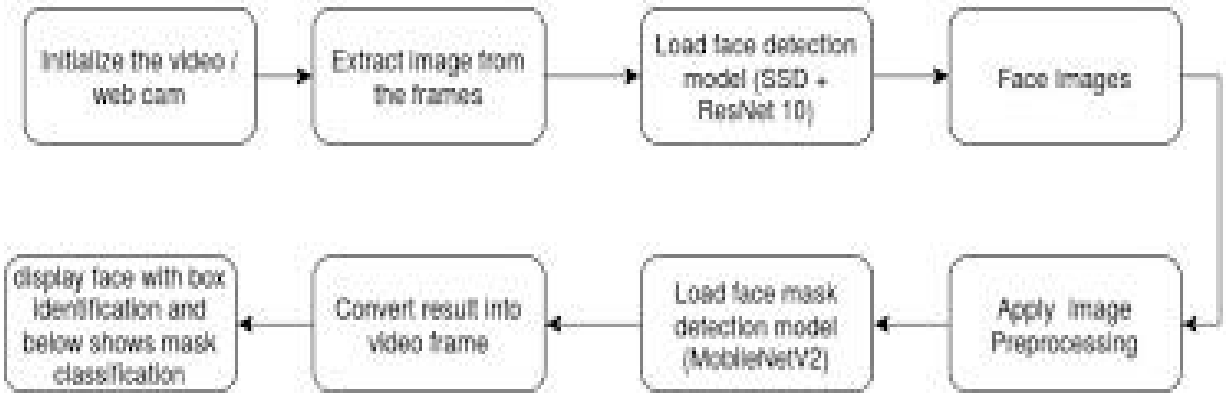

Mask-Detector
detect_mask_video.py
train_mask_detector.py
label = "Mask" if mask > withoutMask else "No Mask"
color = (0, 255, 0) if label == "Mask" else (0, 0, 255)
# include the probability in the label
label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
# display the label and bounding box rectangle on the output
# frame
cv2.putText(frame, label, (startX, startY - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
# show the output frame
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
while True
[INFO] starting video stream...
Type here to search


Mask-Detector
train_mask_detector.py
# import the necessary packages
# initialize the initial learning rate, number of epochs to train for,
# and batch size
INIT_LR = 1e-4
EPOCHS = 20
BS = 32
DIRECTORY = r"C:\Mask Detection\CODE\Face-Mask-Detection-master\dataset"
CATEGORIES = ["with_mask", "without_mask"]
# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class images
print("[INFO] loading images...")
data = []
labels = []
for category in CATEGORIES:
path = os.path.join(DIRECTORY, category)
for img in os.listdir(path):
img_path = os.path.join(path, img)
image = load_img(img_path, target_size=(224, 224))
image = img_to_array(image)
Type here to search


Photos - plot.png
Training Loss and Accuracy
Loss/Accuracy
Epoch #
train_loss
val_loss
train_acc
val_acc
Type here to search



Face Mask detection flow from webcam



```
input: "data"
input_shape {
  dim: 1
  dim: 3
  dim: 300
  dim: 300
}

layer {
  name: "data_bn"
  type: "BatchNorm"
  bottom: "data"
  top: "data_bn"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "data_scale"
  type: "Scale"
  bottom: "data_bn"
  top: "data_bn"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "conv1_h"
  type: "Convolution"
  bottom: "data_bn"
  top: "conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 32
    pad: 3
    kernel_size: 7
    stride: 2
    weight_filler {
      type: "msra"
```

```
      variance_norm: FAN_OUT
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "conv1_bn_h"
  type: "BatchNorm"
  bottom: "conv1_h"
  top: "conv1_h"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "conv1_scale_h"
  type: "Scale"
  bottom: "conv1_h"
  top: "conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "conv1_relu"
  type: "ReLU"
  bottom: "conv1_h"
  top: "conv1_h"
}
layer {
  name: "conv1_pool"
  type: "Pooling"
  bottom: "conv1_h"
  top: "conv1_pool"
  pooling_param {
    kernel_size: 3
    stride: 2
  }
}
layer {
  name: "layer_64_1_conv1_h"
  type: "Convolution"
  bottom: "conv1_pool"
```

```
  top: "layer_64_1_conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 32
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 1
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_64_1_bn2_h"
  type: "BatchNorm"
  bottom: "layer_64_1_conv1_h"
  top: "layer_64_1_conv1_h"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_64_1_scale2_h"
  type: "Scale"
  bottom: "layer_64_1_conv1_h"
  top: "layer_64_1_conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "layer_64_1_relu2"
  type: "ReLU"
  bottom: "layer_64_1_conv1_h"
  top: "layer_64_1_conv1_h"
}
layer {
  name: "layer_64_1_conv2_h"
```

```
  type: "Convolution"
  bottom: "layer_64_1_conv1_h"
  top: "layer_64_1_conv2_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 32
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 1
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_64_1_sum"
  type: "Eltwise"
  bottom: "layer_64_1_conv2_h"
  bottom: "conv1_pool"
  top: "layer_64_1_sum"
}
layer {
  name: "layer_128_1_bn1_h"
  type: "BatchNorm"
  bottom: "layer_64_1_sum"
  top: "layer_128_1_bn1_h"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_128_1_scale1_h"
  type: "Scale"
  bottom: "layer_128_1_bn1_h"
  top: "layer_128_1_bn1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
```

```
}
layer {
  name: "layer_128_1_relu1"
  type: "ReLU"
  bottom: "layer_128_1_bn1_h"
  top: "layer_128_1_bn1_h"
}
layer {
  name: "layer_128_1_conv1_h"
  type: "Convolution"
  bottom: "layer_128_1_bn1_h"
  top: "layer_128_1_conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 128
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 2
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_128_1_bn2"
  type: "BatchNorm"
  bottom: "layer_128_1_conv1_h"
  top: "layer_128_1_conv1_h"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_128_1_scale2"
  type: "Scale"
  bottom: "layer_128_1_conv1_h"
  top: "layer_128_1_conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
```

```
    bias_term: true
  }
}
layer {
  name: "layer_128_1_relu2"
  type: "ReLU"
  bottom: "layer_128_1_conv1_h"
  top: "layer_128_1_conv1_h"
}
layer {
  name: "layer_128_1_conv2"
  type: "Convolution"
  bottom: "layer_128_1_conv1_h"
  top: "layer_128_1_conv2"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 128
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 1
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_128_1_conv_expand_h"
  type: "Convolution"
  bottom: "layer_128_1_bn1_h"
  top: "layer_128_1_conv_expand_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 128
    bias_term: false
    pad: 0
    kernel_size: 1
    stride: 2
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_128_1_sum"
  type: "Eltwise"
```

```
  bottom: "layer_128_1_conv2"
  bottom: "layer_128_1_conv_expand_h"
  top: "layer_128_1_sum"
}
layer {
  name: "layer_256_1_bn1"
  type: "BatchNorm"
  bottom: "layer_128_1_sum"
  top: "layer_256_1_bn1"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_256_1_scale1"
  type: "Scale"
  bottom: "layer_256_1_bn1"
  top: "layer_256_1_bn1"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "layer_256_1_relu1"
  type: "ReLU"
  bottom: "layer_256_1_bn1"
  top: "layer_256_1_bn1"
}
layer {
  name: "layer_256_1_conv1"
  type: "Convolution"
  bottom: "layer_256_1_bn1"
  top: "layer_256_1_conv1"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 256
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 2
    weight_filler {
      type: "msra"
    }
```

```
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_256_1_bn2"
  type: "BatchNorm"
  bottom: "layer_256_1_conv1"
  top: "layer_256_1_conv1"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_256_1_scale2"
  type: "Scale"
  bottom: "layer_256_1_conv1"
  top: "layer_256_1_conv1"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "layer_256_1_relu2"
  type: "ReLU"
  bottom: "layer_256_1_conv1"
  top: "layer_256_1_conv1"
}
layer {
  name: "layer_256_1_conv2"
  type: "Convolution"
  bottom: "layer_256_1_conv1"
  top: "layer_256_1_conv2"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 256
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 1
    weight_filler {
```

```
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_256_1_conv_expand"
  type: "Convolution"
  bottom: "layer_256_1_bn1"
  top: "layer_256_1_conv_expand"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 256
    bias_term: false
    pad: 0
    kernel_size: 1
    stride: 2
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_256_1_sum"
  type: "Eltwise"
  bottom: "layer_256_1_conv2"
  bottom: "layer_256_1_conv_expand"
  top: "layer_256_1_sum"
}
layer {
  name: "layer_512_1_bn1"
  type: "BatchNorm"
  bottom: "layer_256_1_sum"
  top: "layer_512_1_bn1"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_512_1_scale1"
  type: "Scale"
  bottom: "layer_512_1_bn1"
  top: "layer_512_1_bn1"
  param {
```

```
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "layer_512_1_relu1"
  type: "ReLU"
  bottom: "layer_512_1_bn1"
  top: "layer_512_1_bn1"
}
layer {
  name: "layer_512_1_conv1_h"
  type: "Convolution"
  bottom: "layer_512_1_bn1"
  top: "layer_512_1_conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 128
    bias_term: false
    pad: 1
    kernel_size: 3
    stride: 1 # 2
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_512_1_bn2_h"
  type: "BatchNorm"
  bottom: "layer_512_1_conv1_h"
  top: "layer_512_1_conv1_h"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "layer_512_1_scale2_h"
  type: "Scale"
  bottom: "layer_512_1_conv1_h"
```

```
  top: "layer_512_1_conv1_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
layer {
  name: "layer_512_1_relu2"
  type: "ReLU"
  bottom: "layer_512_1_conv1_h"
  top: "layer_512_1_conv1_h"
}
layer {
  name: "layer_512_1_conv2_h"
  type: "Convolution"
  bottom: "layer_512_1_conv1_h"
  top: "layer_512_1_conv2_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 256
    bias_term: false
    pad: 2 # 1
    kernel_size: 3
    stride: 1
    dilation: 2
    weight_filler {
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_512_1_conv_expand_h"
  type: "Convolution"
  bottom: "layer_512_1_bn1"
  top: "layer_512_1_conv_expand_h"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  convolution_param {
    num_output: 256
    bias_term: false
    pad: 0
    kernel_size: 1
    stride: 1 # 2
    weight_filler {
```

```
      type: "msra"
    }
    bias_filler {
      type: "constant"
      value: 0.0
    }
  }
}
layer {
  name: "layer_512_1_sum"
  type: "Eltwise"
  bottom: "layer_512_1_conv2_h"
  bottom: "layer_512_1_conv_expand_h"
  top: "layer_512_1_sum"
}
layer {
  name: "last_bn_h"
  type: "BatchNorm"
  bottom: "layer_512_1_sum"
  top: "layer_512_1_sum"
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
  param {
    lr_mult: 0.0
  }
}
layer {
  name: "last_scale_h"
  type: "Scale"
  bottom: "layer_512_1_sum"
  top: "layer_512_1_sum"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 1.0
  }
  scale_param {
    bias_term: true
  }
}
```

Some code part->

```
# import the necessary packages
```

```
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import
load_model from imutils.video import
VideoStream import numpy as np
import imutils
import time
import cv2
import os


def detect_and_predict_mask(frame, faceNet, maskNet):
    # grab the dimensions of the frame and then construct a blob
    # from it
    (h, w) = frame.shape[:2]
    blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),
                                 (104.0, 177.0, 123.0))

    #     pass the blob through the network and obtain the face
    detections faceNet.setInput(blob)
    detections = faceNet.forward()
    print(detections.shape)

    # initialize our list of faces, their corresponding locations,
    # and the list of predictions from our face mask network
    faces = []
    locs = []
    preds = []

    # loop over the detections
    for i in range(0, detections.shape[2]):
        # extra…

        # import the necessary packages
        from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
        from tensorflow.keras.preprocessing.image import img_to_array
        from tensorflow.keras.models import load_model
        from imutils.video import VideoStream
        import numpy as np
        import imutils
        import time
        import cv2
```

```
import os

def detect_and_predict_mask(frame, faceNet, maskNet):
  # grab the dimensions of the frame and then construct a blob
  # from it
  (h, w) = frame.shape[:2]
  blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),
          (104.0, 177.0, 123.0))

  # pass the blob through the network and obtain the face detections
  faceNet.setInput(blob)
  detections = faceNet.forward()
  print(detections.shape)

  # initialize our list of faces, their corresponding locations,
  # and the list of predictions from our face mask network
  faces = []
  locs = []
  preds = []

  # loop over the detections
  for i in range(0, detections.shape[2]):
          # extract the confidence (i.e., probability) associated with
          # the detection
          confidence = detections[0, 0, i, 2]

          # filter out weak detections by ensuring the confidence is
          # greater than the minimum confidence
          if confidence > 0.5:
                  # compute the (x, y)-coordinates of the bounding box for
                  # the object
                  box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
                  (startX, startY, endX, endY) = box.astype("int")

                  # ensure the bounding boxes fall within the dimensions of
                  # the frame
                  (startX, startY) = (max(0, startX), max(0, startY))
                  (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

                  # extract the face ROI, convert it from BGR to RGB channel
                  # ordering, resize it to 224x224, and preprocess it
                  face = frame[startY:endY, startX:endX]
                  face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
                  face = cv2.resize(face, (224, 224))
                  face = img_to_array(face)
                  face = preprocess_input(face)
```

```
                # add the face and bounding boxes to their respective
                # lists
                faces.append(face)
                locs.append((startX, startY, endX, endY))

    # only make a predictions if at least one face was detected
    if len(faces) > 0:
        # for faster inference we'll make batch predictions on *all*
        # faces at the same time rather than one-by-one predictions
        # in the above `for` loop
        faces = np.array(faces, dtype="float32")
        preds = maskNet.predict(faces, batch_size=32)

    # return a 2-tuple of the face locations and their corresponding
    # locations
    return (locs, preds)

# load our serialized face detector model from disk
prototxtPath = r"face_detector\deploy.prototxt"
weightsPath = r"face_detector\res10_300x300_ssd_iter_140000.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

# load the face mask detector model from disk
maskNet = load_model("mask_detector.model")

# initialize the video stream
print("[INFO] starting video stream...")
vs = VideoStream(src=0).start()

# loop over the frames from the video stream
while True:
    # grab the frame from the threaded video stream and resize it
    # to have a maximum width of 400 pixels
    frame = vs.read()
    frame = imutils.resize(frame, width=400)

    # detect faces in the frame and determine if they are wearing a
    # face mask or not
    (locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

    # loop over the detected face locations and their corresponding
    # locations
    for (box, pred) in zip(locs, preds):
        # unpack the bounding box and predictions
        (startX, startY, endX, endY) = box
        (mask, withoutMask) = pred
```

```
            # determine the class label and color we'll use to draw
            # the bounding box and text
            label = "Mask" if mask > withoutMask else "No Mask"
            color = (0, 255, 0) if label == "Mask" else (0, 0, 255)

            # include the probability in the label
            label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)

            # display the label and bounding box rectangle on the output
            # frame
            cv2.putText(frame, label, (startX, startY - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
            cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)

    # show the output frame
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

    # if the `q` key was pressed, break from the loop
    if key == ord("q"):
            break

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```


Thank you!