Project Title: Face Recognition based Attendance Application

A PROJECT REPORT

by:

Student Name	Student Number
Parth Sharma	251054678

submitted for the requirements of the degree

of

MASTER OF ENGINEERING (M.Eng)

Project Supervisor*:* Dr.Jagath Samarabandu

Department of Electrical and Computer Engineering
ECE – 9000

University of Western Ontario
1151 Richmond Street, London, Ontario
August 2019

Table of Contents

List of Figures

Abstract

The purpose of the project was to develop a Face recognition based Attendance Management System. The project will develop an Android application that can import a classlist (CSV) file with photos to train a face detection algorithm. The face detection algorithm used in this project is the Facenet library, which is one of the best performing facial detection libraries in today's technology. The classification/ recognition in the faces detected was done by implementing the SVC(Support Vector Classifier) model after the implementation of the facenet model(One shot learning model). The program is used to take a series of photos of each class, extract faces and present the recognition results as a series of face images along with the student name to the instructor. The instructor then can manually correct any mis-identified students and use these labels to further train the program to improve the accuracy. The instructor will be able to see the labels with the true and the captured image as a result. This feature ensures that the model is trained well on misclassified labels by correcting them manually on the app. The program produces a .xlsx file containing the attendance record in a form that is suitable for importing to Owl. It shows better than 90% accuracy after 2-3 misidentification corrections.

Acknowledgement

With a deep sense of gratitude, I would like to thank Dr. Jagath Samarabandu for his kind support and encouragement throughout the course of this project. There were times, that I faced challenges due to the constantly evolving nature of the technologies involved, as every release had newer features being offered and certain older aspects that were deprecated. Dr.Samarabandu offered me valuable guidance and insight on how to overcome these challenges and also mentored me about the various software development principles and approaches that had to be undertaken to ensure that the software was engineered was robust and possessed a seamless interface. It was truly a very enriching experience to work under Dr.Samarabandu.

Last but not the least, I would also like to thank my parents who have always been my pillar of support and strength.

Project outline

The objective of this project is to develop a face-recognition based attendance marking android application. The application will enable the professor to add a classroom of students in the application by entering their name, student id and their face photo. New photos can be added for each student by using the "Add picture" button which allows the professor to pick photos saved in the mobile. It is to be noted that adding more pictures for each student will increase the accuracy of our model and will give better results.

The user can import a csv of students which will automatically enter all the students in the application with their respective names and Student IDs. The user can then easily import student photos from his mobile which will map each photo with their respective student portfolio. It is to be noted that, in order to get the mapping done, the label of each photo should be the Student ID of the person in that photo.

The user can capture photo which opens the camera and allows the user to click a group/individual photo. The application will return back the predicted student numbers of the people present in the photo. To make it easier for the user, the user will be able to see both the predicted and the actual faces and labels for the currently clicked photo.

The user can then correct any misidentified people in the photo by updating the incorrectly predicted labels of the people with the correct labels. "Update" button will update the labels and mark the attendance of the students with the new corrected labels. The user will be able to see the predicted individual faces of the students in their correct respective student portfolios.

The user can then observe the attendance for the students one at a time by entering the Student ID or by clicking on "Export student reports" which will show the attendance of the whole classroom between the dates selected by the user in the form of an Excel sheet. This can be saved in the mobile of the user and exported to OWL.

Development Environment & Technologies

Development languages

- **Python:-** Python is an interpreted, high-level, general-purpose programming language.

  The reason for using python for developing the backend of this project is because python offers a lot of libraries for data science, machine learning. Although Java has been used for developing the Front End Android Application, developing the backend in Java would have been very difficult because of the limited options it provides for practicing machine learning techniques.

- **Java:-** Java is a general-purpose programming language that is class-based, object-oriented, and designed to have as few implementation dependencies as possible.

  The project requires to build an Android Application, Java has been used for developing the same. Since, the backend for this application is written in python, I tried to develop the frontend in python as well using KIVY (a free and open source Python library for developing mobile apps), but KIVY did not offer a good and flexible enough medium to develop and code the required Android Application. Hence, Java was picked as the best option for developing the front end that could specifically run on Android systems.

IDEs

- **Pycharm**

  PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda.

  The reason to choose Pycharm is that Pycharm offers an easy way to install all the required packages in the IDE itself. PyCharm provides smart code completion, code inspections, on-the-fly error highlighting and quick-fixes, along with automated code refactorings and rich navigation capabilities.

- **Android studio**

  Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as the primary IDE for native Android application development.

  The reason to choose Android Studio stands for itself as the front end of this project requires an android application, hence this IDE has been used.

Firebase

Firebase is a mobile and web application development platform developed by Firebase. Firebase Storage provides secure file uploads and downloads for Firebase apps, regardless of network quality. The developer can use it to store images, audio, video, or other user-generated content. Firebase Storage is backed by Google Cloud Storage. Firebase Cloud Firestore is the successor to Firebase's original databasing system, Real-time Database, and allows for nested documents and fields rather than the tree-view provided in the Real-time Database.There are multiple reasons as to why firebase has been used for this project.

- Firebase has been used in this project to maintain the database of the students in the form of “Students” collection, where all the images for each student and their Student Numbers, names are stored.
- Firebase also maintains the database of Attendance in the form of “Attendance” collection which contains the record of attendance for each date, details of every student whose face was recognized by the application.

Ngrok

ngrok is an application that gives you external (internet) access to your private systems that are hidden behind NAT or a firewall. It's basically a super slick, encrypted TCP tunnel that provides an internet-accessible address that anyone can get to, and then links the other side of that tunnel to functionality running local.

The reason to choose ngrok is simply to make sure that the front end android application and the backend are connected. The localhost (http://127.0.0.1:5000/) can be accessed as a public URL by using ngrok. The public URL is then used to get results from backend for our application.

Facenet

Face recognition is a computer vision task of identifying and verifying a person based on a photograph of their face. FaceNet is a face recognition system developed in 2015 by researchers at Google that achieved then state-of-the-art results on a range of face recognition benchmark datasets. The FaceNet system can be used broadly thanks to multiple third-party open source implementations of the model and the availability of pre-trained models. It is a system that, given a picture of a face, will extract high-quality features from the face and predict a 128 element vector representation these features, called a face embedding.

The FaceNet system can be used to extract high-quality features from faces, called face embeddings, that can then be used to train a face identification system. The model is a deep convolutional neural network trained via a triplet loss function that encourages vectors for the same identity to become more similar (smaller distance), whereas vectors for different identities are expected to become less similar (larger distance). The focus on training a model to create embeddings directly (rather than extracting them from an intermediate layer of a model) was an important innovation in this work.These face embeddings were then used as the basis for training classifier systems on standard face recognition benchmark datasets, achieving then-state-of-the-art results.

The reason behind choosing facenet as the model for face recognition for this application are:-

- Facenet is a one shot learning model which makes it ideal for this project as we only have 1 image for every student to start with.
- The accuracy on LFW(Labelled Faces in the Wild) for the model 20180402-114759 is 0.99650+-0.00252 which is better than other Deep models like Deepface, Openface, etc.
- We need to first detect and align the face before feeding it to the FaceNet model. This is achieved with a Multi-task cascaded convolutional neural networks.
- Facenet employs the triplet loss function. One of the major advantages of the triplet loss is that it tries to be less "greedy" than the contrastive loss (which considers pairwise examples). This is because the triplet loss takes an anchor example and tries to bring positive examples closer while also pushing away negative example.

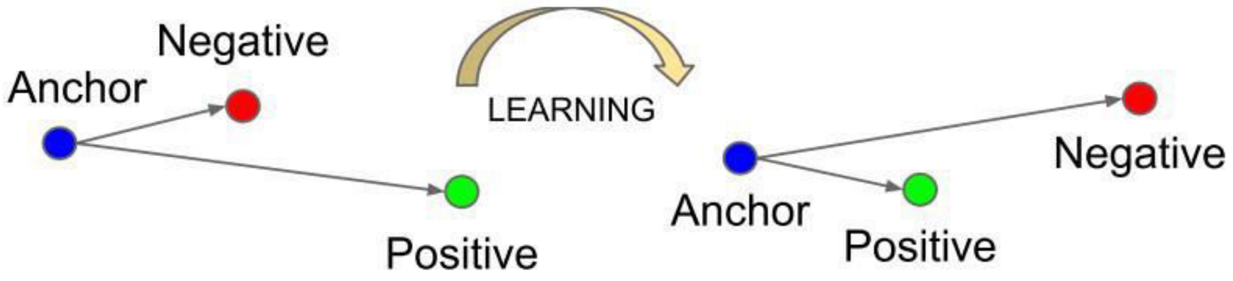


Figure 1. Triplet Loss Function

Figure 1 shows how the triplet loss function works. Triplet loss is only used in training the neural network, and doesn't need to use photos of the people you wish to verify. You are not teaching the network to recognise the faces in your training set, but rather you are teaching it to minimise the difference in output between photos of the same person "dist(A,P)" while maximising the difference between photos of different people "dist(A,N)".

Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program.

The reasons for choosing Flask in this project are:-

- Integrated support for unit testing.
- Built-in development server and fast debugger.
- HTTP request handling function
- Highly flexible
- It is easy to deploy the flask in production

Android Application User Interface

This section will explain the user interface of the android application.

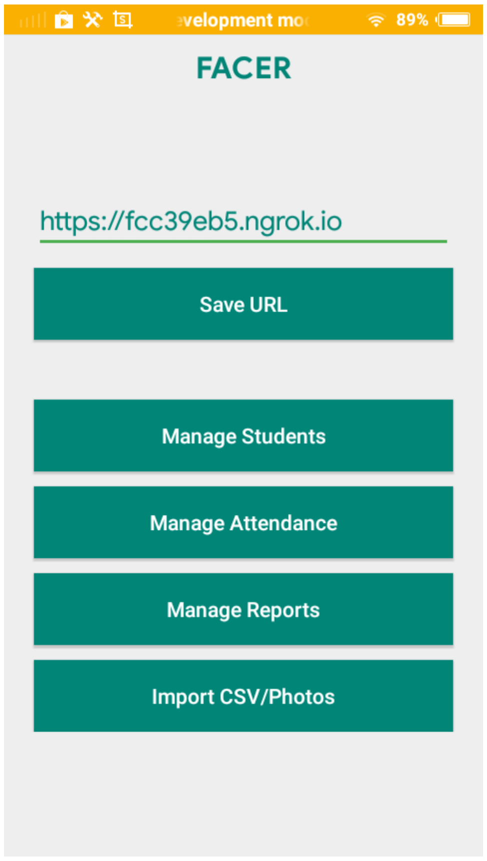


Figure 2. Application Main Screen

The above shown figure depicts the home/main page of our Face recognition Android Application.

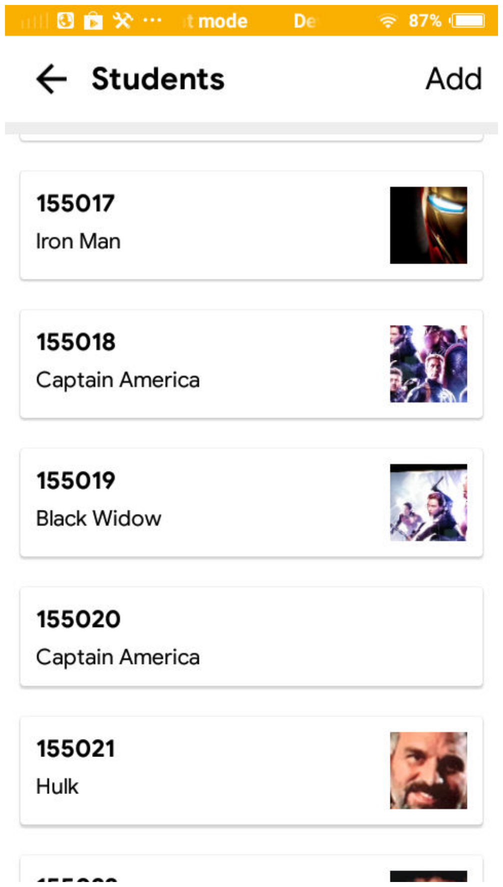
1. Save URL:-

In the Android App Main screen, we can enter and save the URL provided by ngrok. This URL is entered to build the server connectivity. After saving the URL in the android application, this URL will be saved in Firestore for Server Connection.

Ngrok is providing this URL and this will be needed for building a connection between the backend and the Android Application.

2. Manage Students:-

All students are stored in Firestore as documents under the Students collection.


Students

155017
Iron Man

155018
Captain America

155019
Black Widow

155020
Captain America

155021
Hulk

Figure 3. Student List Page

The above figure shows the screen when you click the “Manage Students” button.

The user can easily see every student in the classroom here. Each student details can be viewed individually by clicking on that respective student. The student’s name and Student ID can be seen with the student’s photo.

Now, in order to add a new Student, the user can simply click on “Add” at the top right of the Application’s screen.

This part is clearly shown in the figures below.

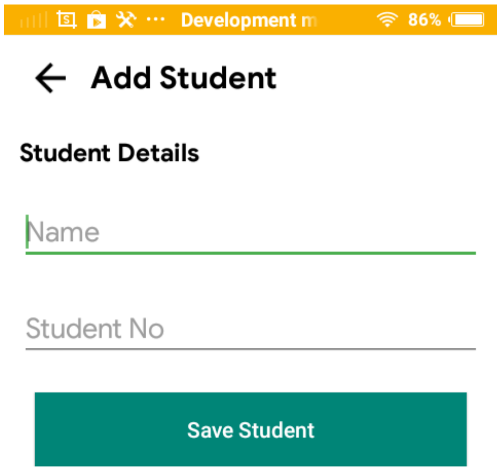
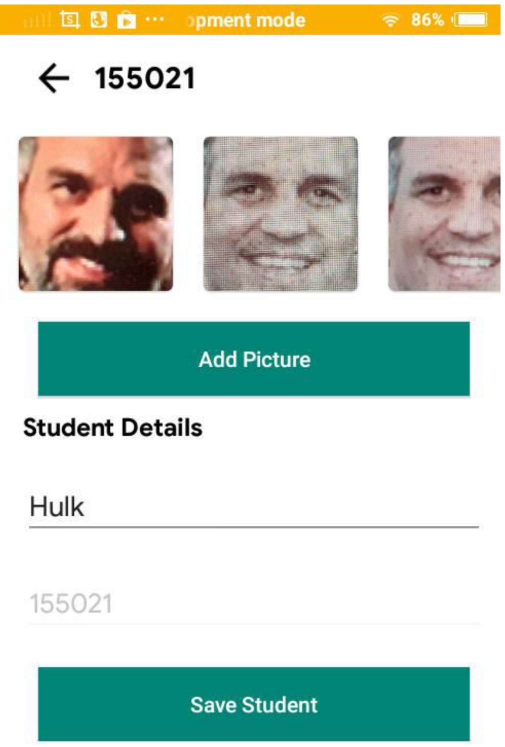


Figure 4. Add a New Student



Hulk

155021

Figure 5. Student Details

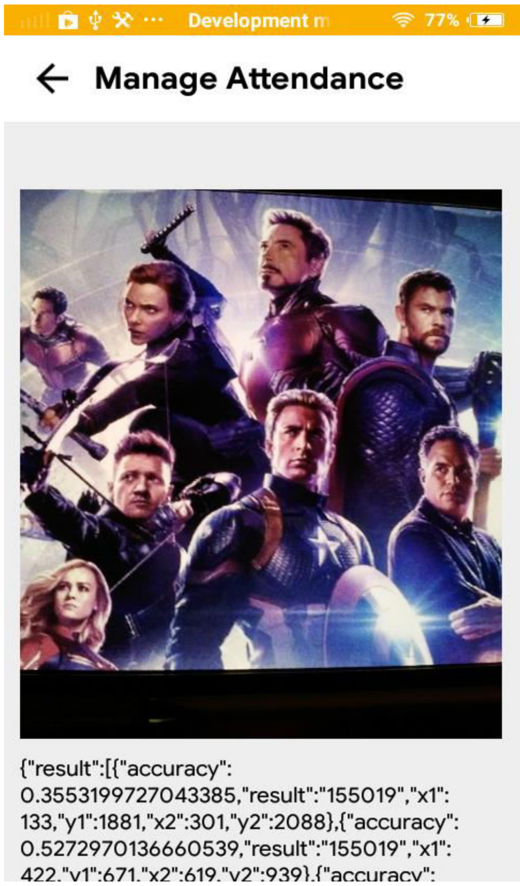
Figure 4 shows the screen that opens when the user clicks the "Add" button on the "Manage Students" Page. The user can easily enter the details for a new student and save it by clicking "Save Student". This will save the new student to the Firebase and the user will be able to see the new student in the "Manage Student" page.

Figure 5 clearly shows the student details of a saved student.

The user can add more pictures for that student by clicking on "Add picture" and "Save Student".

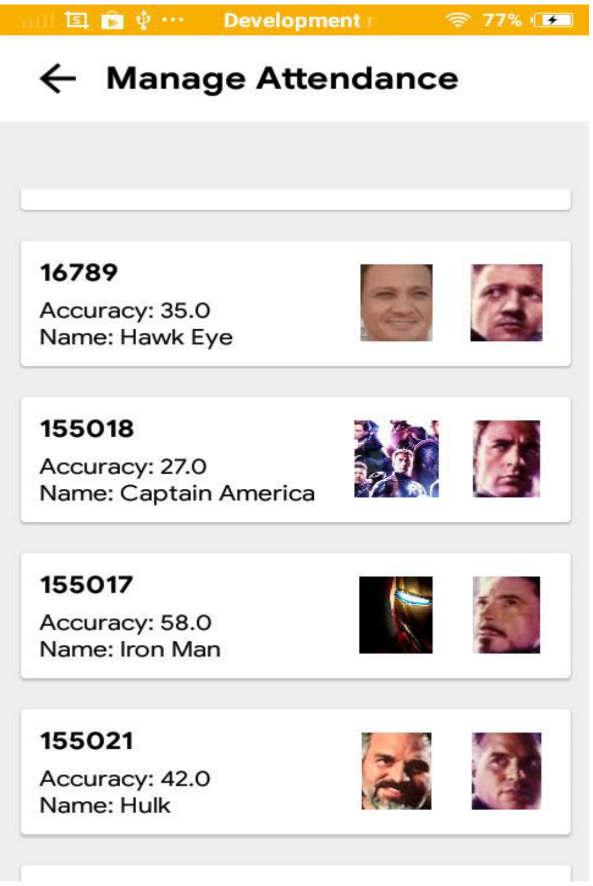
3. Manage Attendance:-

This is the section of our App that will allow the user to click a group photo of the classroom and get the results as shown in the figures below.



```
{"result":[{"accuracy":
0.3553199727043385,"result":"155019","x1":
133,"y1":1881,"x2":301,"y2":2088},{"accuracy":
0.5272970136660539,"result":"155019","x1":
422,"y1":671,"x2":619,"y2":939},{"accuracy":
```

Figure 6. Recognition Results



16789
Accuracy: 35.0
Name: Hawk Eye

155018
Accuracy: 27.0
Name: Captain America

155017
Accuracy: 58.0
Name: Iron Man

155021
Accuracy: 42.0
Name: Hulk

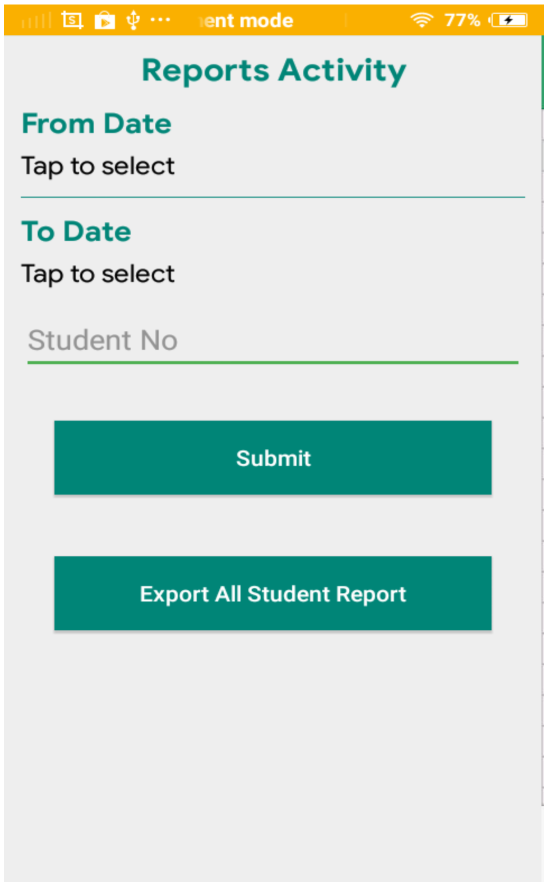
Figure 7. True faces(left) for Predicted Labels, Detected Faces(right)

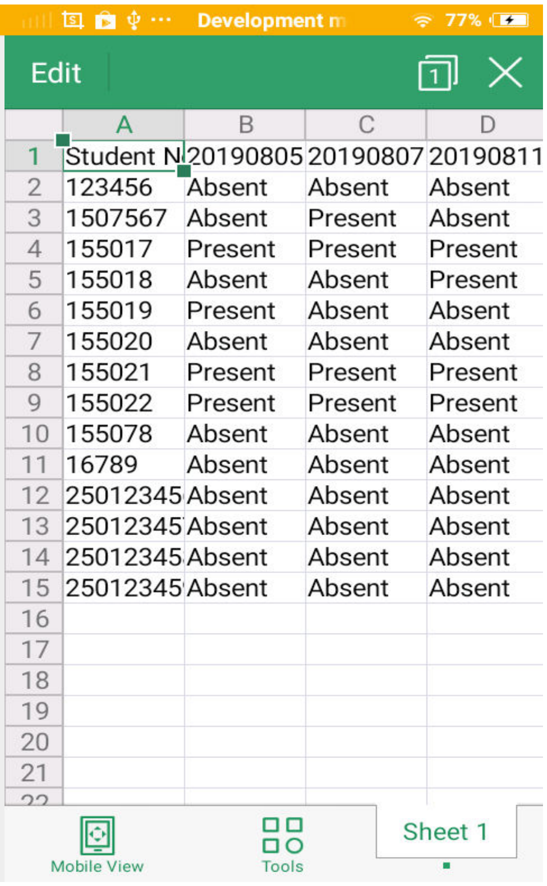
Figure 6 shows the image that was clicked in real time from our camera in the app. The app automatically gives the result as shown in Figure 8. As can be seen, the text below the clicked image is the response generated from the backend Facenet and SVC model. The result is generated in a list where the first dictionary shows the accuracy, Student Number, detected facial coordinates of the first detected student.

Figure 7 shows the predicted and the respective true images, which makes it easier for the user to implement the misidentification check. The faces on the left are the true images and the faces on the right are the faces from the clicked group image mapped to this true label.

4. Manage Reports:-

In the “Manage Reports” section of our android app, the user can see the attendance of the students individually or for the whole classroom between two dates that the user can specify.





	A	B	C	D
1	Student N	20190805	20190807	20190811
2	123456	Absent	Absent	Absent
3	1507567	Absent	Present	Absent
4	155017	Present	Present	Present
5	155018	Absent	Absent	Present
6	155019	Present	Absent	Absent
7	155020	Absent	Absent	Absent
8	155021	Present	Present	Present
9	155022	Present	Present	Present
10	155078	Absent	Absent	Absent
11	16789	Absent	Absent	Absent
12	25012345	Absent	Absent	Absent
13	25012345	Absent	Absent	Absent
14	25012345	Absent	Absent	Absent
15	25012345	Absent	Absent	Absent
16				
17				
18				
19				
20				
21				

Figure 8. Check attendance individually/ collectively between 2 dates

Figure 9. Excel sheet of attendance marked using this app

5. Import CSV/ Photos:-

This option enables a user to import a csv of students, which will automatically register all the students in the csv in the application. The user will be able to see all the newly registered

students with the old students. This saves a lot of time while registering a new batch of students. The user can also select multiple photos at once which will be automatically mapped to their respective students based on their Student Number.
Please note that in order to correctly map all the student photos to their correct profile in the application, the image names should be the Student Number's of the respective student's in the photo.

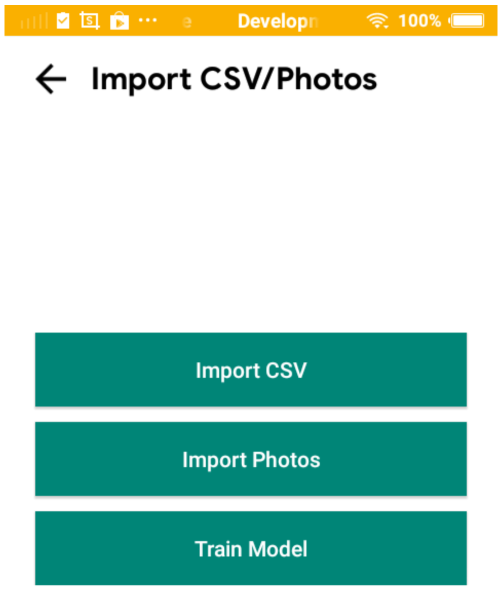


Figure 10. Import Csv/photos for registering new students

As discussed earlier, these buttons allow the user to register all the new students in a classroom very quickly. After importing the csv, the user can import the photos of all the newly registered students which will be automatically mapped to their profile on the basis of their Student Number. The user should make sure to train the model every time new photos have been imported so that the model can perform better on these newly added photos of the students.

Backend Files Description

1. Main.py:-
   This file creates a flask app using FLASK module. This will help to setup the server and provide links for the API like:-

   - / - This will open the home page that gives the message "Welcome to Facer".
   - /train - This is for training the model.
   - /compare - This is for running the recognition part where the segmented faces of the captured group photo is compared with the known pictures in Firebase storage. We get a resultant accuracy after the recognition with each segmented face from the group photo. To make it easier for the user, the user can see both the true and predicted labels for every face in the captured class group photo.
   - /uploadKnownPic – This will store an image with known label to its respective student in Firebase Storage.
   - /KnownPics – This will show all the images of students with known labels.
   - /clearPics - This will clear all the known images of students.
   - /allPics – This shows all the known and unknown images of students in the firebase storage.

2. Train_main.py:-
   This file takes all the preprocessed images stored in "pre_img" folder of our system and train the model on these images.

3. Recognize_face.py:-
   This file recognizes face from an image according to the input from the android app.

4. Preprocess.py:-
   This file contains the code to preprocess an image for training. This will be used when an image is uploaded on the server. This will create preprocessed images in "pre_img" folder from the "train_img" folder.

5. Picture.py:-
   This is the file to get all the preprocessed and known images of students from the system.

6. Identify_face_image.py:-

This file generates the bounding boxes coordinates for each of the detected face in our target group image. The resized and cropped face images will be shown as result in the application with the student’s respective predicted Names and Student Numbers.

7. Detect_face.py:-
   This file detects all the faces in our target group image of the students in a classroom.

8. Classifier.py:-
   In this file, we are loading the facenet model and using the SVC (Support Vector Classifier) to classify and recognize all the faces in our target group image. The model is then saved to the system using pickle.

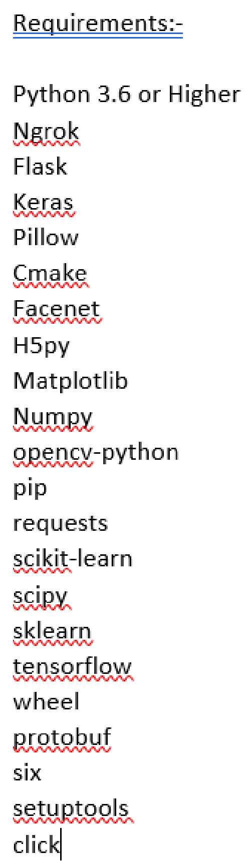
Application Setup

1. Server Connectivity Setup

In order to run the server, all the dependencies for the backend python files need to be installed before.

INSTALLING DEPENDENCIES WITH PYCHARM IDE:-

For developing the backend python files for this application, PYCHARM IDE has been used and is highly recommended for running this project because of the easiness of installing all the dependencies.



Requirements:-

Python 3.6 or Higher
Ngrok
Flask
Keras
Pillow
Cmake
Facenet
H5py
Matplotlib
Numpy
opencv-python
pip
requests
scikit-learn
scipy
sklearn
tensorflow
wheel
protobuf
six
setuptools
click

Figure 11. Required Dependencies for running Backend

INSTALLING DEPENDENCIES IN CMD:-

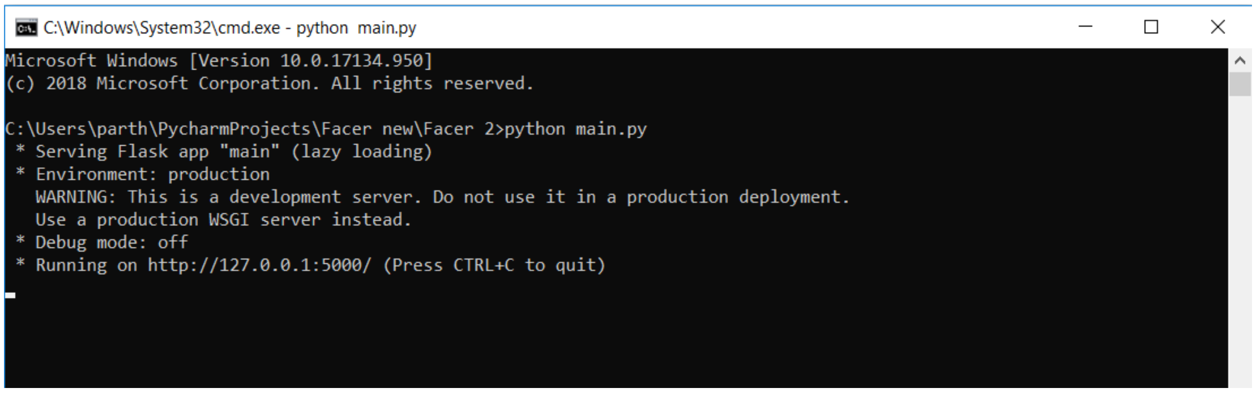
You can also install all these requirements by navigating to the main Backend Folder “Facer New” and use the command “ pip install “ one by one for each dependency.

The better way to install dependencies would be to make a text file “requirements.txt” and copy all these dependencies there.
Then navigate to “Facer New” folder, open cmd and write the command “pip install -r requirements.txt”

2. Running the Server:-

After the dependencies have been installed, the next steps are:-

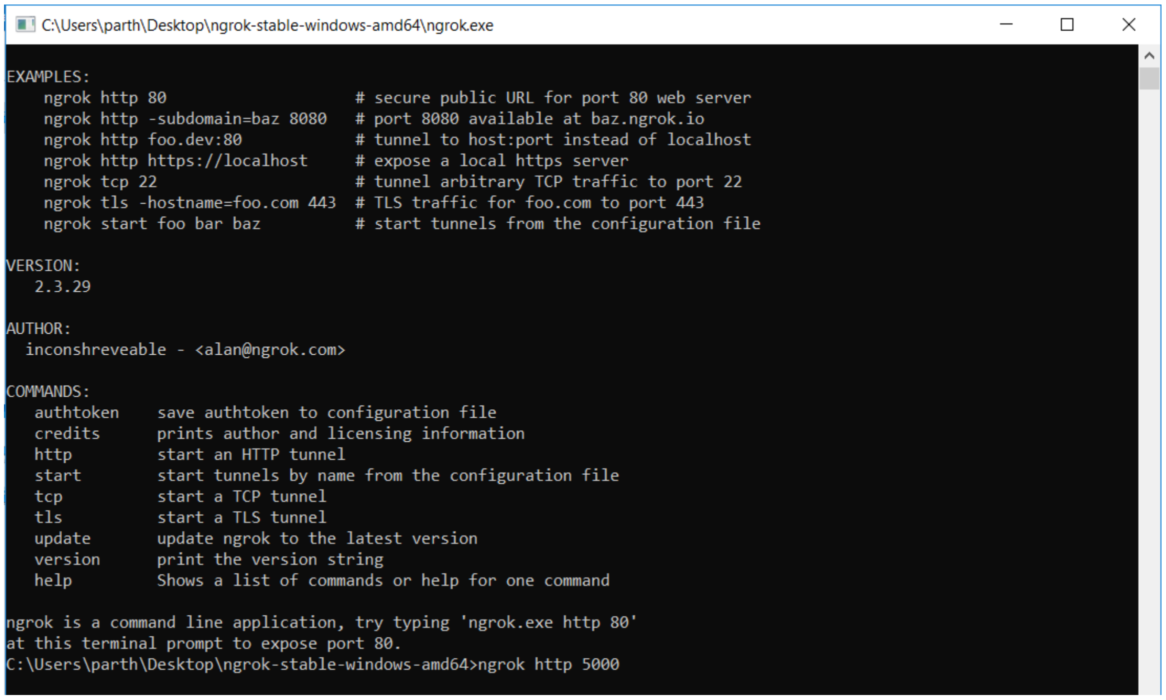
- Navigate to the “Facer New” folder.
- Run the command” set FLASK_APP=main.py” after opening the cmd in this folder.
- Then, run the command “FLASK_DEBUG=1 python3 -m flask run”. This will turn the server on. You can check by opening http://127.0.0.1:5000/ inside your browser. You will be able to see “Welcome to Facer” on your browser. This means that the server is running.



```
Microsoft Windows [Version 10.0.17134.950]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\parth\PycharmProjects\Facer new\Facer 2>python main.py
 * Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Figure 12. Running the server

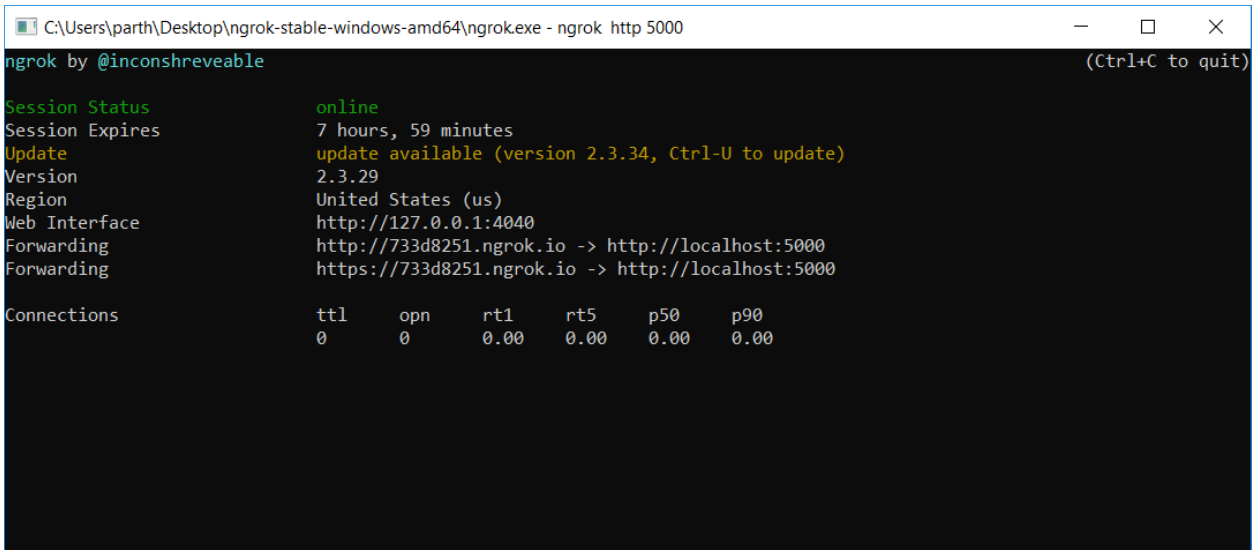
- The next step is to make this URL public, so that our Android Application is able to access it.
- Install ngrok on your system by downloading it from https://ngrok.com/download.
- ngrok exposes local servers behind NATs and firewalls to the public internet over secure tunnels. So, it is good for running mobile apps connected to a locally running backend.
- To do this, navigate to the folder where ngork is downloaded. Run the ngrok application which will show you the screen below, and type “ngrok http 5000” to get a public URL that will be used in our android application.



```
C:\Users\parth\Desktop\ngrok-stable-windows-amd64\ngrok.exe

EXAMPLES:
    ngrok http 80                    # secure public URL for port 80 web server
    ngrok http -subdomain=baz 8080   # port 8080 available at baz.ngrok.io
    ngrok http foo.dev:80            # tunnel to host:port instead of localhost
    ngrok http https://localhost     # expose a local https server
    ngrok tcp 22                     # tunnel arbitrary TCP traffic to port 22
    ngrok tls -hostname=foo.com 443  # TLS traffic for foo.com to port 443
    ngrok start foo bar baz          # start tunnels from the configuration file

VERSION:
   2.3.29

AUTHOR:
  inconshreveable - <alan@ngrok.com>

COMMANDS:
   authtoken    save authtoken to configuration file
   credits      prints author and licensing information
   http         start an HTTP tunnel
   start        start tunnels by name from the configuration file
   tcp          start a TCP tunnel
   tls          start a TLS tunnel
   update       update ngrok to the latest version
   version      print the version string
   help         Shows a list of commands or help for one command

ngrok is a command line application, try typing 'ngrok.exe http 80'
at this terminal prompt to expose port 80.
C:\Users\parth\Desktop\ngrok-stable-windows-amd64>ngrok http 5000
```

Figure 13. ngrok command

After running this command, we will be able to get the public URL we need for running our backend on the android application.



```
C:\Users\parth\Desktop\ngrok-stable-windows-amd64\ngrok.exe - ngrok  http 5000
ngrok by @inconshreveable                                                 (Ctrl+C to quit)

Session Status                online
Session Expires               7 hours, 59 minutes
Update                        update available (version 2.3.34, Ctrl-U to update)
Version                       2.3.29
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://733d8251.ngrok.io -> http://localhost:5000
Forwarding                    https://733d8251.ngrok.io -> http://localhost:5000

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Figure 14. ngrok generated public URL for running backend at our frontend

As shown in the image above, the backend is now accessible through the public URL https://733d8251.ngrok.io .

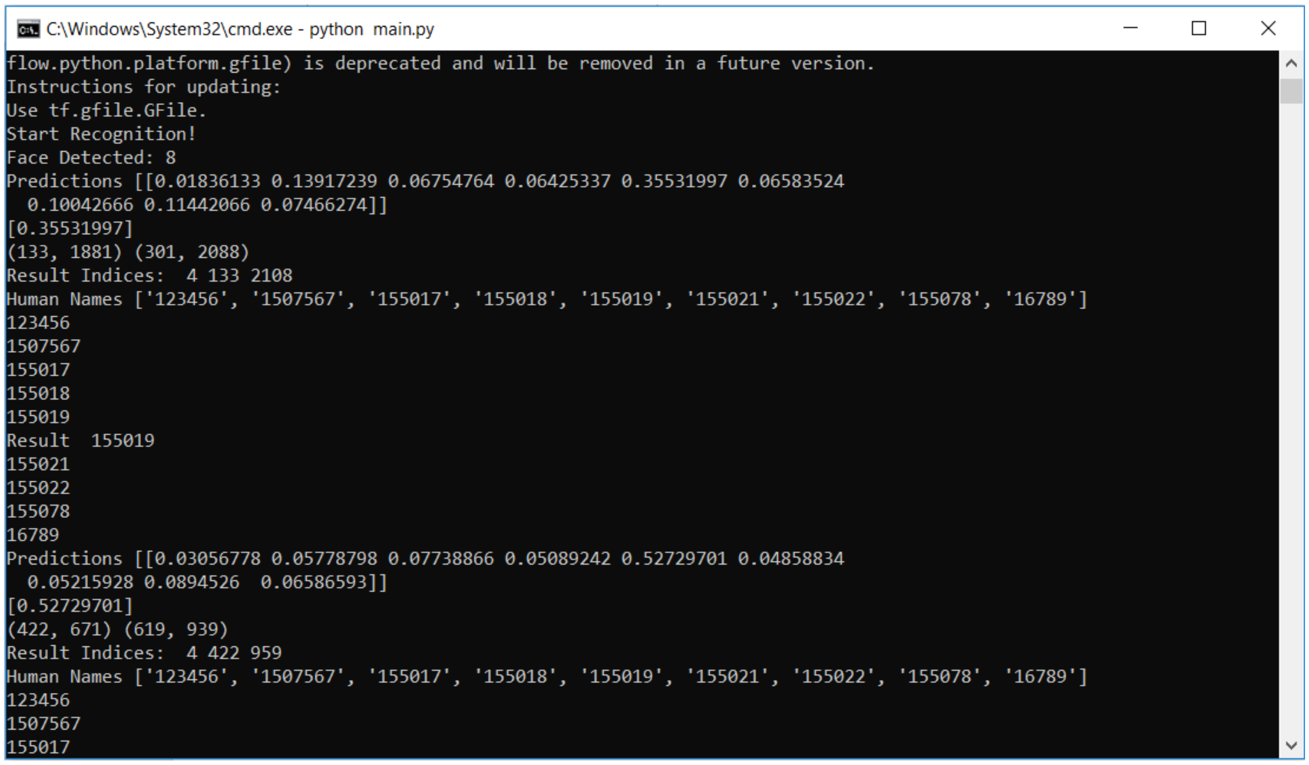
Note that this public URL will be different everytime ngrok is run again. So, you need to enter this URL in the Application everytime. Also, it is recommended to always use the secure public URL with "https" extension.

Checking the Results in cmd, ngrok

The below image shows the result that you will be able to see in your cmd where the FLASK is running on http://127.0.0.1:5000/ after you click the group image of the classroom in the application from your android phone.

The user will be able to see the response from the backend in the cmd which shows:-

- Number of faces detected in the group image.
- The facial coordinates of the detected faces in the group image.
- The predicted Student Numbers of the detected faces in the image.
- The accuracy of each predicted Student Number for every detected face.

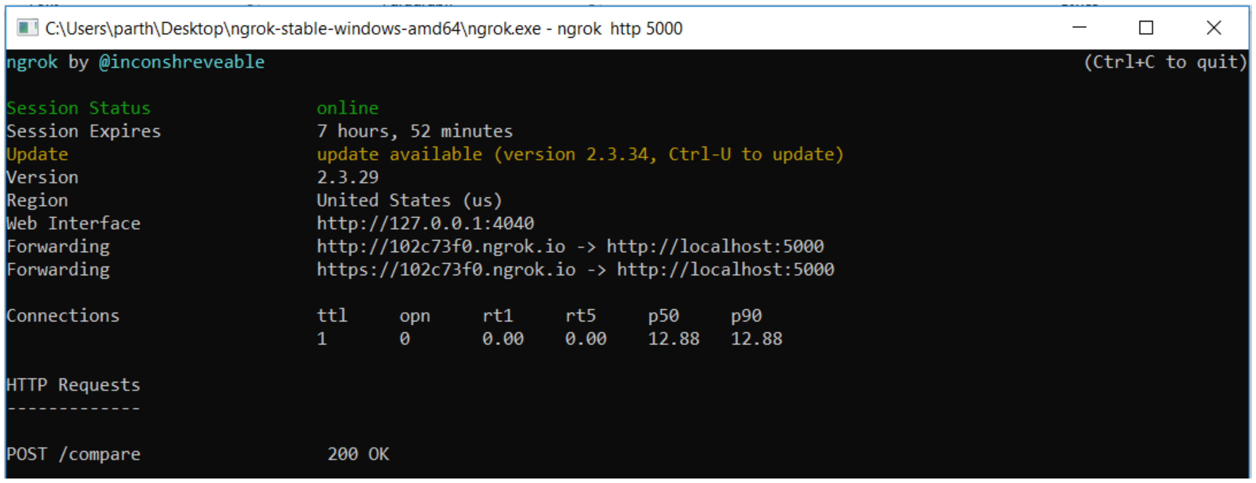


C:\Windows\System32\cmd.exe - python main.py

```
flow.python.platform.gfile) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.gfile.GFile.
Start Recognition!
Face Detected: 8
Predictions [[0.01836133 0.13917239 0.06754764 0.06425337 0.35531997 0.06583524
  0.10042666 0.11442066 0.07466274]]
[0.35531997]
(133, 1881) (301, 2088)
Result Indices:  4 133 2108
Human Names ['123456', '1507567', '155017', '155018', '155019', '155021', '155022', '155078', '16789']
123456
1507567
155017
155018
155019
Result  155019
155021
155022
155078
16789
Predictions [[0.03056778 0.05778798 0.07738866 0.05089242 0.52729701 0.04858834
  0.05215928 0.0894526  0.06586593]]
[0.52729701]
(422, 671) (619, 939)
Result Indices:  4 422 959
Human Names ['123456', '1507567', '155017', '155018', '155019', '155021', '155022', '155078', '16789']
123456
1507567
155017
```

Figure 15. Checking final result in cmd

Figure 13 shows the response from the backend in the cmd. This is useful for debugging and troubleshooting the backend responses.



```
C:\Users\parth\Desktop\ngrok-stable-windows-amd64\ngrok.exe - ngrok  http 5000
ngrok by @inconshreveable                                                        (Ctrl+C to quit)

Session Status                online
Session Expires               7 hours, 52 minutes
Update                        update available (version 2.3.34, Ctrl-U to update)
Version                       2.3.29
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://102c73f0.ngrok.io -> http://localhost:5000
Forwarding                    https://102c73f0.ngrok.io -> http://localhost:5000

Connections                   ttl     opn     rt1     rt5     p50     p90
                              1       0       0.00    0.00    12.88   12.88

HTTP Requests
-------------

POST /compare                  200 OK
```

Figure 16. Checking HTTP requests in ngrok

Figure 14 shows the response in the ngrok. The user will be able to see all the HTTP requests that are being made from the application and their respective response status from the backend.

Installation Guide

The backend code for running this project can be installed from my github repository by using the following command in your local github repository:-

git clone https://github.com/pshar33/face_recognition_backend.git

This repository consists of the backend code and the server code. The user can follow the instructions in this report to setup the backend and server on their own system. The user should make sure to install all the dependencies before running the server as discussed in this report before.

The frontend code for running this project on your android device can also be installed from my github repository by using the following command in your local github repository:-

git clone https://github.com/pshar33/face_recognition_frontend.git

In order to run this code make sure you have installed Android Studio on your system. Also, connect your android phone to your system when running the code in order to install the apk of this project in your android phone.

Scope of Future Work

Although the application currently performs well with photos. There is scope of improvement as this application can be extended to work with videos as well. As of now, the user can click a photo in real time and get the face recognition results from the backend to mark the attendance. It will make the application a lot better if the user could also record and see the face recognition results in live videostream. Also, the application currently uses the FLASK server to run the backend on our android phone, FLASK cannot be used for secure large-scale deployment of our backend for large number of users and large number of android devices. This can certainly be improved by using some other alternative to FLASK server in order to increase the scalability, security and reliability of this application.

Conclusion

Successful implementation of this project enabled me to familiarize myself with the best practices that are followed when developing android based applications and also obtain an in-depth understanding of the various features and aspects of Face-recognition libraries and deep learning models. It also enabled me to get a deep understanding of the principles of transfer learning, machine learning models, image pre-processing. The most attainable advantage from this project is that I can employ the skills that I have assimilated throughout the software development lifecycle of this project, towards building any data science project in the future that would require the knowledge of Deep learning as Backend and a good understanding of Android applications.

References

1. https://flask.palletsprojects.com/en/1.0.x/quickstart/
2. https://ngrok.com/docs
3. https://github.com/davidsandberg/facenet
4. https://medium.com/intro-to-artificial-intelligence/one-shot-learning-explained-using-facenet-dff5ad52bd38
5. https://developer.android.com/docs
6. https://docs.python.org/3/
7. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
8. https://firebase.google.com/docs
9. https://firebase.google.com/docs/android/setup
10. https://dev.to/stmcallister/create-a-quick-local-web-server-with-python-and-ngrok-k0#targetText=In%20Python%202.7%20(which%20is,current%20directory%20on%20port%208000.
11. https://machinelearningmastery.com/one-shot-learning-with-siamese-networks-contrastive-and-triplet-loss-for-face-recognition/