NYP NANYANG
THE INNOVATIVE POLYTECHNIC

ITI110 – DEEP LEARNING PROJECT

JANUARY 2021

FACIAL RECOGNITION

Yap Jing Yang (21B583W)

Low Jack Wai George (21B573H)

Looi Ying Choy (21A458H)

Ng Wei Xiang (21B576M)

Table of Contents

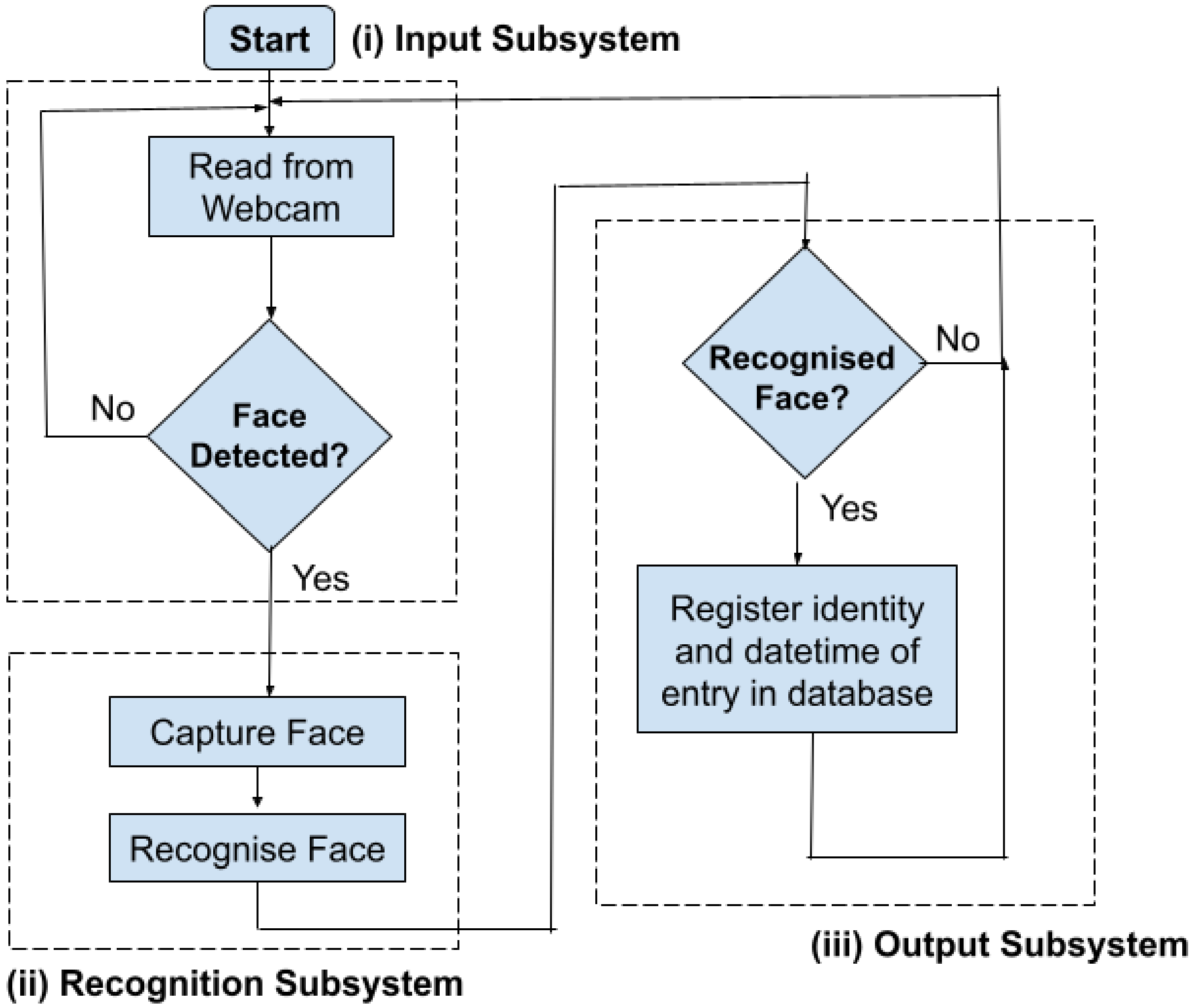
1. Formulation

1.1. Problem Statement

In today's world, we are using face recognition techniques for many security applications such as ICA customs clearance, Singpass login etc. For our project work, our group is going to develop a Neural Network based face recognition security system to identify authorized personnel for daily access to our office.

1.2. System Design & Framework

We had envisioned our security system as three major subsystems and this is illustrated in the below flowchart.



(i) Input Subsystem - we will be using the webcam to simulate as the input sensor to read the employee face.

(ii) Recognition Subsystem - this is the heart of our security system that will perform the detection and recognition of the human face. Given the image, we first need to detect the faces present in the image, then we can crop out the face in the image. Next, we will perform feature extraction on the cropped facial image, before matching it with our database of registered facial features to recognise the identity of the face.

(iii) Output Subsystem - lastly, once the face is recognised as an authorized personnel, the system will register the details of the entry such as identity, date and time into the database.

2. Methodology

For facial detection, we plan to use the pretrained MTCNN model, and for facial feature extraction, we intend to train our own Siamese Network to perform the task.

2.1. Dataset

We would be merging datasets from two sources. One of them is the Pins Face Recognition Dataset that is available on Kaggle, which consists of 105 celebrities and 17,534 cropped faces images collected from Pinterest. While there are about 200 face images per celebrity identity, the number of identities might not be comprehensive enough. As such, we complimented our dataset by merging it with the Labeled Faces in the Wild Dataset, which consists of 5,749 identities and 13,233 face images, with 1,680 identities having two or more images.

However, the Labeled Faces in the Wild Dataset also contains identities with only one or a few images. As the constraint that each identity needing to meet a certain minimum number of images might be imposed for certain training and evaluation, these identities will be dropped.

In addition, we also split the merged dataset into train-validation-test sets with disjoint identities while keeping the distribution of the number of images per identity similar. This is to ensure that the model will be able to generalise well on unseen identities.

3. Intended Experiments

As our focus is on the training of the facial feature extractor model and the recognition model, we would just be using the pretrained MTCNN model out of the box instead of exploring other methods for facial detection. However, we would be experimenting with different architectures and hyperparameter tuning to customise our Siamese Network and classifier model.

To evaluate the performance of our model, we would be using the classification metrics such as the accuracy on the testing data as described above.

4. Performance Tuning and Optimisation

As mentioned in the above section, we plan to explore several different architectures for the customized Siamese Network, such as the depth and width of the network, kernel and channel size of CNN, addition of regularizations and skip connections etc. We will also perform hyperparameter tuning for each of the architectures, which includes the optimizer, learning rate, activation functions etc. This would also include tuning of the classifier model, which may not be a neural network model.

To challenge the model classification, we will be using test images of our look-alikes (our parents, children or twins etc) to try and fool the model. Because these images will have a high resemblance to us, we will be excited to know the performance of our model.

5. Application-Based Integration

To integrate our application into the client system, we will then create the model image on Docker and run the Docker container in the Docker dashboard tool. We will improve the image with a customized working directory, and make it more stable by using environmental variables if necessary.

After we packaged the application into a docker image and saved it in a private repository, we will deploy it on the local server, the user needs to login in order to pull from the private repository. After that, we will use the docker-compose file on the server to deploy the application or services to the office server machine for front-end users. The door scanning machine will perform the scanning of the face.

Annex A. References

1) Pins Face Recognition Kaggle Dataset
   https://www.kaggle.com/hereisburak/pins-face-recognition
2) Labeled Faces in the Wild Dataset
   http://vis-www.cs.umass.edu/lfw/
3) FaceNet: A Unified Embedding for Face Recognition and Clustering
   https://arxiv.org/pdf/1503.03832.pdf
4) Tensorflow Documentation
   https://www.tensorflow.org/
5) Triplet Loss and Online Triplet Mining in TensorFlow
   https://omoindrot.github.io/triplet-loss