NYP NANYANG
THE INNOVATIVE POLYTECHNIC

ITI110 – DEEP LEARNING PROJECT

FINAL REPORT - Mar 2022

FACIAL RECOGNITION

Yap Jing Yang (21B583W)

Low Jack Wai George (21B573H)

Looi Ying Choy (21A458H)

Ng Wei Xiang (21B576M)

Table of Contents

1. Introduction

This final report dictates our conclusion, limitations faced by us as well as future developments that we would like to propose. Our approach is to do individual experiments on selected deep learning architectures and then carefully select our final model. We will then add the rest of the elements to complete this face recognition project.

2. Modeling Experimentations

Each of us selected a deep learning model architecture to experiment and will pick the best possible choice based on several factors to further fine-tune it as our final model. Below are our write-ups which dictate our effort, experiments, limitations and recommendations by us.

2.1. ResNet50 with Triplett Loss Architecture (Experiment done by Yap Jing Yang)

Yap Jing Yang had come up with a short 10-mins video(MileStone_Writeup_YapJingYang.mp4) to explain his portion of work. The video is submitted together with this report.

2.2. Siamese Network with Triplet Loss (Experiment done by Looi)

Algorithm for get triplet using “np.random.choice” for y val label and train label.
Negative image= random pick ‘n’ image.
Anchor,positive image= random pick ‘a’ not equal to ‘n’, followed by using ‘a’ to pick ‘p’ same person but different jpg file.
F1-score results (refer to Appendix 1):
i) 2 conv layer - 0.1
ii) Mobilenet (~0.6% of trainable conv layer set to true) - 0.24
iii) Resnet50 (only conv5 trainable set to true) - 0.27
GPU memory problem:
Run2 Resnet50, image size must reduce to 100x100.
Run3 Resnet50, image size reduce to 100x100 and batch size reduce to 32

2.3. EfficientFaceNet (Experiment done by Ng Wei Xiang)

The EfficientNet B0 was used as the backbone for the encoder model. Various methodologies are experimented and several features highlighted in the following sections.

2.3.1. Siamese Network

The first model is the standard siamese network with triplet loss that tries to pull embeddings of identical class together while pushing embeddings of different classes away. It comprises of the EfficientNet B0 backbone with its corresponding default input shape of 224 x 224, followed by a pooling layer, then a dense layer and lastly a L2 normalisation layer, which acts as the output embedding of the encoder. The normalisation layer provides an additional constraint that defines a relationship between the cosine similarity and euclidean distance metrics, and is especially useful if the latter is used as it would then be otherwise unbounded.

2.3.2. Semi-Freezing of Pretrained Backbone

While the model is initiated with the weights pretrained on ImageNet, our application is specifically on facial dataset and so the pretrained weights in the latter layers might not be as suitable. However, the first few layers would still be useful as they capture more basic structures such as edges. As such, several experimentations are done with different proportions of the backbone frozen midway.

2.3.3. Online Semihard Triplet Mining

As pointed out in the FaceNet paper, selecting the triplets at random often times leads to trivial triplets which are too easy. On the other hand, hard triplet mining is also an option and is in fact explored, it does not lead to model convergence. As such, semihard triplet mining is explored to ensure that the triplets are decently difficult so that the model could learn from them.

However, offline mining is extremely computationally expensive as it requires the model to process all valid triplet permutations of the entire dataset per epoch. Instead, online mining does it similarly but on the batch level, which is more efficient.

2.3.4. Global Depthwise Convolution

Inspired by the MobileFaceNets paper, instead of using the usual global average pooling to collapse the feature maps, a depthwise convolution layer with filter size equaling to the feature map size is used to act as the pooling functionality, but with the added relaxation that each pixel of each feature map can be treated with different importance.

2.3.5. Knowledge Distillation

Using FaceNet as the teacher, the encoder model is trained to produce embeddings as close as possible, based on cosine similarity or euclidean distance, to the target embedding produced by

FaceNet on the same input. While the FaceNet embedding might not be perfect, this allows our smaller encoder model to mimic FaceNet and accelerate its learning.

2.3.6. Encoder Model Evaluation

After training, the model will first be used to generate the embeddings for all the images in the testing dataset. Thereafter, for each class in the testing dataset, one of the embeddings will then be registered in the database. The unregistered images are then compared against all the registered embeddings and the class with the highest cosine similarity is chosen as the model prediction. Lastly, the classification report can be generated for comparison.

It should be noted that this performance would not be reflective of the actual performance on the ground as the concept of unidentified class is undefined. However, this evaluation methodology is sufficient at this point in time as we are only doing the evaluation for model comparison.

2.3.7. Encoder Model Comparison

The first is the model that is initialised with pretrained weights on ImageNet and trained with triplet loss. Unfortunately, the model only managed to achieve a testing accuracy score of about 0.46 and macro F1 of about 0.33. Although it is much higher than random guessing, which puts the accuracy score at about 0.0056, it is not close to ideal at all, considering that FaceNet managed to achieve an accuracy score of about 0.89 and macro F1 of about 0.76. This might be due to the limited dataset and capacity of the model.

The second model is trained using knowledge distillation with cosine similarity as the loss. This model managed to achieve a much better testing accuracy score of about 0.74 and macro F1 of about 0.62. While there is still a gap between the model and FaceNet, it seems to be pretty good considering that the model is more than five times smaller. The reason might be that the model finds it easier to learn from the teacher embeddings rather than through triplet loss, and it could leverage what the larger model has learned while keeping itself compact.

The third model is a combination of the other two models by initiating using the pretrained weights from distillation and trained with triplet loss. The idea is to hopefully finetune the distilled model to get a better performance. However, there is only a slight improvement in performance to a testing accuracy of about 0.76 and macro F1 of about 0.66. This is probably because the FaceNet embedding itself is not perfect and so this extra finetuning could help to push the embeddings slightly in the right direction.

2.3.8. Verification Models

After generating the embeddings, we explored using another shallow neural network that takes in two embeddings and outputs the probability of them being identical. Apart from trying with just the standard shallow classifier, some transformations are also introduced in order to achieve certain desirable properties.

2.3.9. Embedding Transformations

In the usual shallow neural network, the two embeddings are passed into the network separately followed by concatenation. This network however does not possess the commutative property that a function should have when calculating similarity. In other words, the output probability would usually be different by swapping the two input embeddings, when in fact we expect the probability to remain unchanged. As such, transformations are made on the two embeddings such that it retains the original information up to commutativity in the newly mapped space.

Also, rather than taking any possible transformation, the transformation of element wise multiplication and squared difference are used since they incorporate meaningful distance metrics, corresponding to cosine similarity and squared euclidean distance respectively, that might facilitate learning. It can easily be shown that the original input can be deduced given the new transformation up to commutativity.

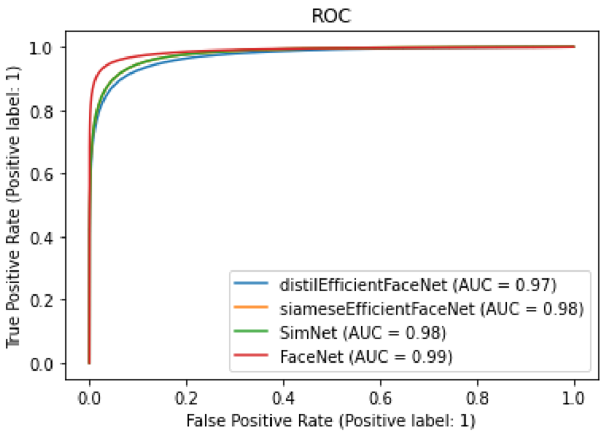
2.3.10. Verification Model Evaluation

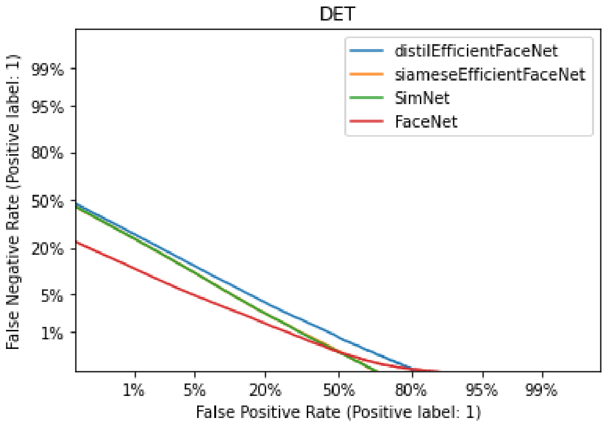
The evaluation methodology is slightly adjusted to fit the verification context. The embeddings are still first generated for all images in the testing dataset. Then, for each embedding, it is compared against all other embeddings to determine whether each pair is identical or not based on a certain threshold, which is taken to be the threshold at which a maximum of 0.1% of False Acceptance Rate (FAR) is obtained.

2.3.11. Verification Model Comparison

The first is the model that based its prediction directly by computing cosine similarity between two embeddings. It managed to achieve an accuracy score of about 0.98 and macro F1 of about 0.83, which seems acceptable, noting FaceNet accuracy score of about 0.99 and macro F1 of about 0.92.

However, the second model where a shallow classifier is used unfortunately also achieved the same performance as the first. As such, it does not seem worthwhile to have an additional classifier model.

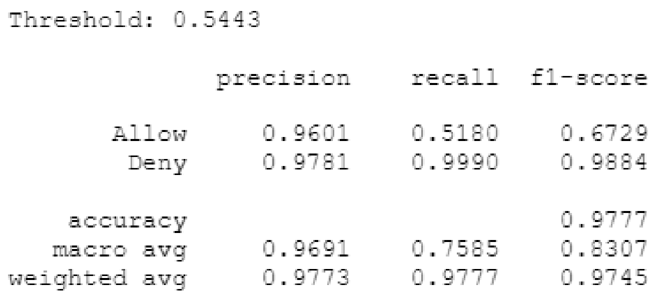


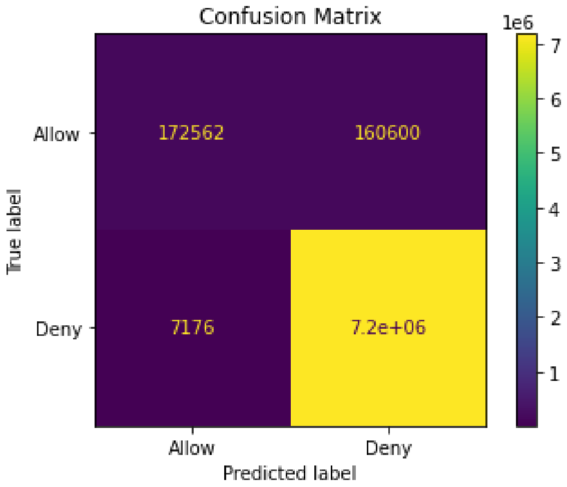


The above shows a summary comparison between the models. First, the distilEfficientNet is the model that is trained using knowledge distillation. Secondly, the siameseEfficientNet is the model that is using pretrained distilEfficientNet weights and further trained using triplet loss. Thirdly, the SimNet is the model that uses pretrained siameseEfficientNet and further trained with a shallow classification head. Lastly FaceNet is the pretrained FaceNet model.

2.3.12. Summary

All in all, the third encoder model with no additional classifier managed to achieve a decent performance and is the best among all other experimentation runs. Its performance is shown below.


```
Threshold: 0.5443

              precision    recall  f1-score

       Allow     0.9601    0.5180    0.6729
        Deny     0.9781    0.9990    0.9884

    accuracy                         0.9777
   macro avg     0.9691    0.7585    0.8307
weighted avg     0.9773    0.9777    0.9745
```



2.4. MobileNet with Triplet Loss Architecture (Experiment done by George)

MobileNetV1 is a lightweight deep NN based on streamlined architecture and built by using separable convolution. The core layer made by MobileNetV1 is a deep separable filter, and a deep separable convolution is a form of deconvolution. It proposes a high-performance architecture with hyperparameters, which can make the model smaller and the calculation speed faster, and it is efficient for face recognition systems.

This particular architecture is an experiment using the Triplet Loss function to train the neural network, and the output is a 128-dimensional vector space. The selected Triplets contain two matching face thumbnails and a non-matching face thumbnail, and the Loss function targets distinguishing positive and negative classes by distance boundaries.

Before starting the training and calculating the Triplet Loss to make the distance between the same images, we must apply L2 feature normalization after MobileNet CNN and face embedding before the Triplet Loss function. Ideally, the positive image should be closer to the anchor, and the negative image should be away from the anchor image.

The initial hyperparameter setting on the model is set with 30 epochs, 64 batch sizes, the learning rate is 0.02. The f1-score accuracy is 0.42, the average precision score is 0.46, and the average recall score is 0.44. This is not the ideal model as the minimum threshold is at least a score of 0.5, and the best target threshold is at least a score of 0.8. Next, fine-tuning the hyperparameters by changing the batch size to 128 and the learning rate to 0.002. The f1-score accuracy is 0.45, the average precision score is 0.49, and the average recall score is 0.47. On the final run, fine turning on the hyperparameters by changing the batch size to 64 and the learning rate to 0.0002. The f1-score accuracy is 0.46, the average precision score is 0.5, and the average recall score is 0.48.

Although changing different hyperparameter settings, the accuracy score does not meet the minimum threshold score of 0.5. However, overall, the computation time is reasonable fast due to the model's architecture. Therefore, it is worth exploring further and improving the accuracy by changing the model to the MobileNet related model and fine-tuning the parameters, FaceNet or Fast FaceNet model, for example.

3. Experimentations Conclusion

After weighing the pros and cons of the different models that we had experimented, we decided to use the model devised by Ng Wei Xiang. The main reason being his model is more complete as our models are lacklustre in several ways and his model has shown the potential to achieving higher heights with more training and fine-tuning.

4. Deployment

A simple application is made with the following capabilities.

1) Registration
   This feature enables users to register their faces. The captured face by the webcam is first converted into its embedding and then stored into the database under the user's identity.
2) Verification
   This feature enables users to test the verification of their faces. Similar to registration, the captured face by the webcam is first converted into its embedding. After which, the embedding is then compared to the registered embedding stored in the database.
3) Logging
   This feature logs the activities into the database for tracking purposes.

5. Future Developments

Due to time constraints, not all our ideas could be implemented. Some potential works that could be considered for future developments are described in the following sections.

5.1. Data Preprocessing

Before feeding the images into the model, the images could have been adjusted for sharpness as well as facial rectification to facilitate feature extraction.

5.2. Data Augmentation

The images could have been augmented with brightness and contrast so that the model could better generalised under different lighting conditions.

5.3. Loss Functions

Other loss functions such as SphereFace, CosFace and ArcFace could have also been explored.

5.4. Model Architecture

Apart from CNNs, transformers have also proven to be effective for vision tasks. As such, Vision Transformers could also have been experimented with as the encoder backbone. Another possible addition that could be experimented would be the Feature Pyramid Network, which could allow for the earlier feature maps to be used directly in the generation of the embedding layer.

5.5. Training Methodology

While we have tried a simple form of knowledge distillation, other forms of knowledge distillation methods, such as distillation of the intermediate layers, could have also been used as well.

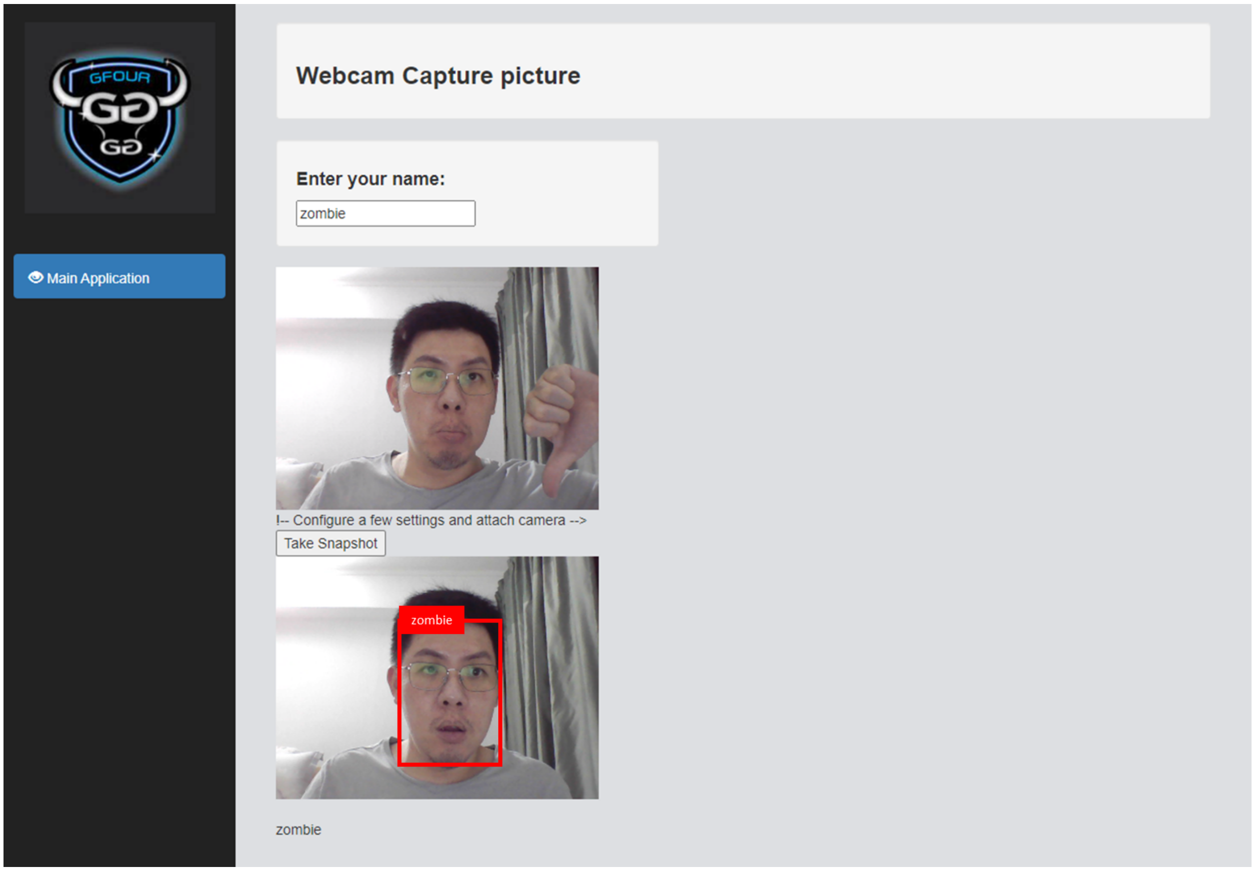
5.6. Verification Methodology

With the consideration that having a representative registered embedding is of importance, multiple embeddings could be registered per identity and the verification comparison could be carried out against the collection of registered embeddings of that particular identity instead.

5.7. Web-based Interface

We will be deploying a website interface so that users can access the application via any web-based device or mobile application.

<ins>Website version</ins>

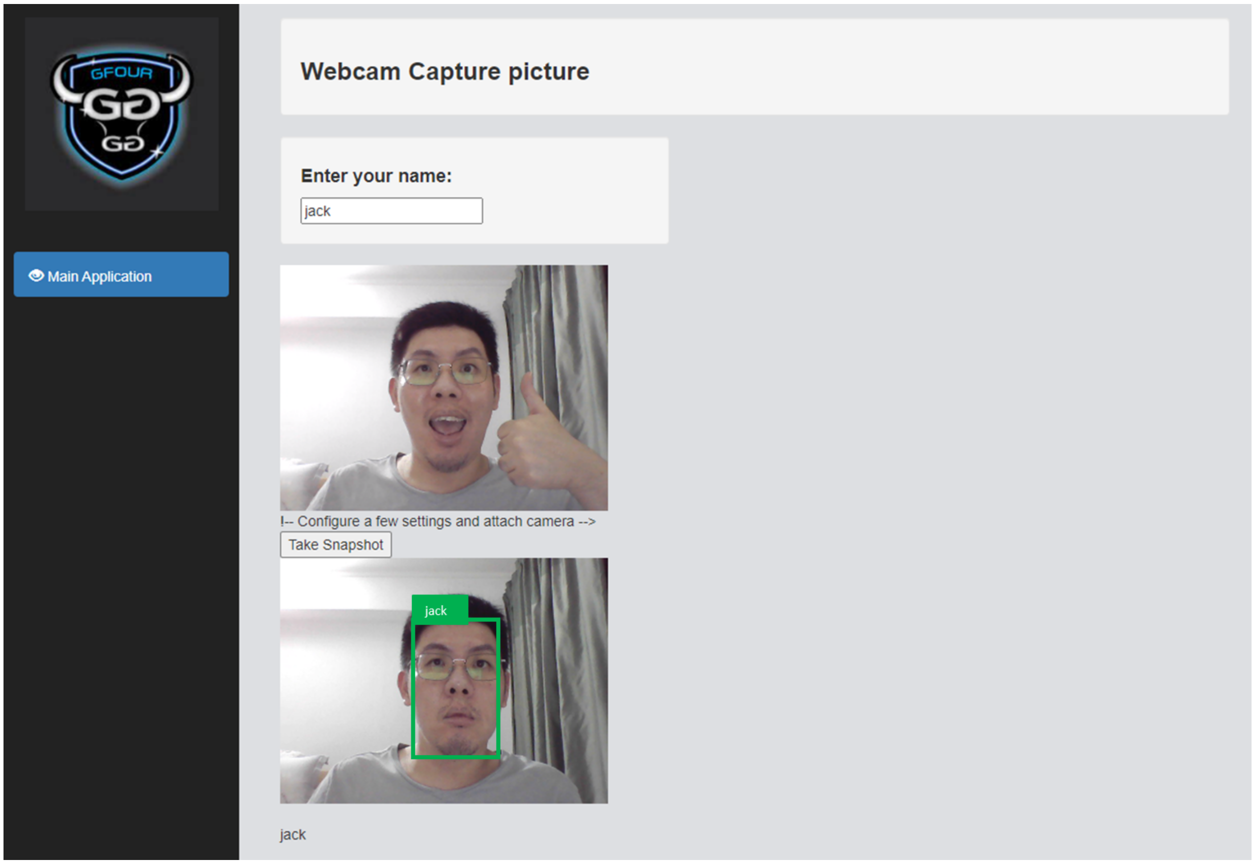


Webcam Capture picture

Enter your name: zombie

!-- Configure a few settings and attach camera -->

zombie



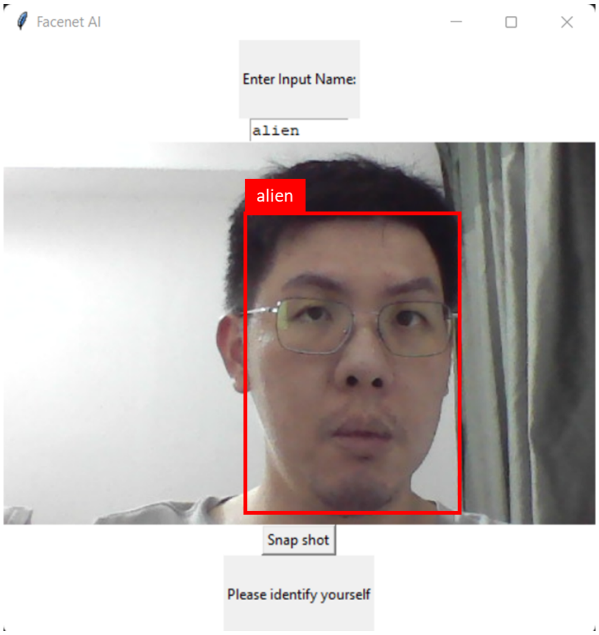
Webcam Capture picture

Enter your name: jack

!-- Configure a few settings and attach camera -->

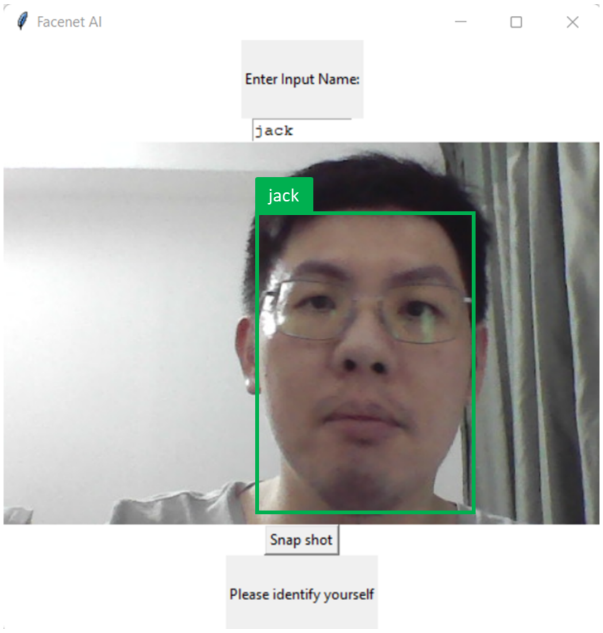
jack

Mobile application version



Enter Input Name: alien

Please identify yourself



Enter Input Name: jack

Please identify yourself

6. Members Contributions

Name	Admin No.	Contributions
Yap Jing Yang	21B583W	● Team Leader (Project planning and coordination works) ● Did experiment runs on ResNet50 with Triplett Loss Architecture ● Did Jupyter notebook for demo of project (backup)
Ng Wei Xiang	21B576M	● Creator of final model that was used in the project ● Deployment of project ● Did experiment runs on EfficientFaceNet
Looi Ying Choi	21A458H	● Did experiment runs on Siamese Network with Triplet Loss ● Did experiment runs on HTML UI for deployment part
George Low	21B573H	● Did experiment runs on MobileNet with Triplet Loss ● Did experiment runs on HTML UI for deployment part

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
6) MobileFaceNets: Efficient CNNs for Accurate RealTime Face Verification on Mobile Devices
   https://arxiv.org/ftp/arxiv/papers/1804/1804.07573.pdf

Annex B. Looi's Model Performances

Base model: 2 conv layers

run	Batch size	Image size	model	Number of Epoch	F1-score for test dataset
1	128	160X160	Two conv layers	13	0.1

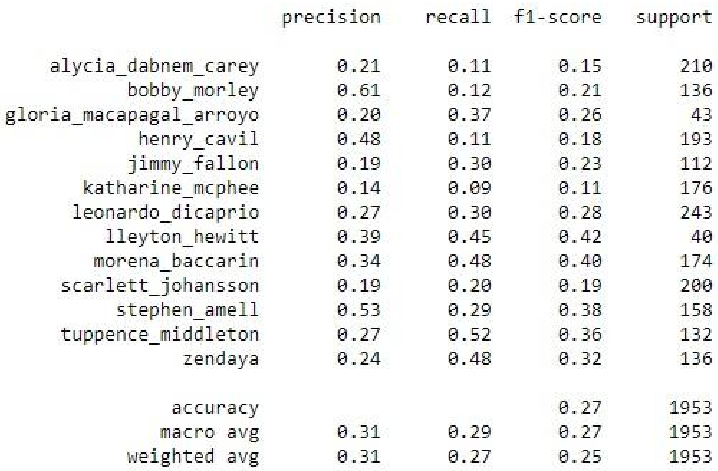
Base model: Mobilenet

run	Batch size	Image size	model	Number of Epoch	F1-score for test dataset
1	128	160X160	Set 54 onward trainable to true (~0.6% conv layers)	27	0.24

Base model: Resnet50

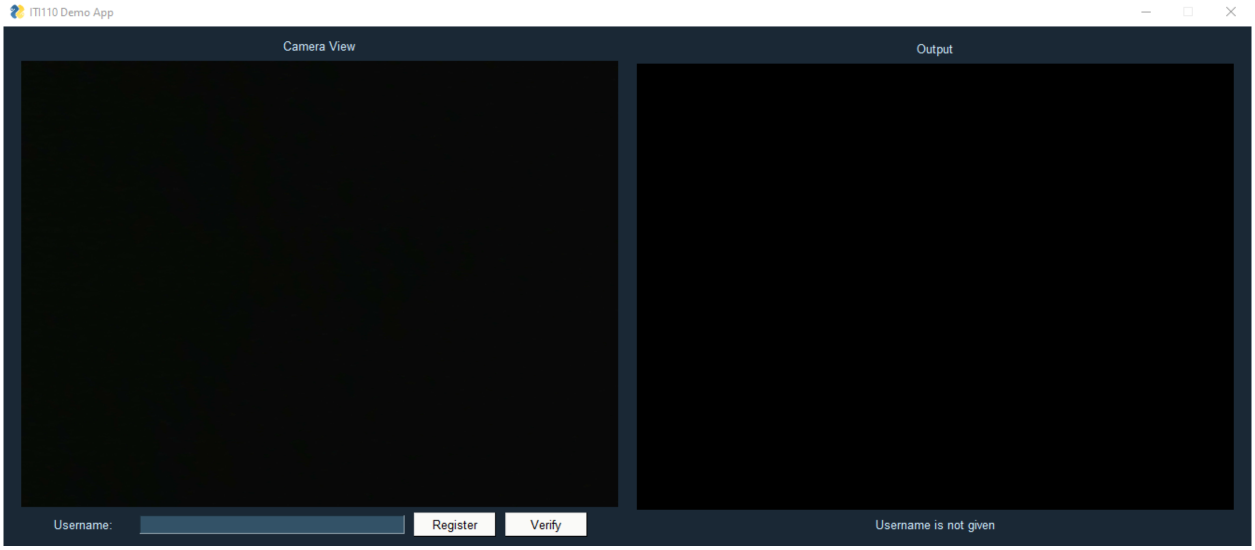
run	Batch size	Image size	model	Number of Epoch	F1-score for test dataset
1	128	160X160	Set Conv5, last layer trainable to true	22	0.27
2	128	100x100	Set Con4 layer onward trainable to true	26	0.27
3	38	100x100	Set Con3 layer onward trainable to true	55	0.18

Run 2 - f1 score

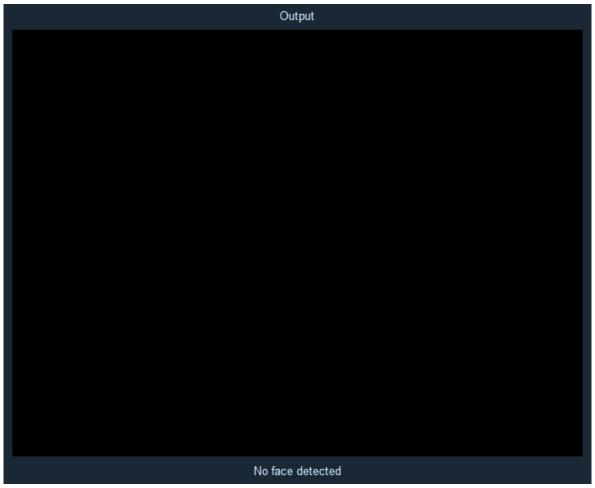


```
                       precision    recall  f1-score   support

     alycia_dabnem_carey       0.21      0.11      0.15       210
            bobby_morley       0.61      0.12      0.21       136
  gloria_macapagal_arroyo       0.20      0.37      0.26        43
             henry_cavil       0.48      0.11      0.18       193
            jimmy_fallon       0.19      0.30      0.23       112
        katharine_mcphee       0.14      0.09      0.11       176
       leonardo_dicaprio       0.27      0.30      0.28       243
          lleyton_hewitt       0.39      0.45      0.42        40
         morena_baccarin       0.34      0.48      0.40       174
      scarlett_johansson       0.19      0.20      0.19       200
           stephen_amell       0.53      0.29      0.38       158
       tuppence_middleton       0.27      0.52      0.36       132
                 zendaya       0.24      0.48      0.32       136

                accuracy                           0.27      1953
               macro avg       0.31      0.29      0.27      1953
            weighted avg       0.31      0.27      0.25      1953
```

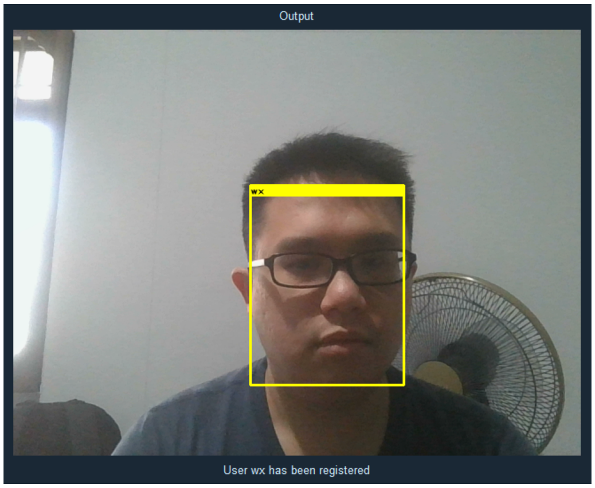
Annex C. Deployment Screenshots



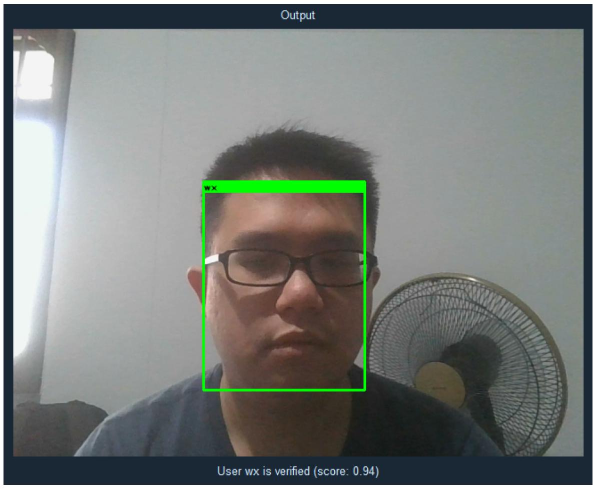
Username is not given



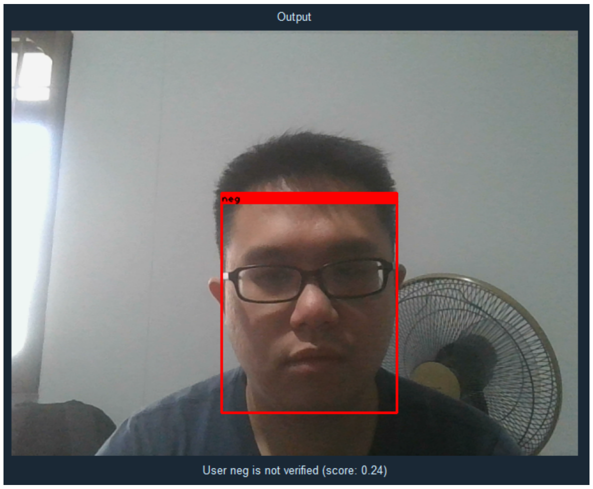
No face detected



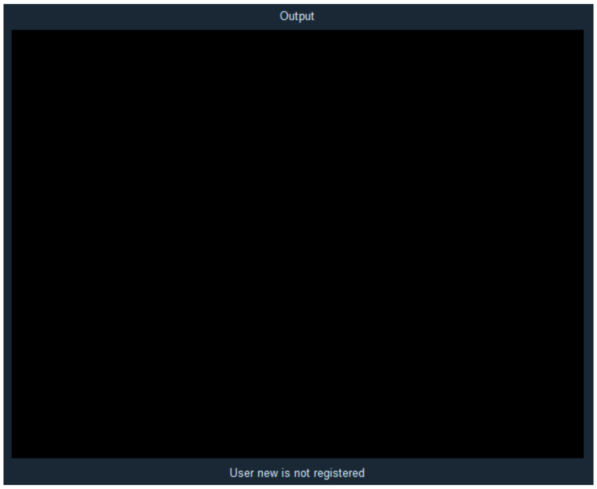
User wx has been registered



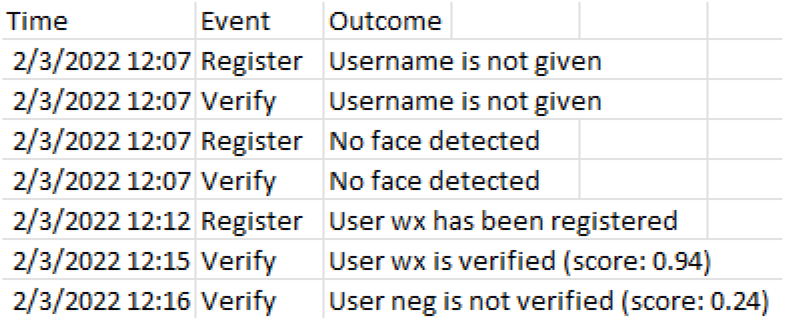
User wx is verified (score: 0.94)



User neg is not verified (score: 0.24)



User new is not registered



Time	Event	Outcome
2/3/2022 12:07	Register	Username is not given
2/3/2022 12:07	Verify	Username is not given
2/3/2022 12:07	Register	No face detected
2/3/2022 12:07	Verify	No face detected
2/3/2022 12:12	Register	User wx has been registered
2/3/2022 12:15	Verify	User wx is verified (score: 0.94)
2/3/2022 12:16	Verify	User neg is not verified (score: 0.24)