Face mask recognition system using MobileNetV2 with optimization function

ATHEER HADI AL-RAMMAHI

- **Import library:**

```
import pandas as pd
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array, load_img
from tensorflow.keras.utils import to_categorical
import tensorflow as tf
from sklearn.preprocessing import LabelBinarizer
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.models import Model
from tensorflow.keras.models import save_model
from tensorflow.keras import regularizers
from tensorflow.keras.layers import AveragePooling2D, Dropout, Flatten, Dense, Input
from tensorflow.keras.optimizers import Adam
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from imutils import paths
import matplotlib.pyplot as plt
import time
import imutils
from imutils.video import VideoStream
import cv2
import numpy as np
import os
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
```

- **Data Loading and Preprocessing**

We used the Mobilenetv2 package from tensorflow to preprocess our image to work with the mobilenetv2 architecture.

```
# This sis the directory of the dataset. Since it is thesame folder with my script, so i  need only the name
 data_paths = list(paths.list_images('dataset'))
# Used to collect the images
 full_data =[]
# used to collect the labels
 labels = []
# get the label of each image from the name of the image.
for image_path in data_paths:
        label = image_path.split(os.path.sep)[-2]
        image = img_to_array(load_img(image_path, target_size=(224, 224)))
```

```
#used a tensorflow package to preprocess the image in a specific format #to enable it work with mobilenetv2 model.
        image = preprocess_input(image)
#collects the image
        full_data.append(image)
# Collects the labels
        labels.append(label)
# converts image to numpy
data =np.array(full_data,dtype='float32')
# converts labels to numpy
labels= np.array(labels)
```

- **Label conversion**

Original Labels	Converted Format
'with_mask' 'without_mask'	[0., 1.] [1., 0.]

```
# Converts the labels of the image to categorical data
lb = LabelBinarizer()
raw_label = labels
labels = lb.fit_transform(labels)
labels = to_categorical(labels)
labels
```

```
array([[0., 1.],
       [0., 1.],
       [0., 1.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]], dtype=float32)
```

```
len(full_data)
```

```
3832
```

- **visualize images**

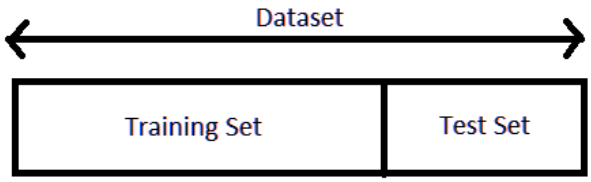
```
# Loads only five image from the directory
masked = []
for image_path in list(paths.list_images(dataset\with_mask'))[:5]:
        image = cv2.imread(image_path)
# Converts to RGB images
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        masked.append(image)
```

```
# Display images with Mathplotlib
fig, axes = plt.subplots(1, 5, figsize=(20, 20))
for img,label,ax in zip(masked[:5],raw_label[2165:2170],axes):
        ax.set_title([label])
        ax.imshow(img)
        ax.axis('off')
plt.show()
```


```
# Loads only five image from the directory
unmasked = []
for image_path in list(paths.list_images(dataset\without_mask'))[:5]:
        image = cv2.imread(image_path)
# Converts to RGB images
        unmasked.append(image)
# Display images with Mathplotlib
fig, axes = plt.subplots(1, 5, figsize=(20, 20))
for img,label,ax in zip(unmasked[:5],raw_label[:5],axes):
        ax.set_title([label])
        ax.imshow(img)
        ax.axis('off')
plt.show()
```

- **Split datasets**

This is done with Sklearn model selection package




```
(x_train, x_test, y_train, y_test) = train_test_split(data, labels, test_size=0.20, stratify=labels,
random_state=250)
```

- **Creating the training model :**

We made used of already existing MobileNetV2 architecture from keras. We remove the add layer and replace it with our own softmax layer brief explanation of the layers.

```
# The pre-made mobilenetv2 model from tensorflow keras
Main_model = MobileNetV2(weights="imagenet", include_top=False
                                    input_tensor=Input(shape=(224, 224, 3)))
Main_out_put = Main_model.output
# We now create our own output layer to use with the main model
head_model = AveragePooling2D(pool_size = (2,2))(Main_out_put)
Flatten_layer = Flatten(name="flatten")(head_model)
Dense_layer = Dense(128, activation="relu")(Flatten_layer)
Dropouts = Dropout(0.5)(Dense_layer)
# Output layer of the head.
Output_layer = Dense(2, activation="softmax")(Dropouts)
# Joining the main model and the created output layer.
model = Model(inputs = Main_model.input, outputs = Output_layer)
#freezing the layers of the main model so that they would not be updated in the first run.
for layer in Main_model.layers
        layer.trainable = False
```

- **Creating The Optimization Function:**

The optimization function contains the training loops and the optimization function. I would be explaining some of the terms here:

```
#initialization function
class Optimizer:
        def __init__(self, model, mb = 8, lr = 0.0001, loss = tf.keras.losses.binary_crossentropy,
                        opt=tf.keras.optimizers.Adam, regularization = "l1",lamda = 0.01):
# model all the way from main class model
self.model = model
self.loss = loss
```

```
# passing the learning rate to the optimization function.
self.optimizer = opt(learning_rate = lr)
#minni batch size
self.mb        = mb
#selects specific regularization type
self.l1_l2_regul = self.regularization_type(regularization)
#regularization parameter
self.reg_const = lamda
#Train and test losses and accuracy
self.train_loss     = tf.keras.metrics.Mean()
self.train_accuracy = tf.keras.metrics.CategoricalAccuracy()
self.test_loss     = tf.keras.metrics.Mean()
self.test_accuracy = tf.keras.metrics.CategoricalAccuracy()
#Training function
@tf.function
def train_step(self, x , y):
        with tf.GradientTape() as tape:
        # Make predictions
        predictions = model(x)
        #Compare the prediction with the ground truth to get the loss.
        loss_temp = self.loss(y, predictions)
        # Apply regularization
        loss = self.apply_reg(loss_temp)
        #Applying gradients
        # Get the trainable weights.
        gradients = tape.gradient(loss, self.model.trainable_variables)
        # apply back propergation.
        self.optimizer.apply_gradients(zip(gradients, self.model.trainable_variables))
        # Update the train loss
        self.train_loss(loss)
        # Get train accuracy
        self.train_accuracy(y, predictions)
        return loss
  @tf.function # This is the test step
def test_step(self, x , y):
        # Make predictions
        predictions = self.model(x)
        #Compare the prediction with the ground truth to get the loss.
        loss = self.loss(y, predictions)
        # Update the test loss
```

```
        self.test_loss(loss)
        # Get updatad test accuracy
        self.test_accuracy(y, predictions)
# This function applies regularization to the weights during training
def apply_reg(self,dyn_loss):
        #gets the weights
        reg_loss =
        tf.compat.v1.get_collection(tf.compat.v1.GraphKeys.REGULARIZATION_LOSSES)
        #using tf.keras.regularizers
        regularizer = self.l1_l2_regul(self.reg_const)
        loss = tf.reduce_mean(dyn_loss + regularizer)
        return loss
#function to recognize and implement the needed regularization according to the imputed argument
def regularization_type(self,querry):
        if querry == "l1":
                return regularizers.l1
        elif querry == "l2":
                return regularizers.l2
# Trains the model by mini batches mb
def train (self):
        batches =0
        for mbX, mbY in self.Augumentation.flow(self.trainX,self.trainY, batch_size = self.mb):
                self.train_step(mbX, mbY)
                batches +=1
                if batches >= len(x_train) / 32:
                # we need to break the loop by hand because
                # the generator loops indefinitely
                        break
  #tests the model also in batches
def test  (self):
        batches = 0
        for mbX, mbY in self.Augumentation.flow(self.testX,self.testY, batch_size =self.mb):
                self.test_step(mbX, mbY)
                batches +=1
                if batches >= len(x_train) / 32:
                # we need to break the loop by hand because
                # the generator loops indefinitely
                    break
```

```
# the run function
def run   (self, dataX, dataY, testX, testY, epochs, verbose=2):
                    # collects the training loss history
          historyTR = []
                # collects the test loss histopry
          historyTS = []
                # collects the training Accuracy history
          historyTR_acc = []
                # collects the training Accuracy history
          historyTS_acc = []
                # for displaying outputs during training
          template = '{} {}, {}: {}, {}: {}, {}: {},{}: {}'
###### Data Augumentation ######
# This is done authomatically during the training process using the keras function ImageDataGenerator.
# At each epoch, it splits the image into batches and generates variation of sample images for each individual image
 # The type of samples to generate is imputed inside the function
 ##########################################
self.Augumentation= ImageDataGenerator(rotation_range=20,zoom_range=0.15,
                                       width_shift_range=0.2,
                                       height_shift_range=0.2,
                                       shear_range=0.15,horizontal_flip=
                                       True,fill_mode="nearest")
self.Augumentation.fit(x_train)
# Collects the imput data and pass them into a global variable of the class
self.trainX = dataX
self.trainY = dataY
self.testX  = testX
self.testY = testY
#training loop
for i in range(epochs):
          self.train () # calls train step
          self.test  () # calls test step immidiately after train step
          #prints train and test loss and accuracy for display
          if verbose > 0:
                    print(template.format("epoch: ", i+1," TRAIN LOSS: ", self.train_loss.result(),
                      " TEST LOSS: " , self.test_loss.result(),
                      " TRAIN ACC: " , self.train_accuracy.result()*100,
                      " TEST ACC: "  , self.test_accuracy.result()*100))
```

```
#gathers training information data for visualization
  temp = '{}'
  historyTR.append(float(temp.format(self.train_loss.result())))
  historyTS.append(float(temp.format(self.test_loss.result() )))
  historyTR_acc.append(float(temp.format(self.train_accuracy.result()*100)))
  historyTS_acc.append(float(temp.format(self.test_accuracy.result()*100 )))
#resets the loss and accuracy history after each epoch
  self.train_loss.reset_states()
  self.train_accuracy.reset_states()
  self.test_loss.reset_states()
  self.test_accuracy.reset_states()
 return historyTR, historyTS,historyTR_acc,historyTS_acc

# Call the optimization function with imput parameters
opt    = Optimizer (model, mb = 20, lr = 0.0001,regularization = "l1",lamda = 0.01 )
# call the run function inside the optimization class with the datasets
los_t,los_v,acc_t,acc_v = opt.run (x_train, y_train, x_test, y_test, 10, verbose=1)
```

```
epoch:  1,  TRAIN LOSS: : 0.14606596529483795,  TEST LOSS: : 0.038260359317064285,  TRAIN ACC: : 95.83332824707031, TEST ACC: :
98.83843994140625
epoch:  2,  TRAIN LOSS: : 0.04056039825081825,  TEST LOSS: : 0.04023775830864906,  TRAIN ACC: : 98.64583587646484, TEST ACC: :
98.83843994140625
epoch:  3,  TRAIN LOSS: : 0.03499419614672661,  TEST LOSS: : 0.04195164889097214,  TRAIN ACC: : 99.01041412353516, TEST ACC: :
98.68004608154297
epoch:  4,  TRAIN LOSS: : 0.034981559962034225,  TEST LOSS: : 0.033784981817007065,  TRAIN ACC: : 98.75, TEST ACC: : 99.2080307
006836
epoch:  5,  TRAIN LOSS: : 0.016051508486270905,  TEST LOSS: : 0.04284724220633507,  TRAIN ACC: : 99.47917175292969, TEST ACC: :
98.73284149169922
epoch:  6,  TRAIN LOSS: : 0.018232503905892372,  TEST LOSS: : 0.03095497190952301,  TRAIN ACC: : 99.375, TEST ACC: : 98.9968261
71875
epoch:  7,  TRAIN LOSS: : 0.013251420110464096,  TEST LOSS: : 0.035314515233039856,  TRAIN ACC: : 99.63541412353516, TEST ACC:
: 98.996826171875
epoch:  8,  TRAIN LOSS: : 0.020469969138503075,  TEST LOSS: : 0.03228059038519859,  TRAIN ACC: : 99.16666412353516, TEST ACC: :
98.8912353515625
epoch:  9,  TRAIN LOSS: : 0.021359754726290703,  TEST LOSS: : 0.06452398002147675,  TRAIN ACC: : 99.16666412353516, TEST ACC: :
98.25765228271484
epoch:  10,  TRAIN LOSS: : 0.019889576360583305,  TEST LOSS: : 0.0381159782409668,  TRAIN ACC: : 99.42708587646484, TEST ACC: :
98.8912353515625
```

```
model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)            [(None, 224, 224, 3  0           []
                                 )]

 Conv1 (Conv2D)                  (None, 112, 112, 32  864         ['input_1
[0][0]']
                                 )

 bn_Conv1 (BatchNormalization)   (None, 112, 112, 32  128         ['Conv1[0
][0]']
                                 )
```

```
 Conv1_relu (ReLU)              (None, 112, 112, 32  0           ['bn_Conv
1[0][0]']
                                )

 expanded_conv_depthwise (Depth  (None, 112, 112, 32  288        ['Conv1_r
elu[0][0]']
 wiseConv2D)                    )

 expanded_conv_depthwise_BN (Ba  (None, 112, 112, 32  128        ['expande
d_conv_depthwise[0][0]']
 tchNormalization)              )

 expanded_conv_depthwise_relu (  (None, 112, 112, 32  0          ['expande
d_conv_depthwise_BN[0][0
 ReLU)                          )                                ]']

 expanded_conv_project (Conv2D)  (None, 112, 112, 16  512        ['expande
d_conv_depthwise_relu[0]
                                )                                [0]']

 expanded_conv_project_BN (Batc  (None, 112, 112, 16  64         ['expande
d_conv_project[0][0]']
 hNormalization)                )
block_1_expand (Conv2D)        (None, 112, 112, 96  1536        ['expanded_co
nv_project_BN[0][0]'
                               )                                 ]

 block_1_expand_BN (BatchNormal  (None, 112, 112, 96  384        ['block_1_ex
pand[0][0]']
 ization)                       )

 block_1_expand_relu (ReLU)     (None, 112, 112, 96  0           ['block_1_ex
pand_BN[0][0]']
                                )

 block_1_pad (ZeroPadding2D)    (None, 113, 113, 96  0           ['block_1_ex
pand_relu[0][0]']
                                )

 block_1_depthwise (DepthwiseCo  (None, 56, 56, 96)  864         ['block_1_pa
d[0][0]']
 nv2D)

 block_1_depthwise_BN (BatchNor  (None, 56, 56, 96)  384         ['block_1_de
pthwise[0][0]']
 malization)
```

```
 block_1_depthwise_relu (ReLU)  (None, 56, 56, 96)   0           ['block_1_de
pthwise_BN[0][0]']

 block_1_project (Conv2D)       (None, 56, 56, 24)   2304        ['block_1_de
pthwise_relu[0][0]']

 block_1_project_BN (BatchNorma  (None, 56, 56, 24)  96          ['block_1_pr
oject[0][0]']
 lization)

 block_2_expand (Conv2D)        (None, 56, 56, 144)  3456        ['block_1_pr
oject_BN[0][0]']

 block_2_expand_BN (BatchNormal  (None, 56, 56, 144)  576        ['block_2_ex
pand[0][0]']
 ization)

 block_2_expand_relu (ReLU)     (None, 56, 56, 144)  0           ['block_2_ex
pand_BN[0][0]']

 block_2_depthwise (DepthwiseCo  (None, 56, 56, 144)  1296       ['block_2_ex
pand_relu[0][0]']
 nv2D)

 block_2_depthwise_BN (BatchNor  (None, 56, 56, 144)  576        ['block_2_de
pthwise[0][0]']
 malization)

 block_2_depthwise_relu (ReLU)  (None, 56, 56, 144)  0           ['block_2_de
pthwise_BN[0][0]']

 block_2_project (Conv2D)       (None, 56, 56, 24)   3456        ['block_2_de
pthwise_relu[0][0]']

 block_2_project_BN (BatchNorma  (None, 56, 56, 24)  96          ['block_2_pr
oject[0][0]']
 lization)

 block_2_add (Add)              (None, 56, 56, 24)   0           ['block_1_pr
oject_BN[0][0]',
                                                                 'block_2_pr
oject_BN[0][0]']
```

```
 block_3_expand (Conv2D)        (None, 56, 56, 144)  3456        ['block_2_ad
d[0][0]']

 block_3_expand_BN (BatchNormal  (None, 56, 56, 144)  576        ['block_3_ex
pand[0][0]']
 ization)

 block_3_expand_relu (ReLU)     (None, 56, 56, 144)  0           ['block_3_ex
pand_BN[0][0]']

 block_3_pad (ZeroPadding2D)    (None, 57, 57, 144)  0           ['block_3_ex
pand_relu[0][0]']

 block_3_depthwise (DepthwiseCo  (None, 28, 28, 144)  1296       ['block_3_pa
d[0][0]']
 nv2D)

 block_3_depthwise_BN (BatchNor  (None, 28, 28, 144)  576        ['block_3_de
pthwise[0][0]']
 malization)
block_3_depthwise_relu (ReLU)  (None, 28, 28, 144)  0           ['block_3_dep
thwise_BN[0][0]']

 block_3_project (Conv2D)       (None, 28, 28, 32)   4608        ['block_3_de
pthwise_relu[0][0]']

 block_3_project_BN (BatchNorma  (None, 28, 28, 32)  128         ['block_3_pr
oject[0][0]']
 lization)

 block_4_expand (Conv2D)        (None, 28, 28, 192)  6144        ['block_3_pr
oject_BN[0][0]']

 block_4_expand_BN (BatchNormal  (None, 28, 28, 192)  768        ['block_4_ex
pand[0][0]']
 ization)

 block_4_expand_relu (ReLU)     (None, 28, 28, 192)  0           ['block_4_ex
pand_BN[0][0]']

 block_4_depthwise (DepthwiseCo  (None, 28, 28, 192)  1728       ['block_4_ex
pand_relu[0][0]']
 nv2D)
```

```
 block_4_depthwise_BN (BatchNor  (None, 28, 28, 192)  768         ['block_4_de
pthwise[0][0]']
 malization)

 block_4_depthwise_relu (ReLU)  (None, 28, 28, 192)  0           ['block_4_de
pthwise_BN[0][0]']

 block_4_project (Conv2D)       (None, 28, 28, 32)   6144        ['block_4_de
pthwise_relu[0][0]']

 block_4_project_BN (BatchNorma  (None, 28, 28, 32)  128         ['block_4_pr
oject[0][0]']
 lization)

 block_4_add (Add)              (None, 28, 28, 32)   0           ['block_3_pr
oject_BN[0][0]',
                                                                  'block_4_pr
oject_BN[0][0]']

 block_5_expand (Conv2D)        (None, 28, 28, 192)  6144        ['block_4_ad
d[0][0]']

 block_5_expand_BN (BatchNormal  (None, 28, 28, 192)  768        ['block_5_ex
pand[0][0]']
 ization)

 block_5_expand_relu (ReLU)     (None, 28, 28, 192)  0           ['block_5_ex
pand_BN[0][0]']

 block_5_depthwise (DepthwiseCo  (None, 28, 28, 192)  1728       ['block_5_ex
pand_relu[0][0]']
 nv2D)

 block_5_depthwise_BN (BatchNor  (None, 28, 28, 192)  768        ['block_5_de
pthwise[0][0]']
 malization)

 block_5_depthwise_relu (ReLU)  (None, 28, 28, 192)  0           ['block_5_de
pthwise_BN[0][0]']

 block_5_project (Conv2D)       (None, 28, 28, 32)   6144        ['block_5_de
pthwise_relu[0][0]']
```

```
 block_5_project_BN (BatchNorma  (None, 28, 28, 32)  128         ['block_5_pr
oject[0][0]']
 lization)

 block_5_add (Add)              (None, 28, 28, 32)   0           ['block_4_ad
d[0][0]',
                                                                  'block_5_pr
oject_BN[0][0]']

 block_6_expand (Conv2D)        (None, 28, 28, 192)  6144        ['block_5_ad
d[0][0]']

 block_6_expand_BN (BatchNormal  (None, 28, 28, 192)  768        ['block_6_ex
pand[0][0]']
 ization)

 block_6_expand_relu (ReLU)     (None, 28, 28, 192)  0           ['block_6_ex
pand_BN[0][0]']

 block_6_pad (ZeroPadding2D)    (None, 29, 29, 192)  0           ['block_6_ex
pand_relu[0][0]']

 block_6_depthwise (DepthwiseCo  (None, 14, 14, 192)  1728       ['block_6_pa
d[0][0]']
 nv2D)

 block_6_depthwise_BN (BatchNor  (None, 14, 14, 192)  768        ['block_6_de
pthwise[0][0]']
 malization)

 block_6_depthwise_relu (ReLU)  (None, 14, 14, 192)  0           ['block_6_de
pthwise_BN[0][0]']

 block_6_project (Conv2D)       (None, 14, 14, 64)   12288       ['block_6_de
pthwise_relu[0][0]']

 block_6_project_BN (BatchNorma  (None, 14, 14, 64)  256         ['block_6_pr
oject[0][0]']
 lization)

 block_7_expand (Conv2D)        (None, 14, 14, 384)  24576       ['block_6_pr
oject_BN[0][0]']
```

```
 block_7_expand_BN (BatchNormal  (None, 14, 14, 384)  1536       ['block_7_ex
pand[0][0]']
 ization)

 block_7_expand_relu (ReLU)     (None, 14, 14, 384)  0          ['block_7_ex
pand_BN[0][0]']

 block_7_depthwise (DepthwiseCo  (None, 14, 14, 384)  3456       ['block_7_ex
pand_relu[0][0]']
 nv2D)

 block_7_depthwise_BN (BatchNor  (None, 14, 14, 384)  1536       ['block_7_de
pthwise[0][0]']
 malization)

 block_7_depthwise_relu (ReLU)  (None, 14, 14, 384)  0          ['block_7_de
pthwise_BN[0][0]']

 block_7_project (Conv2D)       (None, 14, 14, 64)   24576      ['block_7_de
pthwise_relu[0][0]']

 block_7_project_BN (BatchNorma  (None, 14, 14, 64)  256        ['block_7_pr
oject[0][0]']
 lization)

 block_7_add (Add)              (None, 14, 14, 64)   0          ['block_6_pr
oject_BN[0][0]',
                                                                 'block_7_pr
oject_BN[0][0]']

 block_8_expand (Conv2D)        (None, 14, 14, 384)  24576      ['block_7_ad
d[0][0]']

 block_8_expand_BN (BatchNormal  (None, 14, 14, 384)  1536       ['block_8_ex
pand[0][0]']
 ization)

 block_8_expand_relu (ReLU)     (None, 14, 14, 384)  0          ['block_8_ex
pand_BN[0][0]']

 block_8_depthwise (DepthwiseCo  (None, 14, 14, 384)  3456       ['block_8_ex
pand_relu[0][0]']
 nv2D)
```

```
 block_8_depthwise_BN (BatchNor  (None, 14, 14, 384)  1536       ['block_8_de
pthwise[0][0]']
 malization)

 block_8_depthwise_relu (ReLU)  (None, 14, 14, 384)  0          ['block_8_de
pthwise_BN[0][0]']

 block_8_project (Conv2D)       (None, 14, 14, 64)   24576      ['block_8_de
pthwise_relu[0][0]']

 block_8_project_BN (BatchNorma  (None, 14, 14, 64)  256        ['block_8_pr
oject[0][0]']
 lization)

 block_8_add (Add)              (None, 14, 14, 64)   0          ['block_7_ad
d[0][0]',
                                                                 'block_8_pr
oject_BN[0][0]']

 block_9_expand (Conv2D)        (None, 14, 14, 384)  24576      ['block_8_ad
d[0][0]']

 block_9_expand_BN (BatchNormal  (None, 14, 14, 384)  1536       ['block_9_ex
pand[0][0]']
 ization)

 block_9_expand_relu (ReLU)     (None, 14, 14, 384)  0          ['block_9_ex
pand_BN[0][0]']

 block_9_depthwise (DepthwiseCo  (None, 14, 14, 384)  3456       ['block_9_ex
pand_relu[0][0]']
 nv2D)

 block_9_depthwise_BN (BatchNor  (None, 14, 14, 384)  1536       ['block_9_de
pthwise[0][0]']
 malization)

 block_9_depthwise_relu (ReLU)  (None, 14, 14, 384)  0          ['block_9_de
pthwise_BN[0][0]']

 block_9_project (Conv2D)       (None, 14, 14, 64)   24576      ['block_9_de
pthwise_relu[0][0]']
```

```
 block_9_project_BN (BatchNorma  (None, 14, 14, 64)  256         ['block_9_pr
oject[0][0]']
 lization)

 block_9_add (Add)              (None, 14, 14, 64)   0           ['block_8_ad
d[0][0]',
                                                                  'block_9_pr
oject_BN[0][0]']

 block_10_expand (Conv2D)       (None, 14, 14, 384)  24576       ['block_9_ad
d[0][0]']

 block_10_expand_BN (BatchNorma  (None, 14, 14, 384)  1536       ['block_10_e
xpand[0][0]']
 lization)

 block_10_expand_relu (ReLU)    (None, 14, 14, 384)  0           ['block_10_e
xpand_BN[0][0]']

 block_10_depthwise (DepthwiseC  (None, 14, 14, 384)  3456       ['block_10_e
xpand_relu[0][0]']
 onv2D)

 block_10_depthwise_BN (BatchNo  (None, 14, 14, 384)  1536       ['block_10_d
epthwise[0][0]']
 rmalization)

 block_10_depthwise_relu (ReLU)  (None, 14, 14, 384)  0          ['block_10_d
epthwise_BN[0][0]']

 block_10_project (Conv2D)      (None, 14, 14, 96)   36864       ['block_10_d
epthwise_relu[0][0]']

 block_10_project_BN (BatchNorm  (None, 14, 14, 96)  384         ['block_10_p
roject[0][0]']
 alization)

 block_11_expand (Conv2D)       (None, 14, 14, 576)  55296       ['block_10_p
roject_BN[0][0]']

 block_11_expand_BN (BatchNorma  (None, 14, 14, 576)  2304       ['block_11_e
xpand[0][0]']
 lization)
```

```
 block_11_expand_relu (ReLU)    (None, 14, 14, 576)  0           ['block_11_e
xpand_BN[0][0]']

 block_11_depthwise (DepthwiseC  (None, 14, 14, 576)  5184       ['block_11_e
xpand_relu[0][0]']
 onv2D)

 block_11_depthwise_BN (BatchNo  (None, 14, 14, 576)  2304       ['block_11_d
epthwise[0][0]']
 rmalization)

 block_11_depthwise_relu (ReLU)  (None, 14, 14, 576)  0          ['block_11_d
epthwise_BN[0][0]']

 block_11_project (Conv2D)      (None, 14, 14, 96)   55296       ['block_11_d
epthwise_relu[0][0]']

 block_11_project_BN (BatchNorm  (None, 14, 14, 96)  384         ['block_11_p
roject[0][0]']
 alization)

 block_11_add (Add)             (None, 14, 14, 96)   0           ['block_10_p
roject_BN[0][0]',
                                                                  'block_11_p
roject_BN[0][0]']

 block_12_expand (Conv2D)       (None, 14, 14, 576)  55296       ['block_11_a
dd[0][0]']

 block_12_expand_BN (BatchNorma  (None, 14, 14, 576)  2304       ['block_12_e
xpand[0][0]']
 lization)

 block_12_expand_relu (ReLU)    (None, 14, 14, 576)  0           ['block_12_e
xpand_BN[0][0]']

 block_12_depthwise (DepthwiseC  (None, 14, 14, 576)  5184       ['block_12_e
xpand_relu[0][0]']
 onv2D)

 block_12_depthwise_BN (BatchNo  (None, 14, 14, 576)  2304       ['block_12_d
epthwise[0][0]']
 rmalization)
```

```
 block_12_depthwise_relu (ReLU)  (None, 14, 14, 576)  0          ['block_12_d
epthwise_BN[0][0]']

 block_12_project (Conv2D)      (None, 14, 14, 96)   55296       ['block_12_d
epthwise_relu[0][0]']

 block_12_project_BN (BatchNorm  (None, 14, 14, 96)  384         ['block_12_p
roject[0][0]']
 alization)

 block_12_add (Add)             (None, 14, 14, 96)   0           ['block_11_a
dd[0][0]',
                                                                  'block_12_p
roject_BN[0][0]']

 block_13_expand (Conv2D)       (None, 14, 14, 576)  55296       ['block_12_a
dd[0][0]']

 block_13_expand_BN (BatchNorma  (None, 14, 14, 576)  2304       ['block_13_e
xpand[0][0]']
 lization)

 block_13_expand_relu (ReLU)    (None, 14, 14, 576)  0           ['block_13_e
xpand_BN[0][0]']

 block_13_pad (ZeroPadding2D)   (None, 15, 15, 576)  0           ['block_13_e
xpand_relu[0][0]']

 block_13_depthwise (DepthwiseC  (None, 7, 7, 576)   5184        ['block_13_p
ad[0][0]']
 onv2D)

 block_13_depthwise_BN (BatchNo  (None, 7, 7, 576)   2304        ['block_13_d
epthwise[0][0]']
 rmalization)

 block_13_depthwise_relu (ReLU)  (None, 7, 7, 576)   0          ['block_13_d
epthwise_BN[0][0]']

 block_13_project (Conv2D)      (None, 7, 7, 160)    92160       ['block_13_d
epthwise_relu[0][0]']

 block_13_project_BN (BatchNorm  (None, 7, 7, 160)   640         ['block_13_p
roject[0][0]']
```

```
 alization)

 block_14_expand (Conv2D)       (None, 7, 7, 960)    153600      ['block_13_p
roject_BN[0][0]']

 block_14_expand_BN (BatchNorma  (None, 7, 7, 960)   3840        ['block_14_e
xpand[0][0]']
 lization)

 block_14_expand_relu (ReLU)    (None, 7, 7, 960)    0           ['block_14_e
xpand_BN[0][0]']

 block_14_depthwise (DepthwiseC  (None, 7, 7, 960)   8640        ['block_14_e
xpand_relu[0][0]']
 onv2D)

 block_14_depthwise_BN (BatchNo  (None, 7, 7, 960)   3840        ['block_14_d
epthwise[0][0]']
 rmalization)

 block_14_depthwise_relu (ReLU)  (None, 7, 7, 960)   0           ['block_14_d
epthwise_BN[0][0]']

 block_14_project (Conv2D)      (None, 7, 7, 160)    153600      ['block_14_d
epthwise_relu[0][0]']

 block_14_project_BN (BatchNorm  (None, 7, 7, 160)   640         ['block_14_p
roject[0][0]']
 alization)

 block_14_add (Add)             (None, 7, 7, 160)    0           ['block_13_p
roject_BN[0][0]',
                                                                  'block_14_p
roject_BN[0][0]']

 block_15_expand (Conv2D)       (None, 7, 7, 960)    153600      ['block_14_a
dd[0][0]']

 block_15_expand_BN (BatchNorma  (None, 7, 7, 960)   3840        ['block_15_e
xpand[0][0]']
 lization)

 block_15_expand_relu (ReLU)    (None, 7, 7, 960)    0           ['block_15_e
xpand_BN[0][0]']
```

```
 block_15_depthwise (DepthwiseC  (None, 7, 7, 960)   8640        ['block_15_e
xpand_relu[0][0]']
 onv2D)

 block_15_depthwise_BN (BatchNo  (None, 7, 7, 960)   3840        ['block_15_d
epthwise[0][0]']
 rmalization)

 block_15_depthwise_relu (ReLU)  (None, 7, 7, 960)   0           ['block_15_d
epthwise_BN[0][0]']

 block_15_project (Conv2D)      (None, 7, 7, 160)    153600      ['block_15_d
epthwise_relu[0][0]']

 block_15_project_BN (BatchNorm  (None, 7, 7, 160)   640         ['block_15_p
roject[0][0]']
 alization)

 block_15_add (Add)             (None, 7, 7, 160)    0           ['block_14_a
dd[0][0]',
                                                                  'block_15_p
roject_BN[0][0]']

 block_16_expand (Conv2D)       (None, 7, 7, 960)    153600      ['block_15_a
dd[0][0]']

 block_16_expand_BN (BatchNorma  (None, 7, 7, 960)   3840        ['block_16_e
xpand[0][0]']
 lization)

 block_16_expand_relu (ReLU)    (None, 7, 7, 960)    0           ['block_16_e
xpand_BN[0][0]']

 block_16_depthwise (DepthwiseC  (None, 7, 7, 960)   8640        ['block_16_e
xpand_relu[0][0]']
 onv2D)

 block_16_depthwise_BN (BatchNo  (None, 7, 7, 960)   3840        ['block_16_d
epthwise[0][0]']
 rmalization)

 block_16_depthwise_relu (ReLU)  (None, 7, 7, 960)   0           ['block_16_d
epthwise_BN[0][0]']
```

```
 block_16_project (Conv2D)      (None, 7, 7, 320)    307200      ['block_16_d
epthwise_relu[0][0]']

 block_16_project_BN (BatchNorm  (None, 7, 7, 320)   1280        ['block_16_p
roject[0][0]']
 alization)

 Conv_1 (Conv2D)                (None, 7, 7, 1280)   409600      ['block_16_p
roject_BN[0][0]']

 Conv_1_bn (BatchNormalization)  (None, 7, 7, 1280)  5120        ['Conv_1[0][
0]']

 out_relu (ReLU)                (None, 7, 7, 1280)   0           ['Conv_1_bn[
0][0]']

 average_pooling2d (AveragePool  (None, 3, 3, 1280)  0           ['out_relu[0
][0]']
 ing2D)

 flatten (Flatten)              (None, 11520)        0           ['average_po
oling2d[0][0]']

 dense (Dense)                  (None, 128)          1474688     ['flatten[0]
[0]']

 dropout (Dropout)              (None, 128)          0           ['dense[0][0
]']

 dense_1 (Dense)                (None, 2)            258         ['dropout[0]
[0]']

==================================================================================
Total params: 3,732,930
Trainable params: 1,474,946
Non-trainable params: 2,257,984
```

 # Display the details of the training process

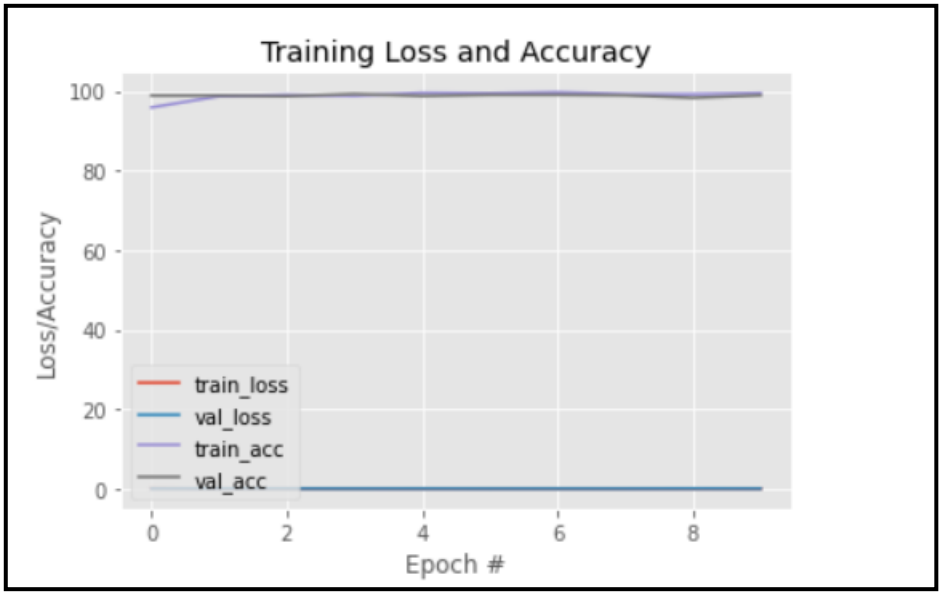
N = 10

plt.style.use("ggplot")

plt.figure()

plt.plot(np.arange(0, N), los_t, label="train_loss")

```
plt.plot(np.arange(0, N), los_v, label="val_loss")
plt.plot(np.arange(0, N), acc_t, label="train_acc")
plt.plot(np.arange(0, N), acc_v, label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
#.savefig(args["plot"])
```



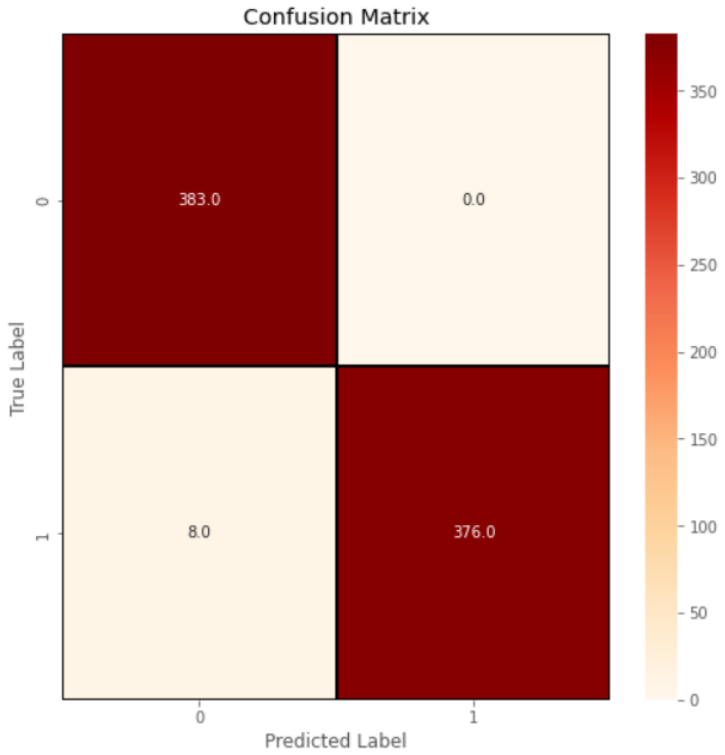

```
# Model accuracy
correct = 0
total = 0
pred = np.argmax(model.predict(x_test), axis=1)
for i, img in enumerate(pred):
    if img == np.argmax(y_test[i]):
        correct += 1
    total += 1
print(correct/total * 100)
```

```
98.95697522816167
```

```
# Classification report
cr = classification_report(np.argmax(model.predict(x_test), axis=1), np.argmax(y_test, axis = 1))
print(cr)
```

	precision	recall	f1-score	support
0	1.00	0.98	0.99	391
1	0.98	1.00	0.99	376
accuracy			0.99	767
macro avg	0.99	0.99	0.99	767
weighted avg	0.99	0.99	0.99	767

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
Y_pred = model.predict(x_test)
Y_pred_classes = np.argmax(Y_pred,axis = 1)
Y_true = np.argmax(y_test,axis = 1)
confusion_mtx = confusion_matrix(Y_true, Y_pred_classes)
f,ax = plt.subplots(figsize=(8, 8))
sns.heatmap(confusion_mtx, annot=True, linewidths=0.01,cmap="OrRd",linecolor="black",
fmt= '.1f',ax=ax)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()
```



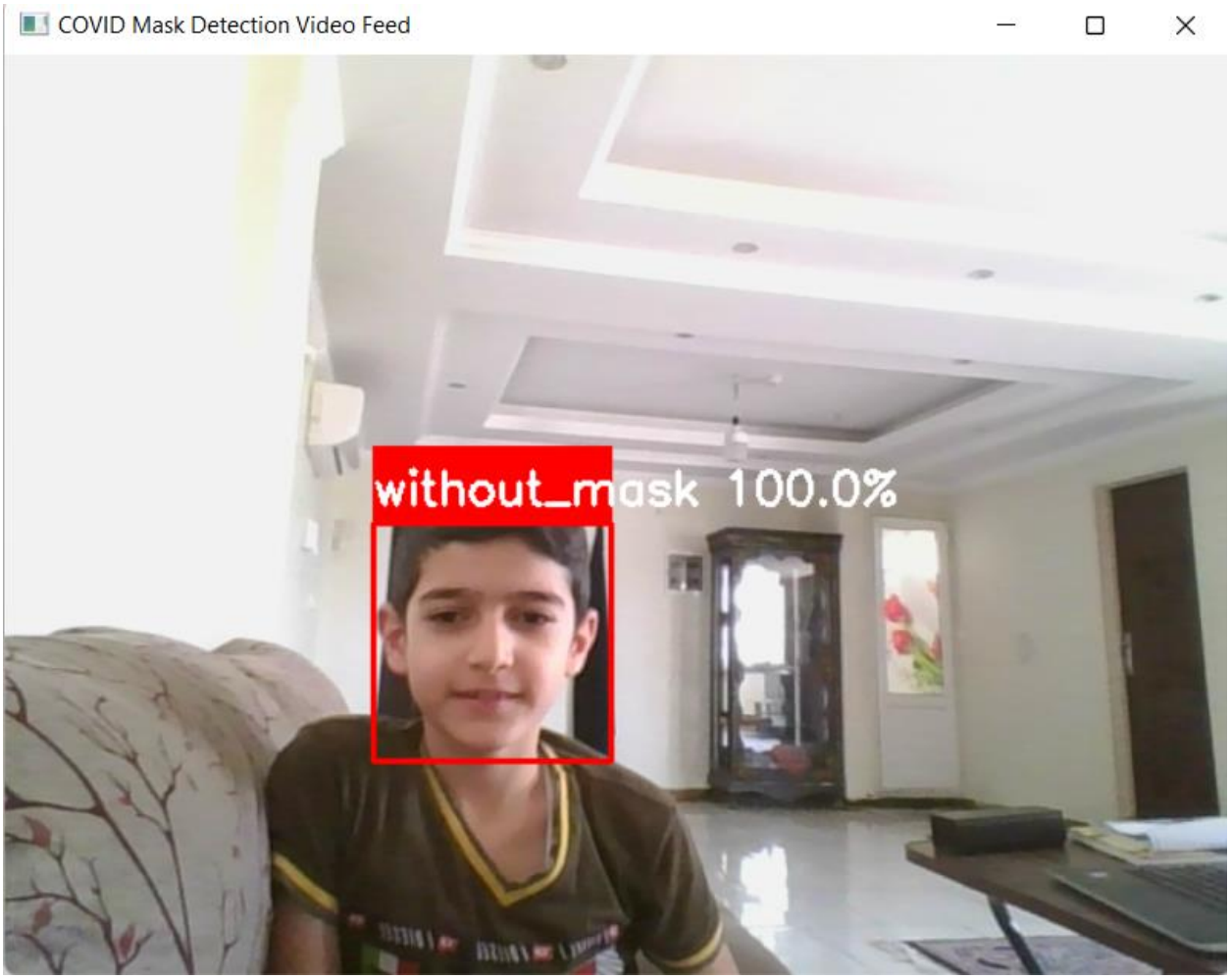

```
# Saving the model
model.save('my_model.h5')
```

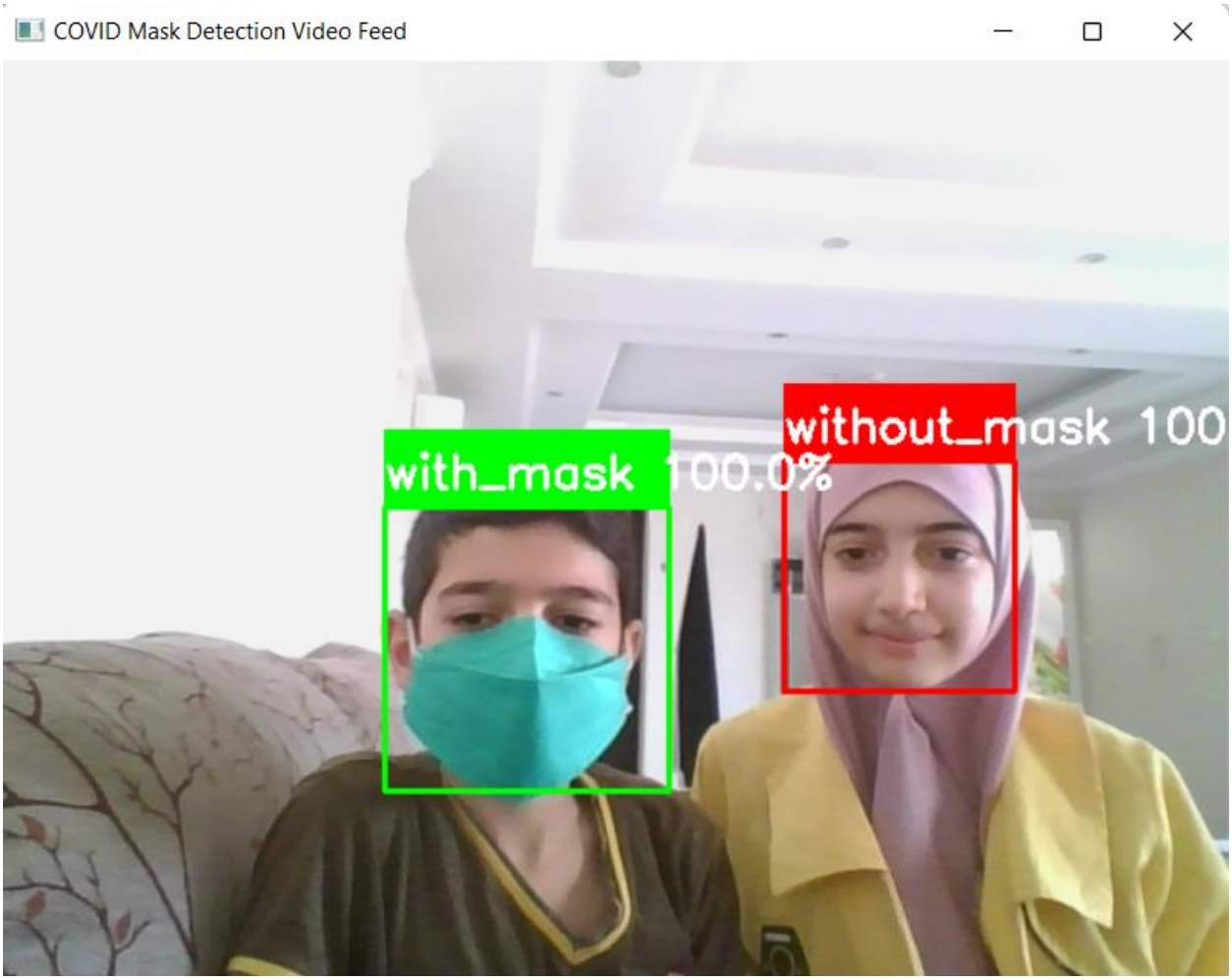
- **Test with real time video**

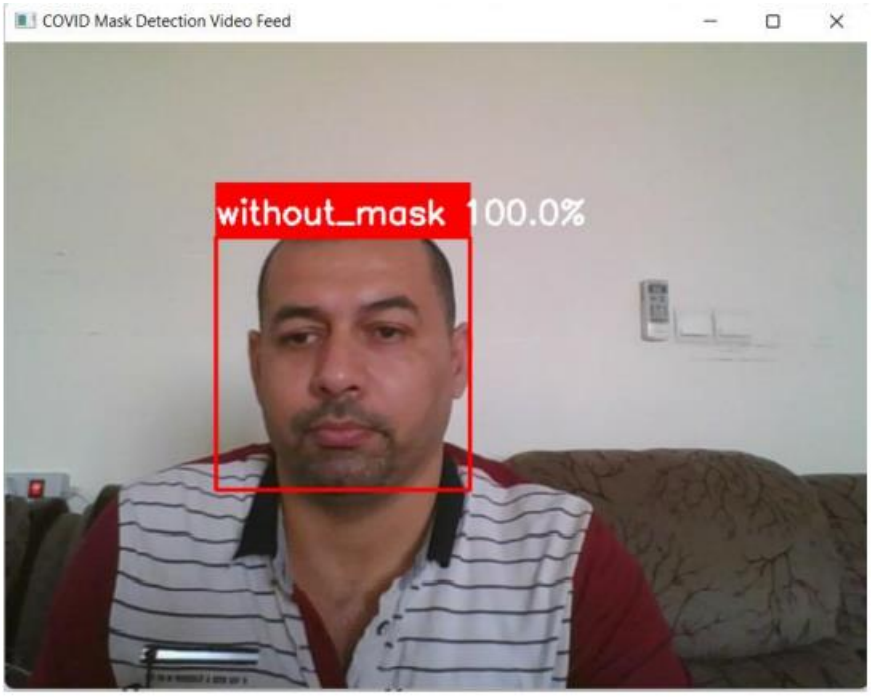
First we load an helper script that helps to detect face in an image before using our model to detect if there is a mask on the face or not

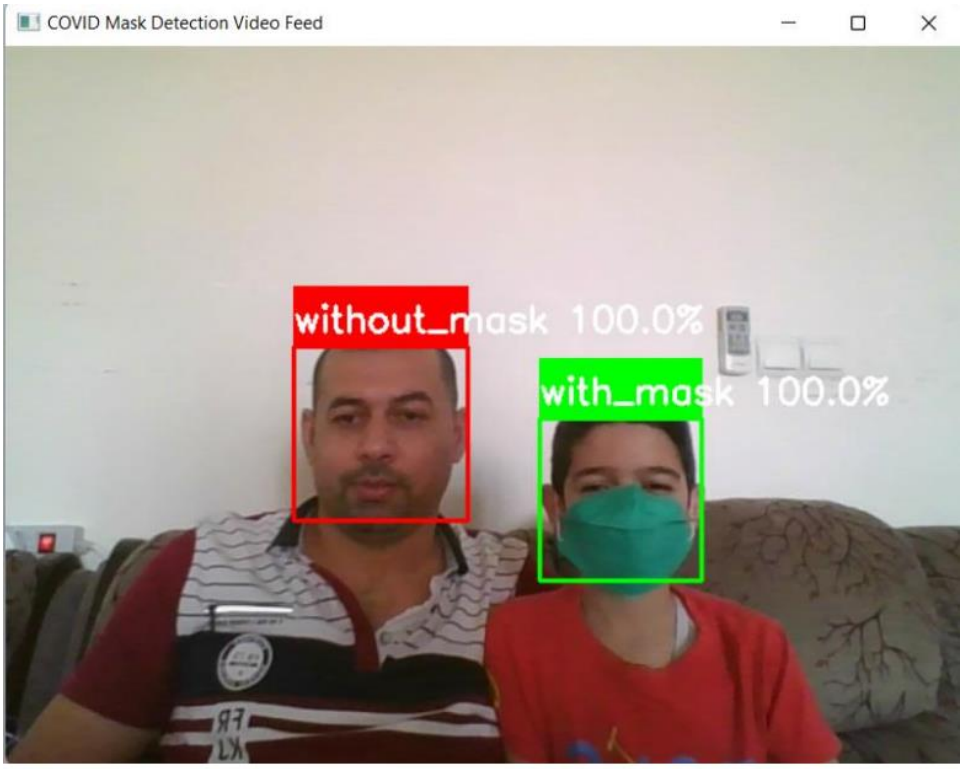
```
load_model = keras.models.load_model('my_model.h5')
face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
labels_dict={0:'with_mask',1:'without_mask'}
color_dict={0:(0,255,0),1:(0,0,255)}
size = 4
cv2.namedWindow("COVID Mask Detection Video Feed")
webcam = cv2.VideoCapture(0)
classifier = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
while True:
    rval, im = webcam.read()
    im=cv2.flip(im,1,1)
    mini = cv2.resize(im, (im.shape[1] // size, im.shape[0] // size))
    faces = classifier.detectMultiScale(mini)
    for f in faces:
        (x, y, w, h) = [v * size for v in f]
        face_img = im[y:y+h, x:x+w]
        resized=cv2.resize(face_img,(224,224))
        normalized=resized/255.0
        reshaped=np.reshape(normalized,(1,224,224,3))
        reshaped = np.vstack([reshaped])
        result=load_model.predict(reshaped)
        print(result)
        if result[0][0] > result[0][1]:
            percent = round(result[0][0]*100,2)
        else:
            percent = round(result[0][1]*100,2)

        label=np.argmax(result,axis=1)[0]

        cv2.rectangle(im,(x,y),(x+w,y+h),color_dict[label],2)
        cv2.rectangle(im,(x,y-40),(x+w,y),color_dict[label],-1)
        cv2.putText(im, labels_dict[label] + " " + str(percent) + "%", (x, y-
                        10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)

    if im is not None:
        cv2.imshow('COVID Mask Detection Video Feed', im)
    key = cv2.waitKey(10)
```

```
    # Exit
    if key == 27: #The Esc key
        break
# Stop video
webcam.release()
# Close all windows
cv2.destroyAllWindows()
```




COVID Mask Detection Video Feed
with_mask 100.0%
without_mask 100


COVID Mask Detection Video Feed
without_mask 100.0%


COVID Mask Detection Video Feed
without_mask 100.0%
with_mask 100.0%


without_mask 100.0%
without_mask 100.0