School of Information & Software Engineering


# Experiment Report

Student ID 2019090146011, 2019090146008, 2019090146010

Student Name ESSET WISDOM ESSET, BRENDA CHELAGAT ROTICH, EKO MFONISO MICHAEL

Experiment Name Real-Time Face Mask Detector with Python, OpenCV, Keras

Experiment Time 12/24/2021

# Introduction

### Project Objective

During the COVID19 pandemic, WHO mandated that a mask be worn to protect against this deadly virus. In this tutorial, we are developing a real-time face mask detector using Python, a machine learning project. We will build a system that judges whether a mask is worn in real time. Train a face mask detector model using Keras and OpenCV.

# Steps involved in making the program

### Getting the dataset

First we need a dataset to train our python program. We got one from the internet containing 1376 images with 690 images containing images of people wearing mask and 686 images with people without mask. You would be able to see them in the folders test and train.


### Writing the code

We will make this project in two parts. In the first part, we

write a Python script to train a face mask detector model using Keras. The second part uses OpenCV to test the results on a live webcam. Create a Python file train.py to write code to train a neural network on the data set. Follow these steps:

1. Imports:

Import all the libraries and modules required.

```
from tensorflow.keras.optimizers import RMSprop
from keras.preprocessing.image import ImageDataGenerator
import cv2
from keras.models import Sequential
from keras.layers import Conv2D, Input, ZeroPadding2D, BatchNormalization, Activation
from keras.models import Model, load_model
from keras.callbacks import TensorBoard, ModelCheckpoint
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.utils import shuffle
import imutils
import numpy as np
import os
```

2. Building a Neural Network:

This convolutional network consists of two pairs of Conv and MaxPool layers for retrieving features from the data set. This is followed by Flatten and Dropout layers to transform the data into 1D and provide retraining.

Then two dense layers for classification.

```
model =Sequential([
    Conv2D(100, (3,3), activation='relu', input_shape=(150, 150, 3)),
    MaxPooling2D(2,2),

    Conv2D(100, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Flatten(),
    Dropout(0.5),
    Dense(50, activation='relu'),
    Dense(2, activation='softmax')
])
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])
```

3. Image Data Generation/Augmentation:

```
TRAINING_DIR = "./train"
train_datagen = ImageDataGenerator(rescale=1.0/255,
                                   rotation_range=40,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True,
                                   fill_mode='nearest')

train_generator = train_datagen.flow_from_directory(TRAINING_DIR,
                                                    batch_size=10,
                                                    target_size=(150, 150))
```

4. Initialize a callback checkpoint to keep saving best model after each epoch while training:

```
checkpoint = ModelCheckpoint('model2-{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True,mode='auto')
```

5. Train the model:

```
history = model.fit(train_generator,
                    epochs=10,
                    validation_data=validation_generator,
                    callbacks=[checkpoint])
```

Now we will test the results of face mask detector model using OpenCV.

We made a python file “test.py” and typed the code below.

```python
import cv2
import numpy as np
from keras.models import load_model

model = load_model("./model2-001.model/model-010.h5")

results = {0: 'without mask', 1: 'mask'}
GR_dict = {0: (0, 0, 255), 1: (0, 255, 0)}

rect_size = 4
cap = cv2.VideoCapture(0)

haarcascade = cv2.CascadeClassifier(r'C:\Users\Ponder Piece\AppData\Roaming\Python\Python39\site-packages\cv2\data\haarcascade_frontalface_default.xml' )

while True:
    (rval, im) = cap.read()
    im = cv2.flip(im, 1, 1)

    rerect_size = cv2.resize(im, (im.shape[1] // rect_size, im.shape[0] // rect_size))
    faces = haarcascade.detectMultiScale(rerect_size)
    for f in faces:
        (x, y, w, h) = [v * rect_size for v in f]

        face_img = im[y:y + h, x:x + w]
        rerect_sized = cv2.resize(face_img, (150, 150))
```

```python
    for f in faces:
        (x, y, w, h) = [v * size for v in f] #Scale the shapesize backup
        #Save just the rectangle faces in SubRecFaces
        face_img = im[y:y+h, x:x+w]
        resized=cv2.resize(face_img,(150,150))
        normalized=resized/255.0
        reshaped=np.reshape(normalized,(1,150,150,3))
        reshaped = np.vstack([reshaped])
        result=model.predict(reshaped)
        #print(result)

        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], (x, y-10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)

    # Show the image
    cv2.imshow('LIVE',   im)
    key = cv2.waitKey(10)
    # if Esc key is press then break out of the loop
    if key == 27: #The Esc key
        break
# Stop video
webcam.release()

# Close all started windows
cv2.destroyAllWindows()
```

Please note: We used the path to haarcascade_frontalface_default.xml using our system at line 13 above. For the software to run properly on another system, change

the path to where haarcascade_frontalface_default.xml is on that particular system.

Then run the project and observe the model performance.

# Summary

In this project, we developed a deep learning model for face mask detection using Python, Keras and OpenCV. We designed a face mask detector model to detect whether a person is wearing a mask. I trained the model using Keras with network architecture. Training the model is the first part of this project and testing it with a webcam using OpenCV is the second part.

A good project for beginners to apply their knowledge and experience.