

Project Name: Facial Recognition App

Student 1 Name: Andrew Nechifor

Student 1 ID: 16415432

Student 2 Name: Mahjabeen Soomro

Student 2 ID: 17362496

Project Supervisor: Suzanne Little

Date finished working on the document: 06/05/2021

Table of Contents

| | |
|--|----------|
| 1. General Overview | 1 |
| 2. Organization Of The Manual | 1 |
| 3. Contingencies | 2 |
| 4. Installing The Application | 2 |
| 5. Navigating The Application | 3 |
| 6. Training The Facial Recognition Model Yourself | 7 |

General Overview

The goal of the Facial Recognition Application is an application that involves the user opening their camera and being assisted in the act of taking a picture or uploading a picture in order to have a facial recognition model applied to the image. Once the model has analyzed the picture, the results are returned and displayed to the user. The application will inform the user of whether or not they are wearing a mask and if they are wearing the mask correctly or incorrectly.

The aim of this project is to provide users with the ability to detect if someone is wearing a face mask and if that person is wearing the mask correctly or incorrectly. This can help provide businesses or groups of people perform these safety checks autonomously. The information available in this guide will give you instructions on how to install and use this application to perform its goal.

Organization Of The Manual

The user manual consists of 5 sections: General Overview, Contingencies, Installing The Application and Navigating The Application.

- The General Overview section explains in general, what the Brand Analyzer App will do for you and what the goal of the application is.
- The Contingencies section explains what could potentially stop the application from performing in a functional manner.
- The Installing The Application section explains how any user can install the application onto their Android mobile device.
- The Navigating The Application section explains how the user interface operates, what the workflow design of the application is like and how to use the app.

Contingencies

You must have an Android Version 4.0 mobile device or later to run the application properly. If you are utilising an Android Virtual Machine, you must use Android Version 4.1 later or the application may not function properly.

Installing The Application

In order to properly install the application, we recommend using the link to the Google Drive that contains the apk file of our project. The apk link can be found in the documentation folder of the gitlab repo or it can be found below within this documentation. Depending on what Android device you may have, there may be settings that might prevent the installation of this application on your device due to the application not being Google Play Store certified or approved. We recommend giving your device the permission to only install our application when prompted or notified.

<https://drive.google.com/drive/folders/1v2uG7Q3AX7i8vrIZoVPKV2v6Krf8DuS?usp=sharing>

Navigating The Application

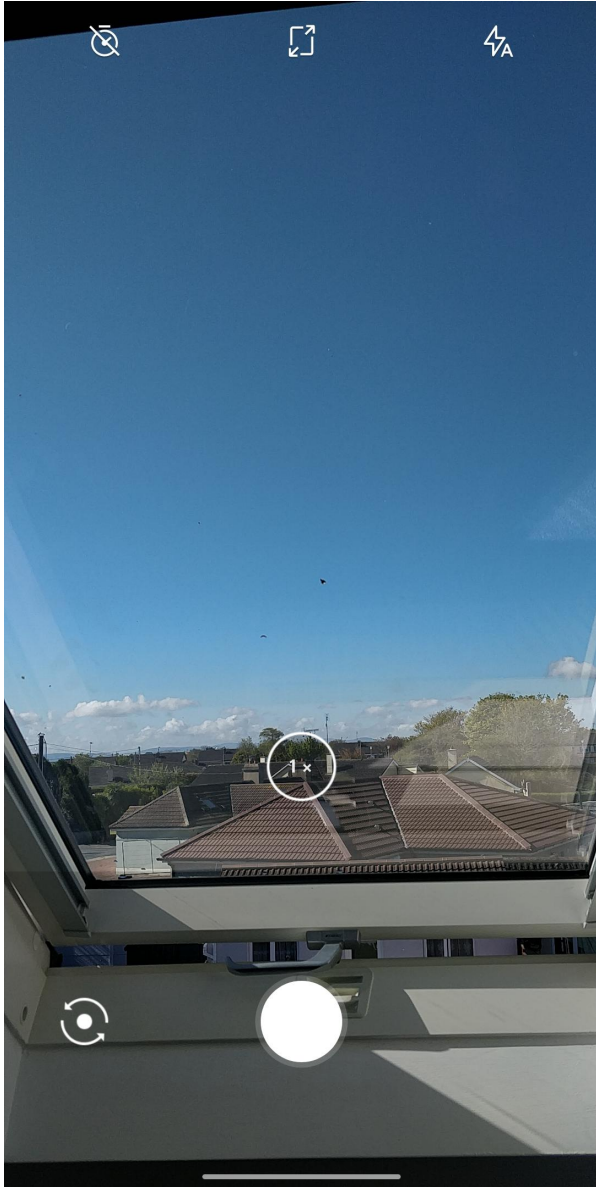
When entering the application for the first time, you will be entering onto the home page of the application. You will see the name of the application at the top of the screen and two buttons at the bottom of the screen. The home page of the application will look like this image below:



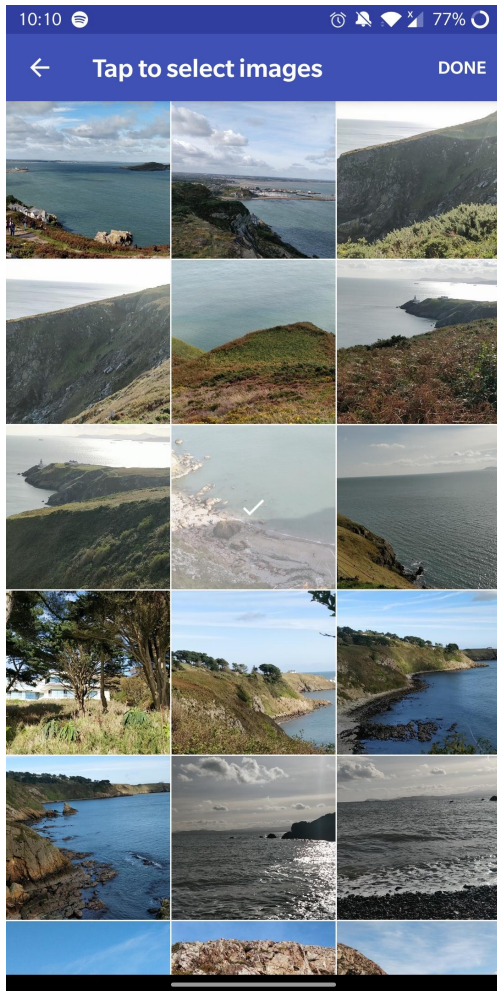
When you tap the “Take Photo” button, you will be taken to your mobile device’s default gallery that contains all of your images. The menu and images may differ depending on what particular Android device you have due to the differences in Android versions. Below is an image of what this menu might look like:



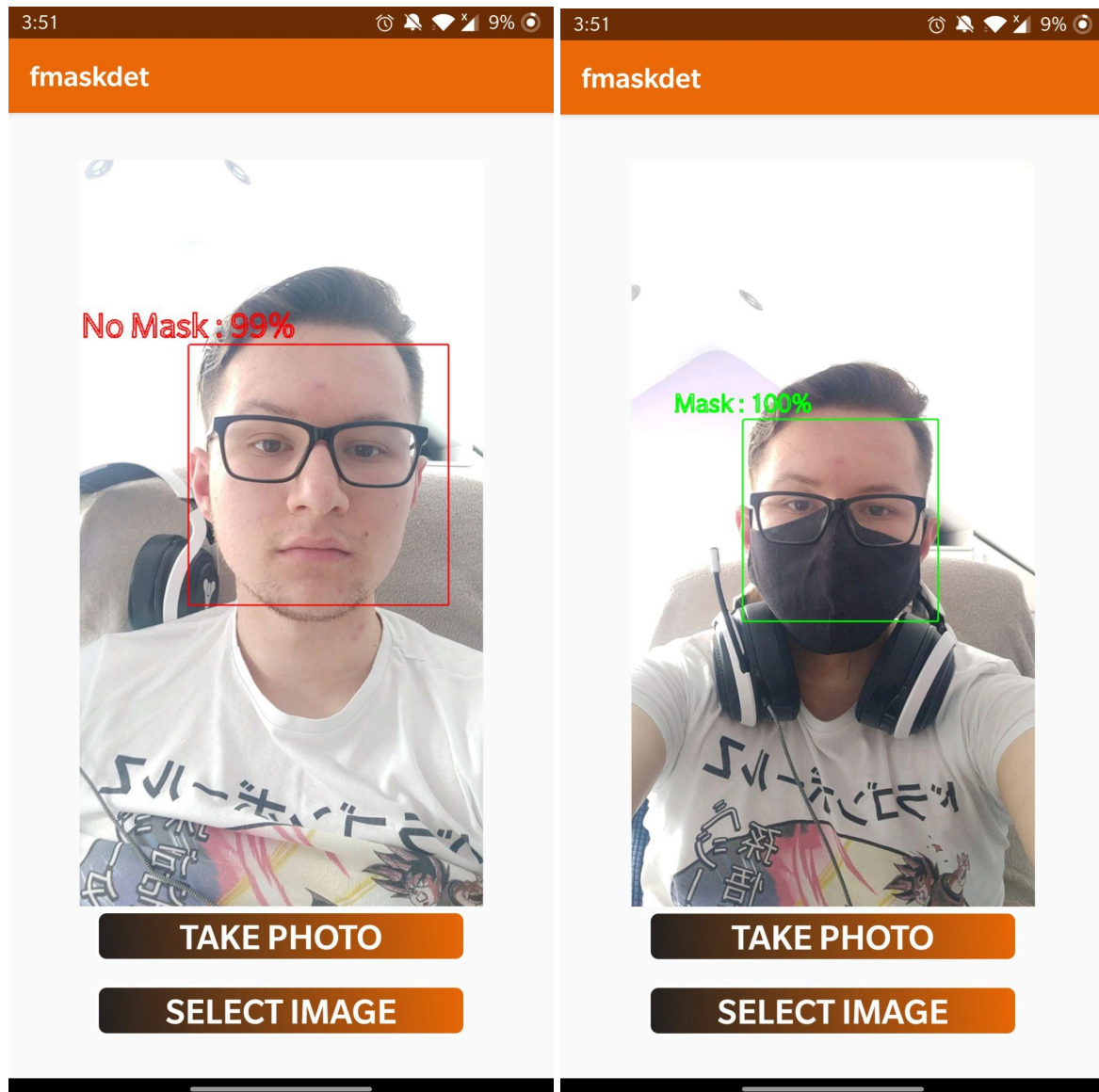
In the top right corner of the screen, there will be a camera icon that will allow you to take images of yourself within the application. The camera quality and layout will depend on the device you are using but an idea of what the layout might look like within the application can be found below:



The second button named “Select Image” will take you to the gallery of images that can be found on your mobile device. Here you can tap on any single image in order to perform the facial recognition contained on the application. After tapping an image, the word “Done” will appear in the top right corner in order to select the image to send into the facial recognition system. An image of what this might look like, depending on your Android device, can be found below:



After taking a picture with the camera or tapping “Done” button, you will be taken to the menu where you can view the results of the facial recognition system. The image will have a red box around the face if no mask is being worn, a green box around the face if a mask is being worn and there will be no box found on the image if the system has not detected a face. An idea of what these will could look like depending on your Android voice can be found below:



Training The Facial Recognition Model Yourself

In order to perform the training of a facial recognition model yourself, you are required to download files and code found in the `model_training_code` folder of the repository. The links to the dataset and unconverted facial recognition model can be found in the `model_training_code` directory as well. These are large files that could not be submitted to Gitlab and as a result, should be downloaded by clicking the lines in their respective files and placed into the `model_training_code` folder. It is necessary to have the proper packages installed for the training to be performed. These packages include Tensorflow, Tensorflow Lite, Imutils, Sklearn, Numpy and NLTK. When the files, dataset and packages have been properly installed, in order to begin training the facial recognition model you must type “python train_mask_detector.py -d dataset” into a command terminal. You must perform this inside the `model_training_code` folder on the terminal. The process of training may take a while depending on the machine you are using.

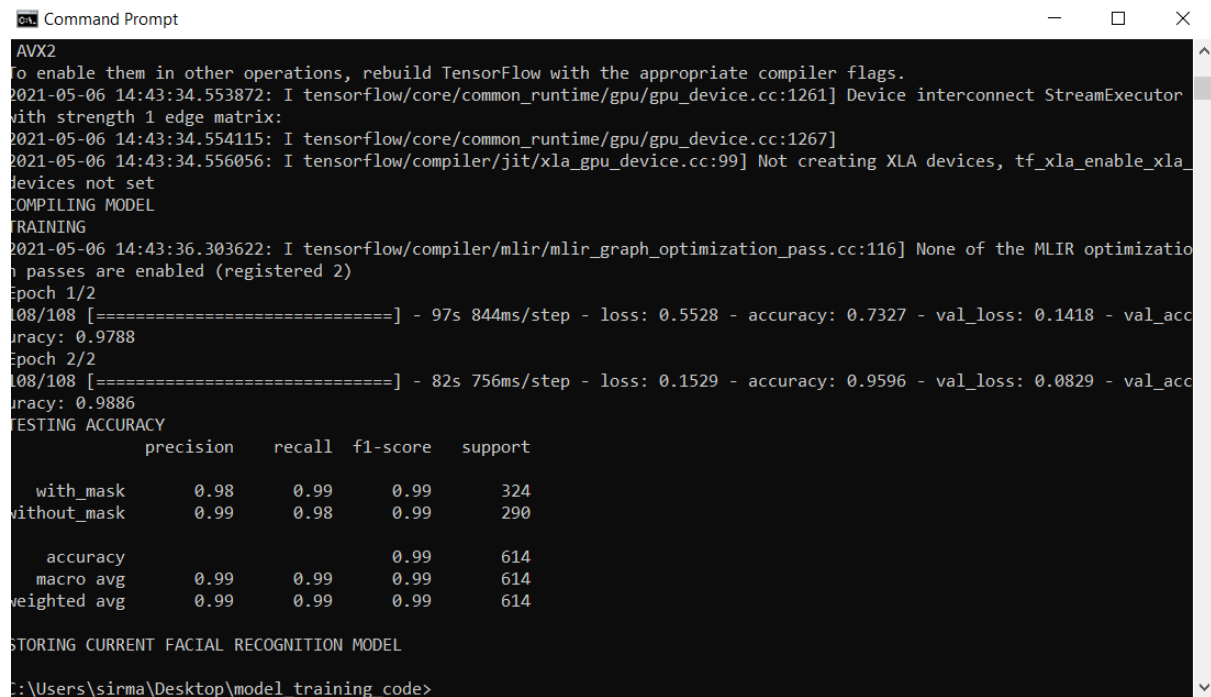
This is what the process should begin as if you have installed the necessary packages

```
Command Prompt - python train_mask_detector.py -d dataset
```

```
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
.\\Users\\sirma\\Desktop\\model_training_code>
python train_mask_detector.py -d dataset
2021-05-06 14:43:22.068896: W tensorflow/stream_executor/platform/default/dso_loader.cc:60] Could not load dynamic library 'cudart64_110.dll'; dlerror: cudart64_110.dll not found
2021-05-06 14:43:22.069091: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
```

LOADING IMAGES

This is an example of what the facial recognition model training could look like when finished:



```
Command Prompt
AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-05-06 14:43:34.553872: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1261] Device interconnect StreamExecutor
with strength 1 edge matrix:
2021-05-06 14:43:34.554115: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1267]
2021-05-06 14:43:34.556056: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_xla_
Devices not set
COMPILING MODEL
TRAINING
2021-05-06 14:43:36.303622: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimizatio
n passes are enabled (registered 2)
Epoch 1/2
108/108 [=====] - 97s 844ms/step - loss: 0.5528 - accuracy: 0.7327 - val_loss: 0.1418 - val_acc
uracy: 0.9788
Epoch 2/2
108/108 [=====] - 82s 756ms/step - loss: 0.1529 - accuracy: 0.9596 - val_loss: 0.0829 - val_acc
uracy: 0.9886
TESTING ACCURACY
      precision    recall  f1-score   support

 with_mask         0.98         0.99         0.99         324
without_mask         0.99         0.98         0.99         290

   accuracy
 macro avg         0.99         0.99         0.99         614
weighted avg         0.99         0.99         0.99         614

STORING CURRENT FACIAL RECOGNITION MODEL
C:\Users\sirma\Desktop\model_training_code>
```

If you wish to take this new model into the application and have the app use your newly trained model, you must perform Tensorflow Lite model conversion. This is the process of converting a Tensorflow model to a tflite model (Tensorflow Lite). Within the model_training_code folder, on the terminal you must type the following command:

**“ tflite_convert \ --keras_model_file=face_mask_detector.model \
--output_file=model.tflite ”**

This process may take a bit of time, but afterwards a converted version of the facial recognition model you created will be in the directory you are using. If you wish to use your version of the model in our app, you can replace our facial recognition model named model.tflite that can be found within the /src/fmaskdet/assets folder with the model you made. Then you can generate your own apk or emulate the application on Android Studio to use the app with your new model.