UNIVERSITI MALAYA

WID3008 - Image Processing

Project Title : Facial Recognition with Medical Masks for Identity Verification

Name	Matric no.
Tan Mei Yih	WID180044(17059516/1)

Introduction

Since the 1960s, face recognition has occurred in various forms. It has become a part of our daily lives as a result of recent technological advancements. For example:

- **Facebook** introduced **DeepFace** with an accuracy rate of 97.25% , which was just 0.28 percent lower than that of a person in 2014.
- **Google** moved a step better with **FaceNet** which achieved a higher accuracy of 99.63% on the commonly used Labeled Faces in the Wild (LFW) dataset in 2015.
- **Apple** released the iPhone X which has the **Face ID feature** in 2017.

Present Challenge

Face recognition trained to usual face images has proven to give good accuracy. However, in the recent ongoing outbreak of Covid19, wearing face masks has become mandatory. With the majority of people using face masks, the face recognition system fails to perform. This caused problems for the current identification systems. For instance, FaceID, hadtrouble recognizing people wearing masks. Thus, people wearing masks are having trouble unlocking their iPhones.

In 2020, Chinese researchers suggested Face ID could be unofficially reset to recognize masked faces by folding a mask in half, holding it over the lower part of the face and rescanning.

Indeed, based on the latest update on April 27, 2021, it is shown that this method is implemented to unlock iPhone with Apple Watch when Face ID detects the owner is wearing a mask.

Figure 1: Face ID with a mask, including new Apple Watch Unlock feature[1]

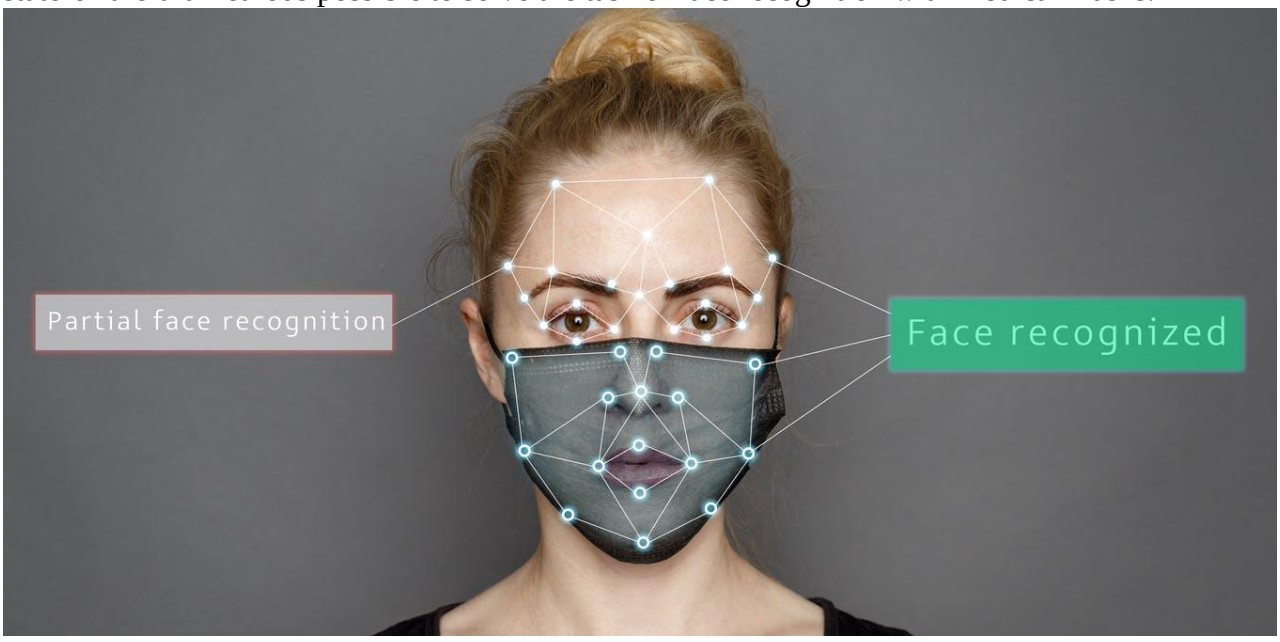
Another solution was given by Maskalike[2]. They designed face masks that resemble our faces. But the effectiveness of this particulate-blocking polyester and cotton made masks on

providing protection against the coronavirus disease (COVID-19) compared to the medical mask is under debate.

Figure 2 : Face masks that resemble our faces, Maskalike[2]

Therefore, there is still a great room of improvement for face recognition with mask and the task in this project is to create a system that can recognize faces with medical masks (to match a face image without any occlusion with a face image with a mask).

This project will show and build a system implementing one of the most modern state-of-the-art methods possible to solve the task of face recognition with medical masks.



Objectives

1. To recognize the face location of a masked individual.
2. To evaluate the accuracy of the facial recognition net.

Dataset source

1. Vgg_face2 dataset[3]

VGGFace2[4] is a large-scale face recognition dataset. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession.

- 9,000 +identities
  - VGGFace2 contains images from identities spanning a wide range of different ethnicities, accents, professions and ages.
- 3.3 million +faces
  - All face images are captured "in the wild", with pose and emotion variations and different lighting and occlusion conditions.
- 362 ~per-subject samples
  - Face distribution for different identities is varied, from 87 to 843, with an average of 362 images for each subject.

2. https://github.com/aqeelanwar/MaskTheFace/tree/master/datasets[4]

Literature review

Face representation using Deep Convolutional Neural Network (DCNNs) embedding is the method of choice for face recognition. DCNNs map the face image, typically after a pose normalization step [5], into a feature that has small intra-class and large inter-class distance.

There are two main lines of research to train DCNNs for face recognition:

- Train a **multi-class classifier** which can separate different identities in the training set, such by using a **softmax classifier**
- Learn directly an **embedding**, such as the **triplet loss**.

Based on the large-scale training data and the elaborate DCNNs architectures, both the softmax-loss-based methods and the triplet-loss-based methods can obtain excellent performance on face recognition. However, both the softmax loss and the triplet loss have some **drawbacks**.

For the softmax loss: (1) the **size of the linear transformation matrix** $W \in R^{d \times n}$ **increases** linearly with the identities number n; (2) the learned features are separable for the closed-set classification problem but **not discriminative enough** for the open-set face recognition problem.

For the triplet loss: (1) there is a **combinatorial explosion in the number of face triplets** especially for large-scale datasets, leading to a significant **increase in the number of iteration steps**; (2) **semi-hard sample mining** is a quite difficult problem for effective model training.

Below are some of the related work on face recognition discuss on the gap, advantages or disadvantages on the loss, gap and aim of work proposed in the methods,

Ref	Aim of work	Result/ Gap
Wen et al.(2016)[6]	Propose the centre loss, the Euclidean distance between each feature vector and its class centre, to obtain intra-class compactness while the interclass dispersion is guaranteed by the joint penalisation of the softmax loss.	Updating the actual centres during training is extremely difficult as the number of face classes available for training has recently dramatically increased.
W. Liun et al. (2016)[7]	Proposed a multiplicative angular margin penalty to enforce extra intra-class compactness and inter-class discrepancy simultaneously.	Leading to a better discriminative power of the trained model.
W. Liu et al. (2017)[8]	• Introduced the important idea of angular margin, their loss function required a series of approximations in order to be computed, which resulted in an unstable training of the network. • Proposed a hybrid loss function which includes the standard softmax loss to stabilise training.	The softmax loss dominates the training process, because the integer-based multiplicative angular margin makes the target logit curve very precipitous and thus hinders convergence
F. Wang et al. (2017)[9]	Directly adds cosine margin penalty to the target logit.	Obtains better performance compared to SphereFace but admits much easier implementation and relieves the need for joint supervision from the softmax loss.
Yang et al.(2020)[10]	Proposed a system that uses features from a pre-trained facial landmark localization network to enhance face recognition accuracy.	Show improves recognition on the most difficult face recognition datasets, setting a new state-of-the-art on IJB-B, IJB-C and MegaFace datasets.

Deng et al. (2019) [11]	Proposed an Additive Angular Margin Loss to **obtain highly discriminative features for face recognition**. This proposed ArcFace has a clear geometric interpretation due to the exact correspondence to the geodesic distance on the hypersphere.	In performance, ArcFace achieves sota results on the MegaFace Challenge, which is the largest public face benchmark with one million faces for recognition. Advantages of the proposed ArcFace can be summarised as follows: ● **Engaging.** ArcFace directly optimises the geodesic distance margin by virtue of the exact correspondence between the angle and arc in the normalised hypersphere. ● **Effective**. ArcFace achieves sota performance on ten face recognition benchmarks. ● **Easy.** ArcFace only needs several lines of code as given in Algorithm 1 and is extremely easy to implement in MxNet, Pytorch and Tensorflow. ● **Efficient.** ArcFace only adds negligible computational
		complexity during training.

Table 1: Literature review

Analysis of system requirements

Tools suitable for this system development include:

Aspect	**Item**	**Usage**
Programming Language	● Python	● Model training, evaluating
API / Python Libraries	● Numpy	● Mathematical operation
	● PyTorch	● Training & Transfer learning ● Develop AI model which : 1. Makes augmentations that transform the initial training dataset(persons without mask) into persons wearing medical masks 2. Recognize the identity of a masked individual. 3. Achieve great accuracy on facial recognition.

	• Pandas	• Data cleaning
	• OpenCV • Pillow	• Image Processing
	• Matplotlib	• Inline plots for data visualisation
	• SciPy	• Compute triangulations
	• face_alignment	• Simplified Face Recognition • Detect facial landmarks
Software	• Google Colab	• Standardize training environment
	• Git	• Version control

Table 2: Analysis of system requirement

System design

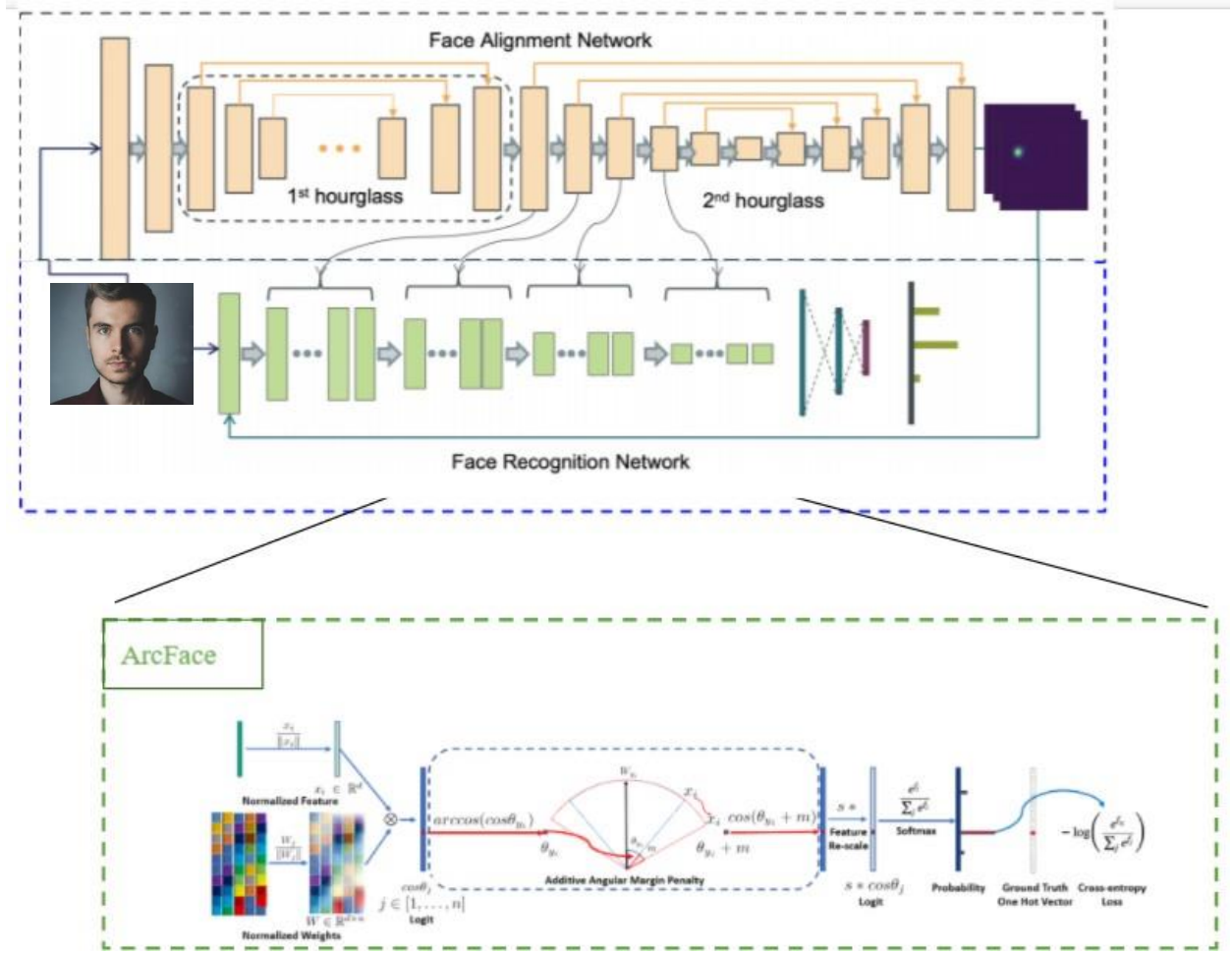


Figure 3: System architecture

Flowchart

Training Flowchart

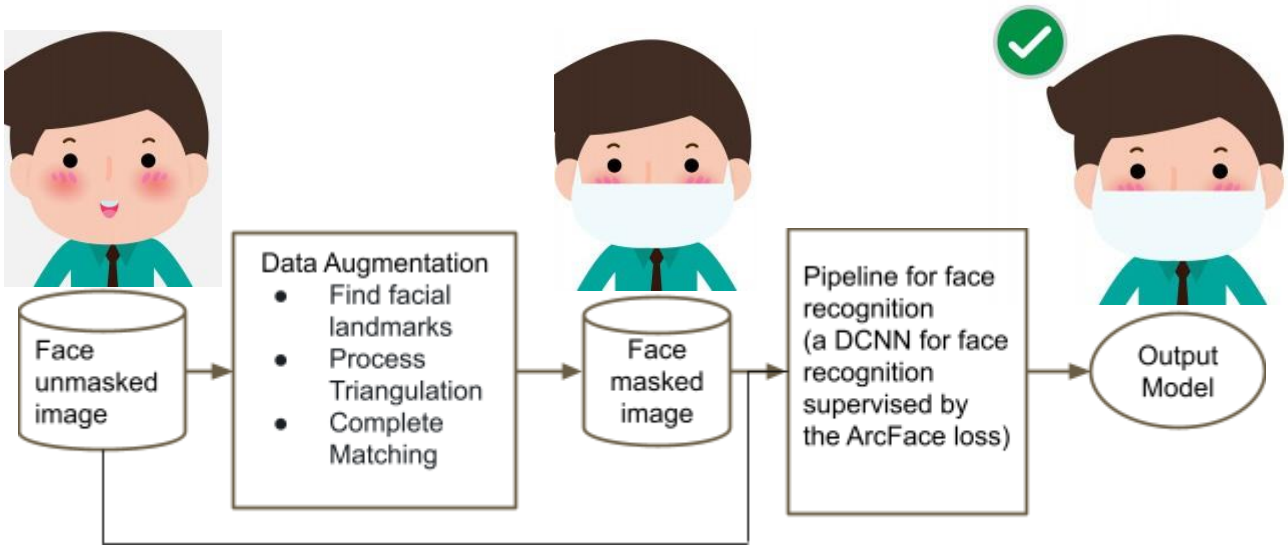
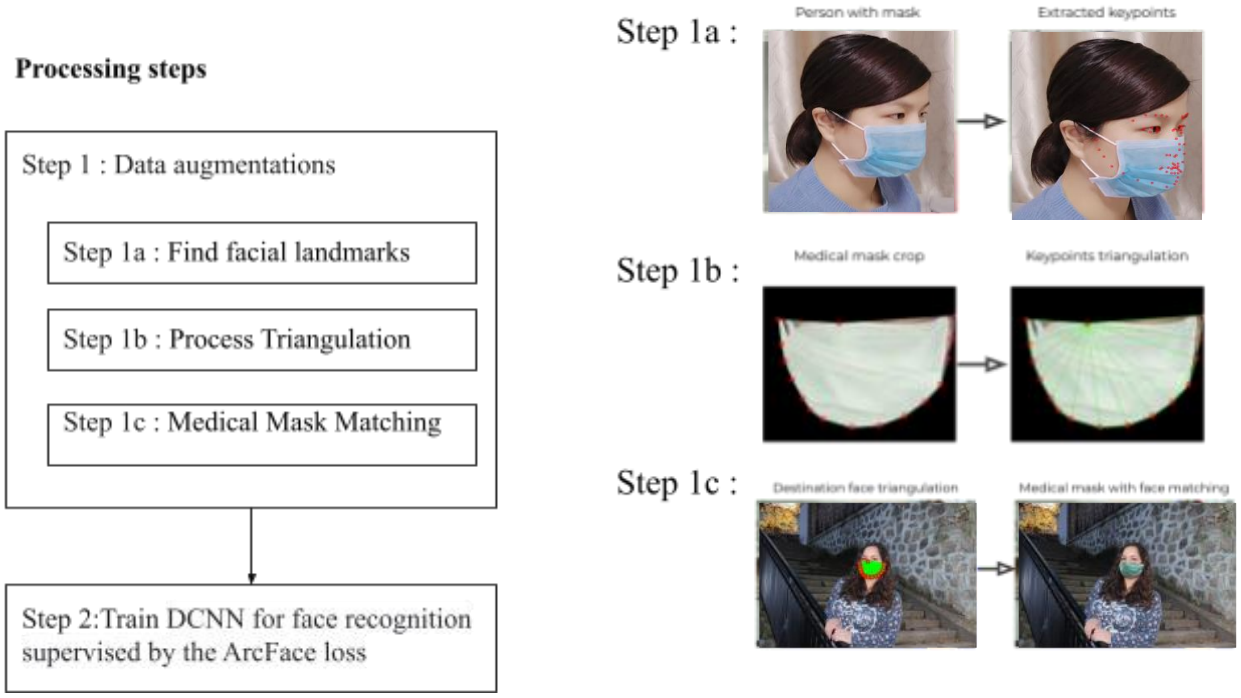


Figure 4: Training flowchart

Face unmasked images will pass through data augmentation process to convert into face masked images.Then the face unmasked images and face masked images will be feed into pipeline of face recognition model to learn to recognise the face,which after finish train will tell whether the face masked images can be match with or recognise as the face unmasked images.

Processing steps

The system flow of this face recognition system is as below :



Processing steps in explanation

Step 1 : Data augmentations

Transform the training dataset from image of person face without a medical mask into image of person face with a medical mask.

Aim:

- Increases the amount of training data(mask images) using information available from original images to capture as much data variation as possible.
- Creates more data(mask images) to get better generalization in the model.

Figure 3 : Process of augmentation (without and with augmented medical mask)

Augmentation task in 3 steps:

Step 1a : Find facial landmarks

I will use face-alignment[12] python library in order to extract keypoints(find facial landmarks) of

the initial face(person with mask).
Figure 4 : Process of facial keypoints extraction

Step 1b : Process Triangulation

In this cropping and triangulation process, about 250 masks will be extracted from the handcrafted dataset with the medical masks, which will be then use to match to other persons(into image of a person's face without a medical mask).

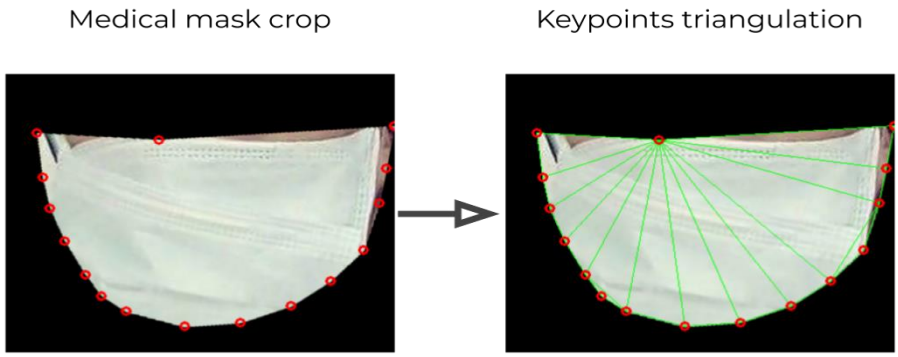


Figure 5: Process of mask cropping and triangulation

Step 1c : Medical Mask Matching

The last step of the augmentation is to **extract keypoints**, **triangulate the face regions** (lower face regions) that are needed and then **match the random extracted mask** from the previous step, Step1b **to our destination face**.

Another advantage of this step is that it can help to **recognize faces in a variety of positions, including face rotation.** The database of medical masks will be stored in JSON. It includes calculated parameters of rotation. This enables matching of images with face rotation with masks that come under a certain interval of rotation for a given face.

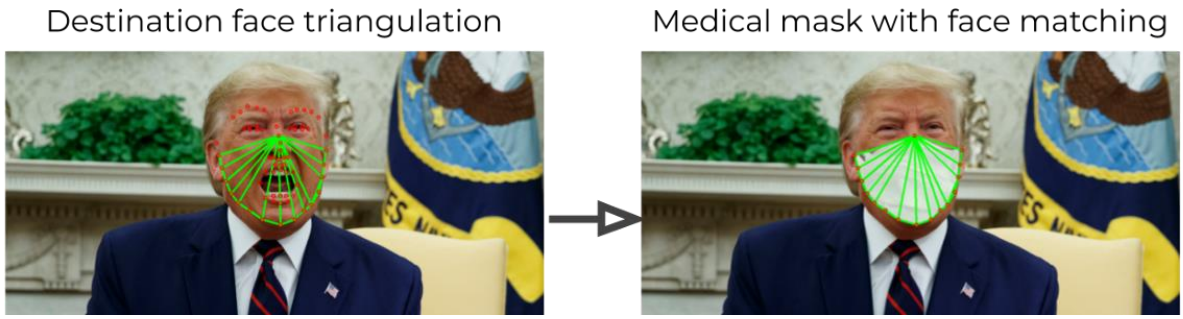


Figure 6 : Complete Matching of Medical Mask

With this 3 sub-steps, we successfully transform the training dataset from image of person face without a medical mask into an image of person face with a medical mask.

Step 2:Train ArcFace model on Vgg_face2_dataset[3]

The preprocessed part (Step1 :Augmentation process) of the Vgg_face2 dataset[3] with medical masks is available at this Google Drive link.

Experiment

Concept (DCNNs training for face recognition supervised by the ArcFace loss):

ArcFace: Additive Angular Margin Loss for Deep Face Recognition[11], ArcFace loss is implemented in the training of this face recognition system.

One of the main challenges in feature learning using Deep Convolutional Neural Networks (DCNNs) for large scale face recognition is the design of **appropriate loss functions** that enhance discriminative power.

ArcFace paper addresses this problem by proposing an Additive Angular Margin Loss (ArcFace) to **obtain highly discriminative features for face recognition**. This proposed ArcFace has a clear geometric interpretation due to the exact correspondence to the geodesic distance on the hypersphere. In performance, ArcFace achieves state-of-the-art results on the MegaFace Challenge, which is the largest public face benchmark with one million faces for recognition.

Experimental Settings

For data prepossessing(data augmentation), normalised face crops (112 × 112) are generated by utilising facial points.

For the embedding network, the widely used CNN architecture, ResNet18 is employed. After the last convolutional layer, the BN -Dropout -FC-BN structure is explored to get the final 512-D embedding feature.

for DCNNs training :

In this work, the experimental settings solely follow the settings in ArcFace work, where the feature scale s is set to 64 and the angular margin m of ArcFace is set to 0.5.In this work the batch size is set to 100 and model is trained on Google Colab GPUs. The training process is finished at 3000 iterations.

During testing, only the feature embedding network is kept without the fully connected layer (ResNet18) and extract the 512- D features for each normalised face.

This is a process of training a DCNN for face recognition supervised by the ArcFace loss:

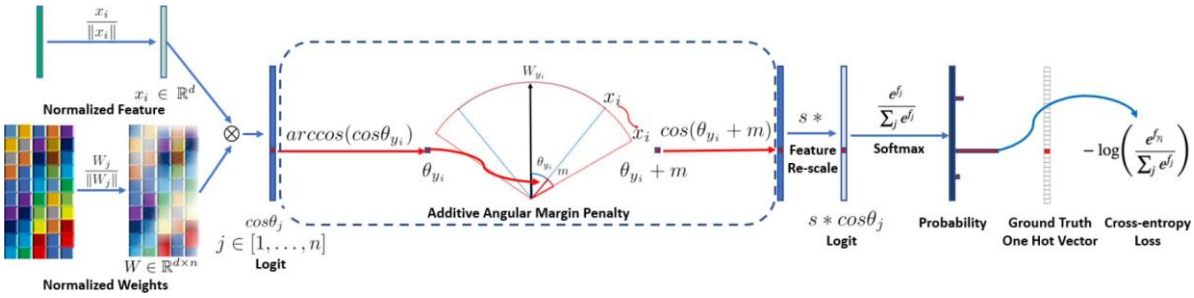


Figure 7 : Training a DCNNs for face recognition supervised by the ArcFace loss. Based on the feature x_i and weight W normalisation, we get the cos θj (logit) for each class as $W^T_j x_i$. We calculate the arccosθ_{yi} and get the angle between the feature xi and the ground truth weight W_{yi} . In fact, W_j provides a kind of centre for each class. Then, we add an angular margin penalty m on the target (ground truth) angle θ_{yi} . After that, we calculate cos(θ_{yi} + m) and multiply all logits by the feature scale s. The logits then go through the softmax function and contribute to the cross entropy loss.

The algorithm and explanation in this session is reference from the original paper. The deeper theoretical part of this ArcFace loss, can be found at the original research paper[11] and this medium article[13]. All experiments in this ArcFace paper are implemented by MXNet[14],Thus, The algorithm shown here is experiment of ArcFace on MxNet .

Algorithm : The Pseudocode of ArcFace on MxNet

Input: Feature Scale s, Margin Parameter m in L_3, Class Number *n*, Ground-Truth ID *gt*.

1. x = mx.symbol.L2Normalization (x, mode = 'instance')
2. W = mx.symbol.L2Normalization (W, mode = 'instance')
3. fc7 = mx.sym.FullyConnected (data = x, weight = W, no bias = True, num hidden = n)
4. original target logit = mx.sym.pick (fc7, gt, axis = 1)

5. theta = mx.sym.arccos (original target logit)
6. marginal target logit = mx.sym.cos (theta + m)
7. one hot = mx.sym.one hot (gt, depth = n, on value = 1.0, off value = 0.0)
8. fc7 = fc7 + mx.sym.broadcast mul (one hot, mx.sym.expand dims (marginal target logit - original target logit, 1))
9. fc7 = fc7 * s

Output: Class-wise affinity score $fc7$

There are 7 losses introduced in ArcFace to **convert the loss from Softmax to ArcFace** (as the most widely used classification loss function is Softmax loss).Here in this report, I only show the two important loss to aid the explanation,The further explanation of this ArcFace loss, can be found at the original research paper[11].

The L_3 [7] in input: $$L_3 = -\frac{1}{m}\sum_{i=1}^{m}\log\frac{e^{\|x_i\|\cos(m\theta_{y_i})}}{e^{\|x_i\|\cos(m\theta_{y_i})}+\sum_{j=1,j\neq y_i}^{n}e^{\|x_i\|\cos\theta_j}}$$

ArcFace is defined as: $$L_7 = -\frac{1}{m}\sum_{i=1}^{m}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$

The **pipeline** of this work implementing ArcFace loss :

1. Build face-alignment for face recognition

Detect facial landmarks using face-alignment network[12], to detect points(find facial landmarks) in 2D coordinates of the initial face .

The facial landmarks provide **pose, expression and shape information**.When matching, a mask face to a frontal unmask one, knowledge of these landmarks is useful for establishing correspondence which can help improve recognition.Facial landmarks also used for **face cropping** in order to remove scale, rotation and translation variations.

Figure 8 : Detect 2D facial landmarks in pictures

Build using a state-of-the-art deep learning based face alignment method called FAN-Face: a Simple Orthogonal Improvement to Deep Face Recognition[10].

2. Import face mask SDK

Here

- end2end_mask_generation (for end to end mask generation of mask on unmask image)

- PipelineFacesDatasetGenerator (generator for train dataset)
- Backbone, ArcFaceLayer, FaceRecognitionModel, resnet18 (backbone for the face mask recognition network)
- default_acc_function, test_embedding_net (for training)

are imported.

3. **Generate masks database**
4. **Prepare training and test dataset**
5. **Initialize constants, base variables for training**

Initialise constants	Initialise base variables for training
• batch_size = 32 • n_jobs = 4 • epochs = 100 • image_shape = (112, 112) • embedding_size = 256 • device = 'cuda:0'	• generator_train_dataset to generate train dataset • train_loader for loading of the generated train dataset • model is **FaceRecognitionModel** with pretrained resnet18 as Backbone and ArcFace Layer as head • loss_function use is torch.nn.**CrossEntropyLoss()** • Optimizer use is torch.optim.**SGD** with default model parameter and **learning rate of 0.00001**

6. **Run test for embedding net**

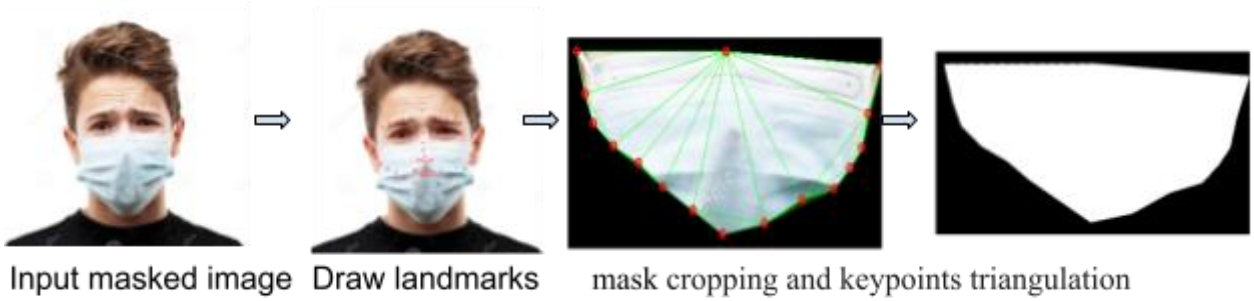
Start accuracy rate:

```
  0%|          | 0/36 [00:00<?, ?it/s]Loading and process test dataset:
100%|██████████| 36/36 [00:41<00:00,  1.16s/it]
(11482, 256)
Evaluate accuracy rate...
Start accuracy rate = 0.18307
```

7. **Perform training process (Training of DCNNs supervised by ArcFace Loss)**

In the training process, the Face recognition net creates unified embeddings of the faces and then compares the faces in the embedding space to carry out decision making. During the training phase, multiple image pairs are provided to the network. The network maps these image pairs to embedding vectors and calculates **CrossEntropyLoss.**

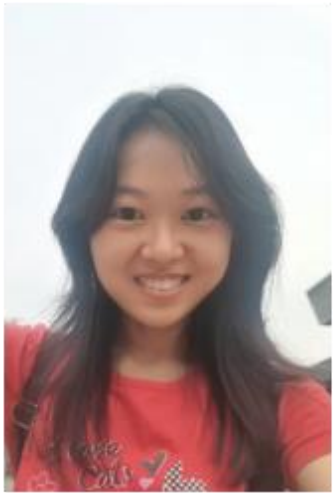
8. **Plot the results**
9. **Testing**



<u>Results</u>

Personal passport photo of Tutorial week1 as one of the experiment input.

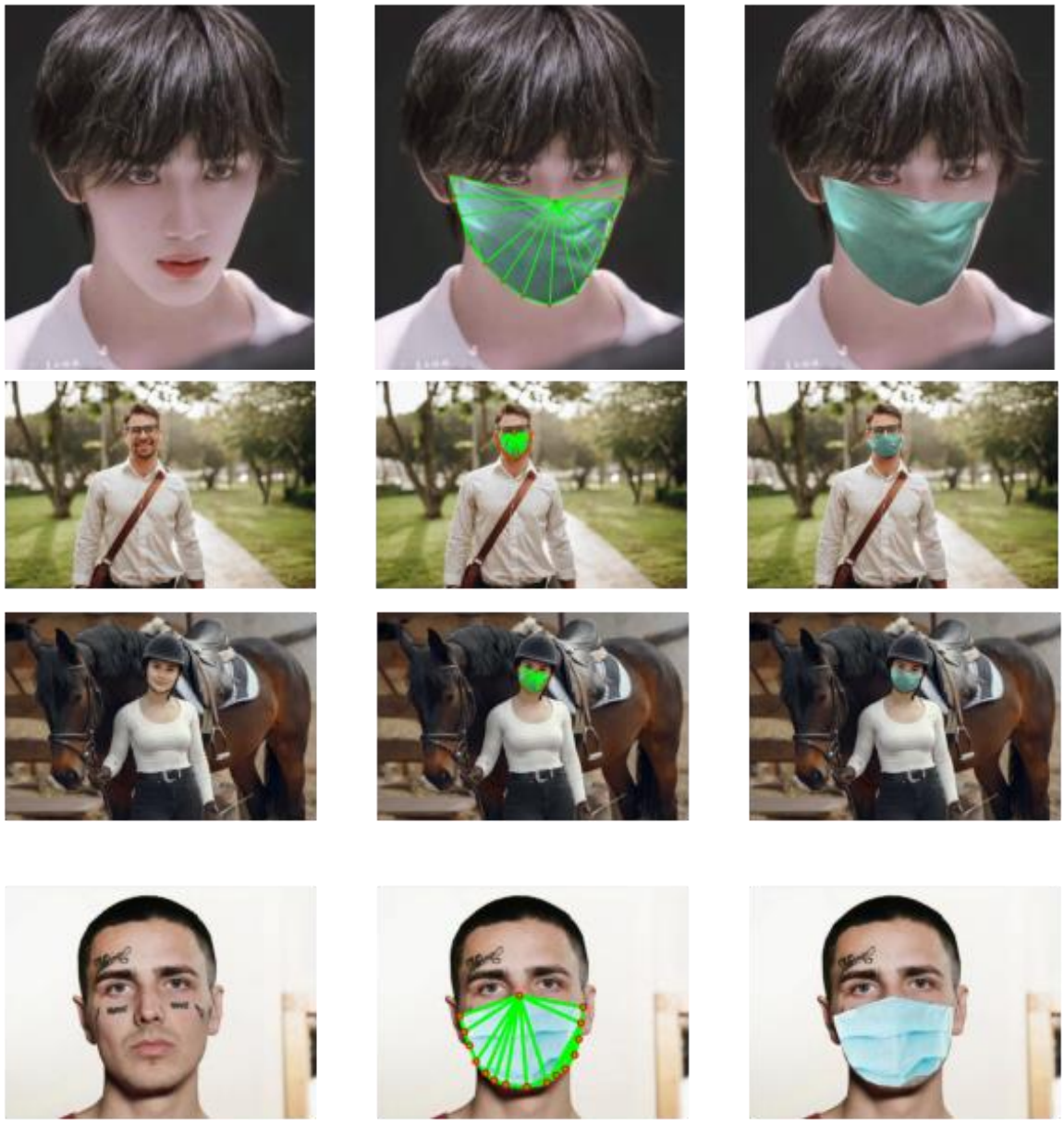


Results on some other images :





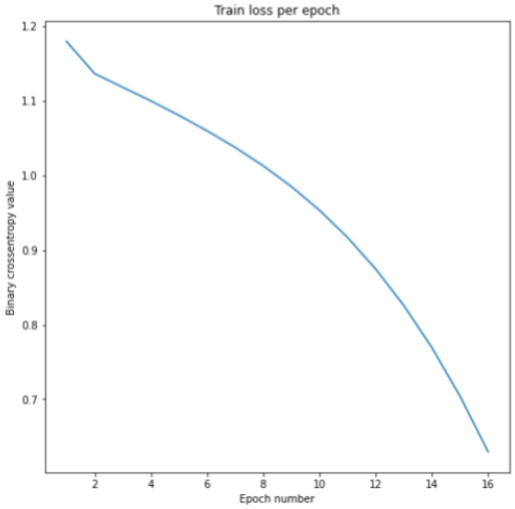
Results analysis

Metric use to analyze the performance of the trained networks : $$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

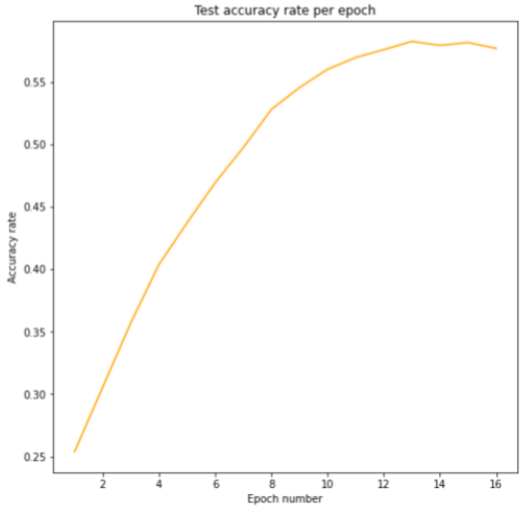
• Accuracy (%): The maximum accuracy of the network in terms of identifying the test input image pairs as -ve or +ve.

This work able to achieve **58 %** accuracy with custom metrics on the test dataset.

Train loss



Test accuracy



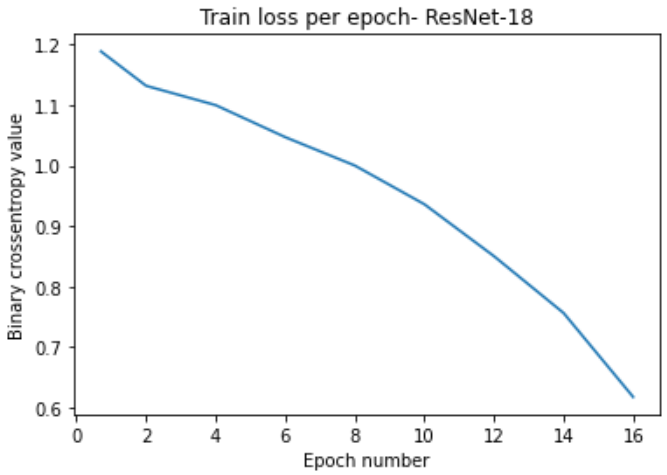
The ability to show impressive results for such limited training time proves that pipeline is able to solve face recognition with medical masks.

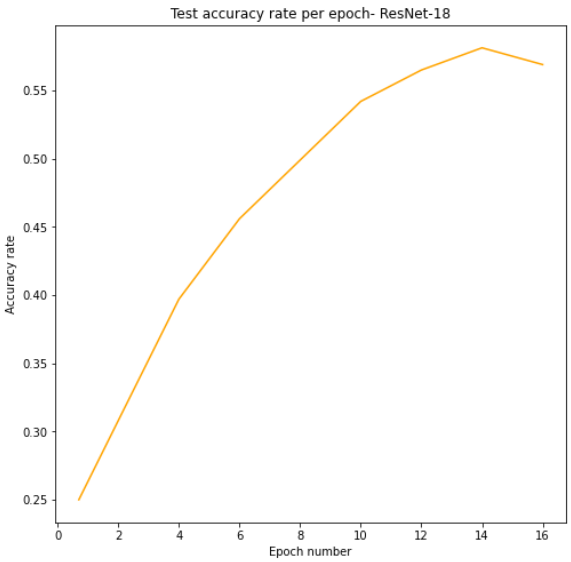
Comparison/Ablation study on different ResNet

Verification accuracy (%) under different embedding network settings

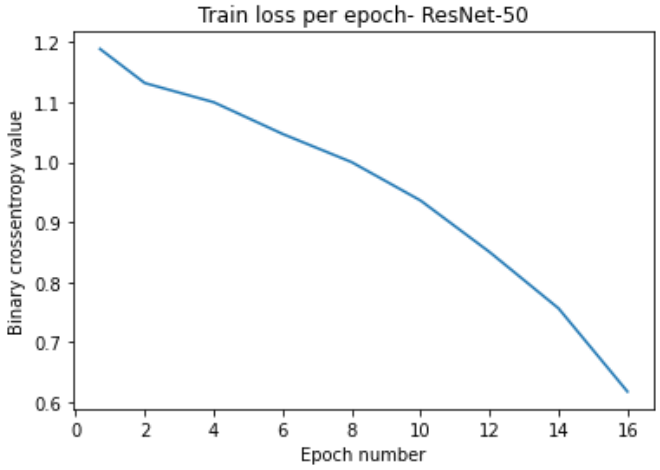
Networks	Vgg_face2 dataset
ResNet18	58%
ResNet50	58%
ResNet101	57.2%

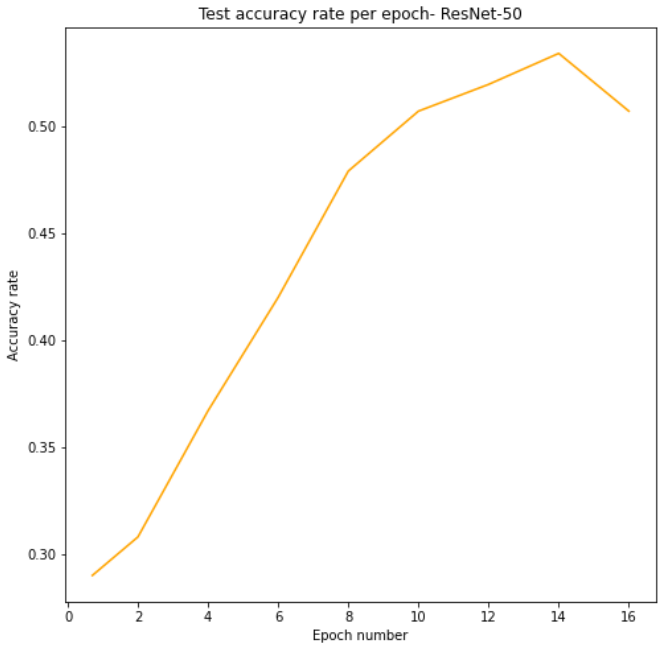
ResNet18 training losses and testing accuracy on Vgg_face2 dataset :



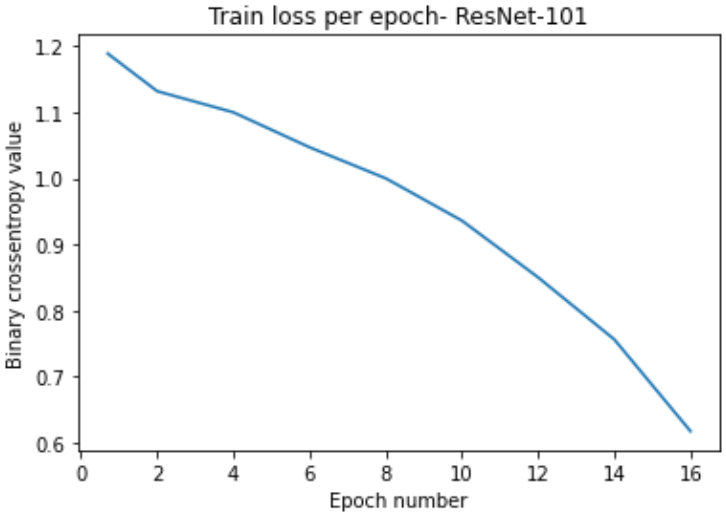


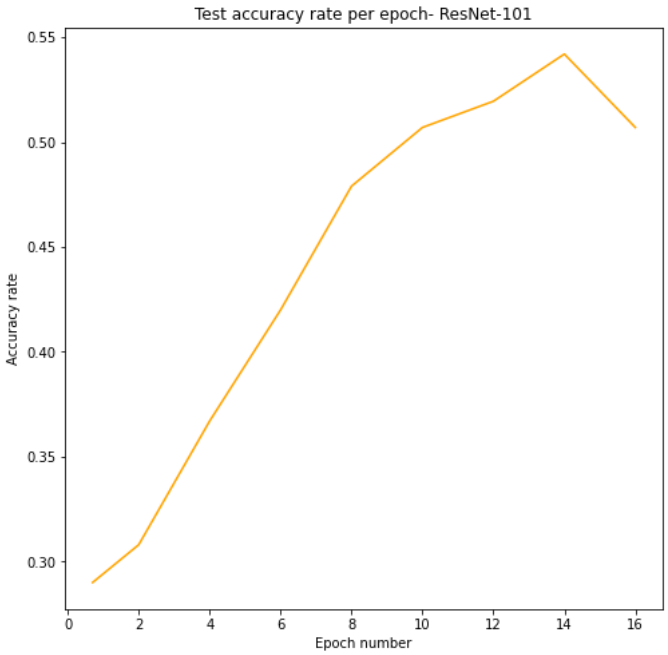
ResNet50 training losses and testing accuracy on Vgg_face2 dataset:





ResNet100 training losses and testing accuracy on Vgg_face2 dataset:





Verification accuracy (%) under different dataset.

Networks	Vgg_face2 dataset	LFW(Labelled Faces in the Wild)
ResNet18	**58%**	**59.38%**
ResNet50	58%	54.01%
ResNet101	57.2%	56.53%

From the experiments done, ResNet18 able to achieve an accuracy of 58% , ResNet50 able to achieve an

accuracy of 58% and ResNet101 able to achieve an accuracy of 57.2% when tested on Vgg_face2 dataset. On the other hand, when tested on LFW(Labelled Faces in the Wild) dataset , ResNet18 can achieve an accuracy of 59.38% , ResNet50 able to achieve an accuracy of 54.01% and ResNet101 able to achieve an accuracy of 56.53%.

It can be notice that ResNet18 achieve the best inference accuracy for both Vgg_face2 dataset and LFW(Labelled Faces in the Wild) dataset. Thus, I employ ResNet18 in the embedding network of this project.

Conclusion

Achievements

Handle situation with the face rotation

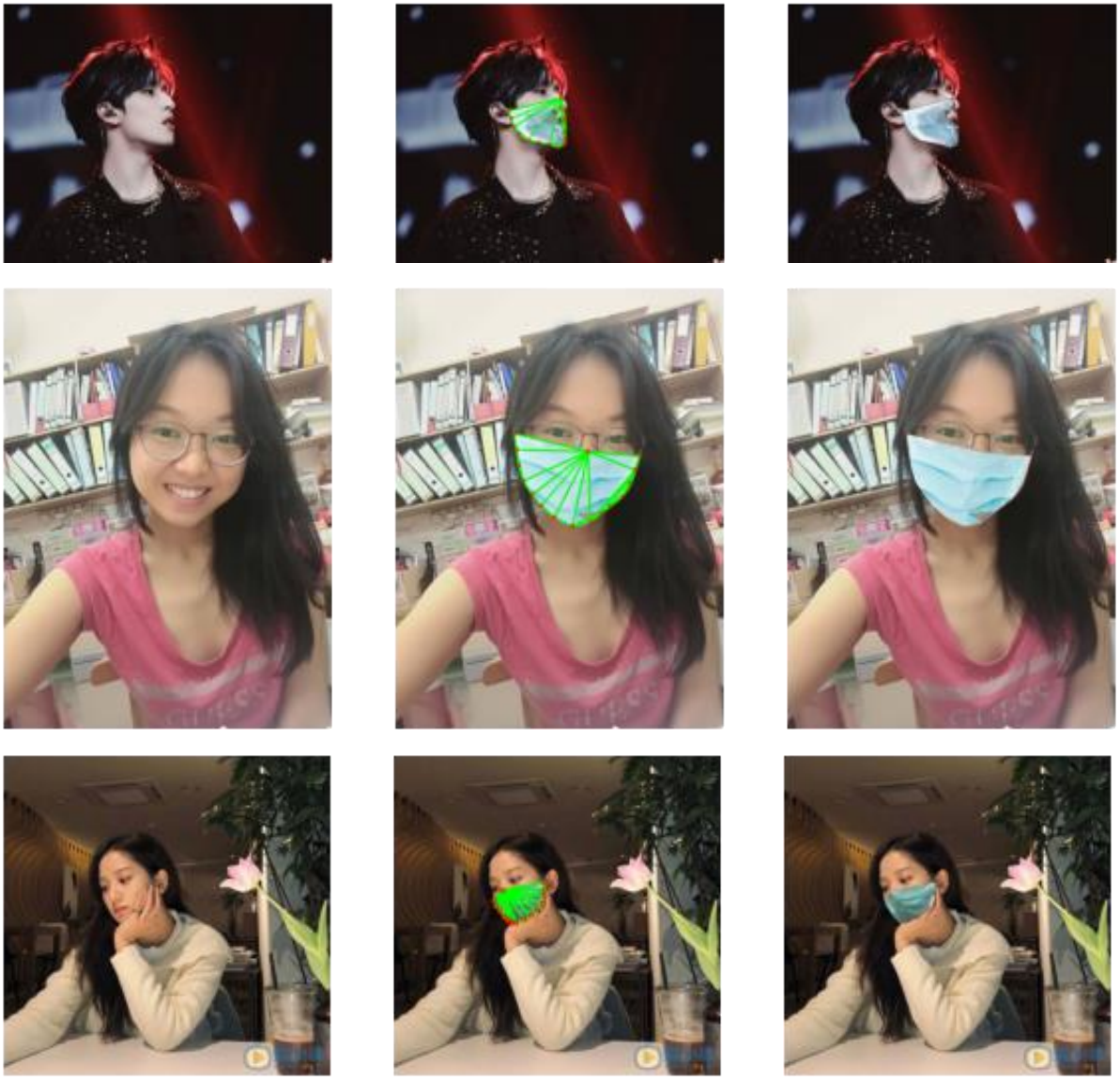
This work can handle the situation with face rotation. The medical masks database is stored in json with the calculated parameter of rotation, this allows the matching of images with face rotation with masks that are falling in concrete interval of rotation for a given face in that image.

Figure 9 : VGGFace dataset samples augmented with medical masks

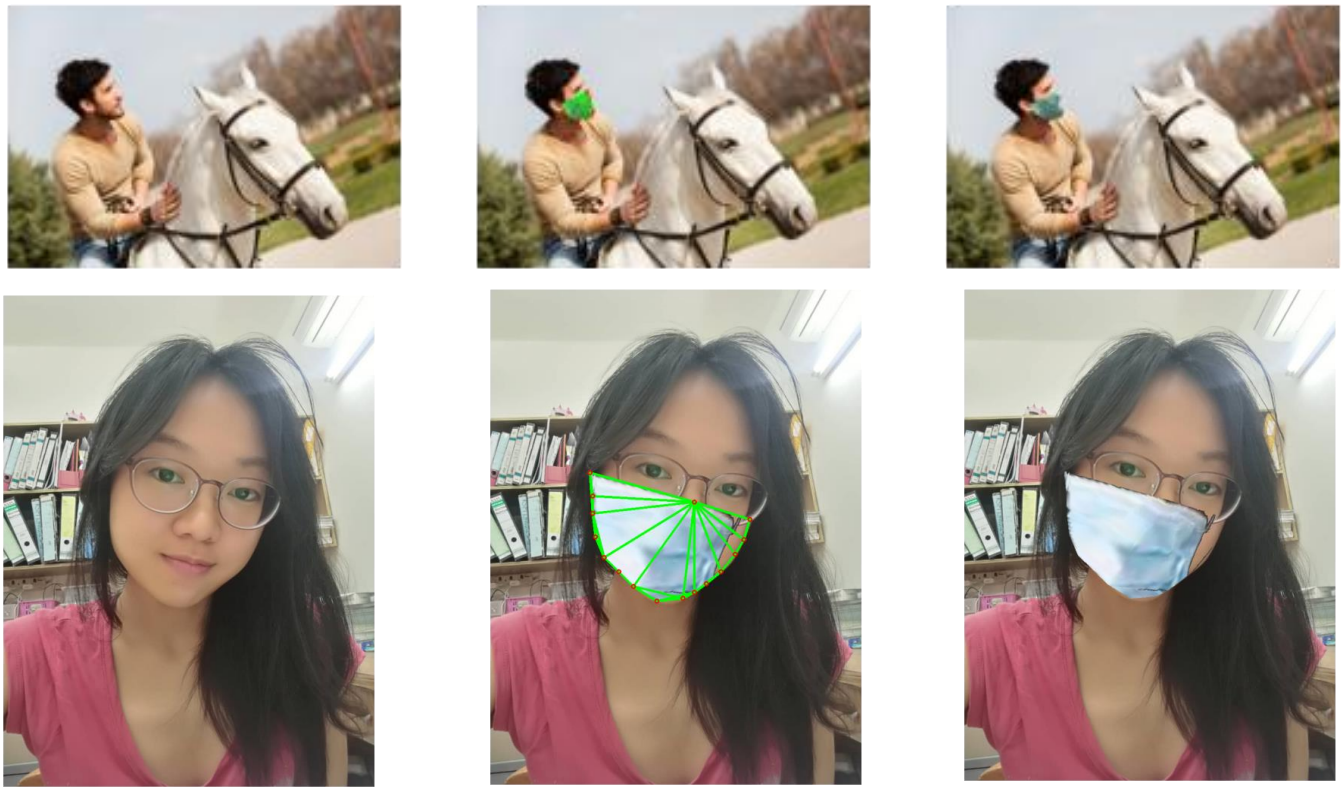
Figure 10 : Wide face angle coverage

Some **good examples** of data augmentation for image with situation of face rotation :
In the examples shown below the mask is masked nicely on the face (where the correct mask

with suitable rotation is selected to do masking).



Some **bad examples** of data augmentation for image with situation of face rotation that still need improvement on:
In the examples shown below the mask is masked nicely on the face (where the mask selected to do masking has the situation of being **unable to mask the whole face completely or masked at an inaccurate angle**).



Challenges/Further Improvement

1. Due to limited mask images, there is not enough masked data available to train a new system.Can increase dataset using MaskTheFace dataset, MaskTheFace[4]
2. Only one type of mask(surgical mask) is supported in this work.Other type of masks such as N95,KN95,Cloth and Gas mask can be next step of further improvement.

Figure 11 : Different type of masks

3. Different mask variations might affect mask augmentation and it affects recognition performance not tested.

Mask variations:Textures/Patterns variations

Figure 12 : Different masks varies in pattern/textures

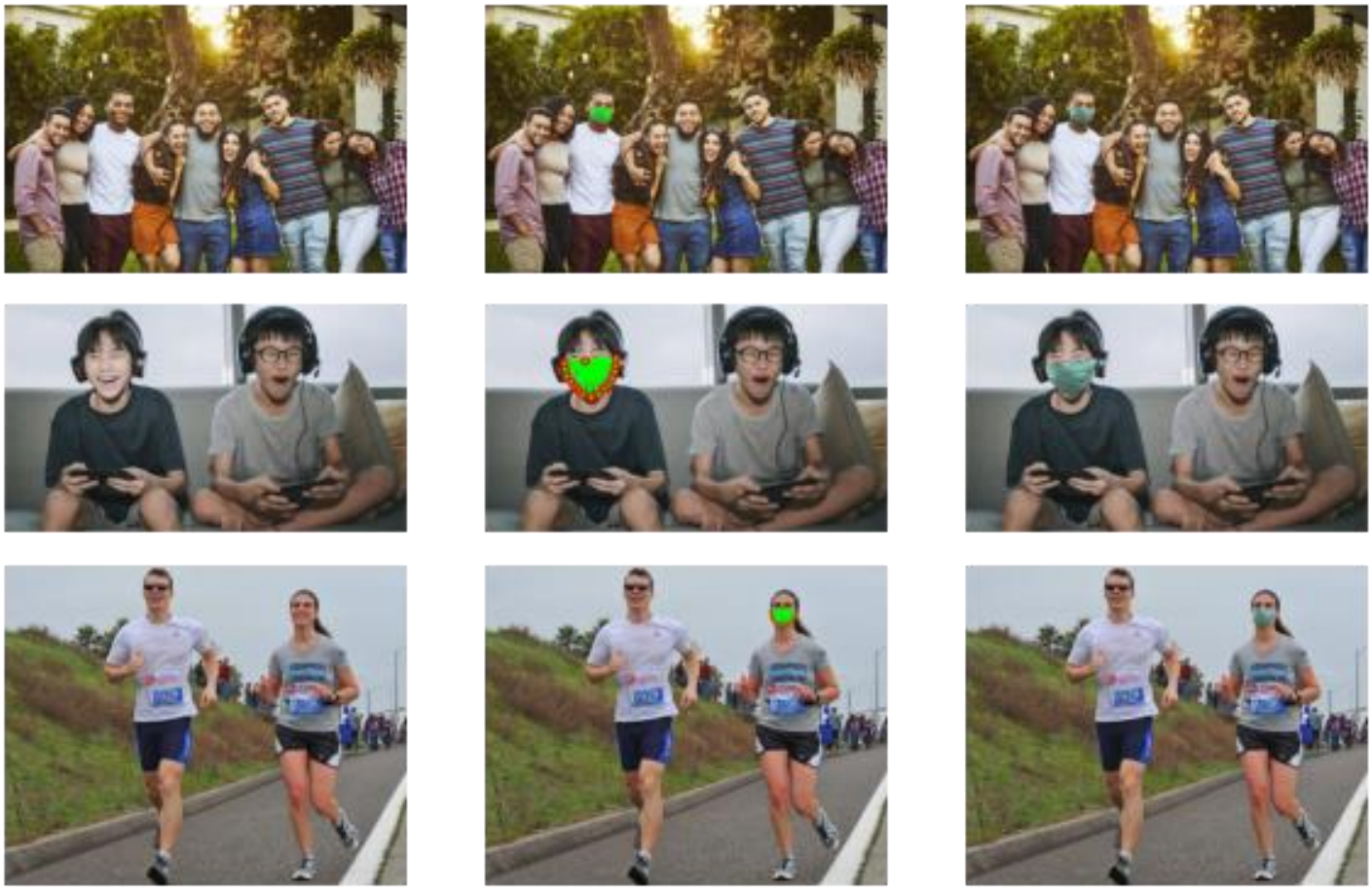
Mask variations:Colours variations

Figure 13 : Different masks varies in colours

Mask variations:Intensity variations

Figure 14 : Different masks varies in intensities

4. Cannot solve for multiple people in an image (Example tested in this work: only one face is masked in a multi-face image) as shown below :



However, this is done successfully in https://github.com/aqeelanwar/MaskTheFace[4]

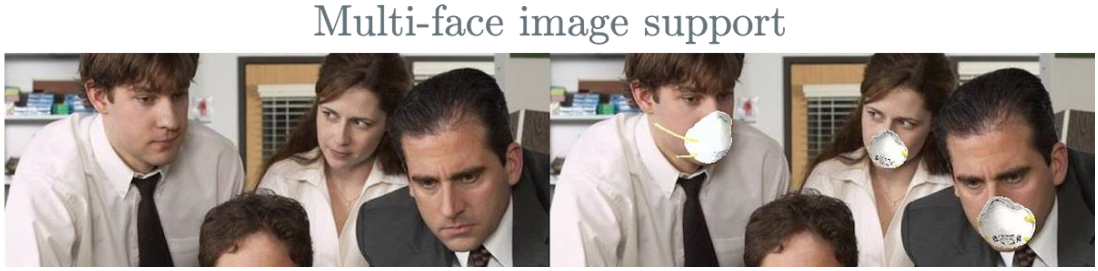


Figure 15 :Multi-face image support

But this is okay for this task because faceID aim is to recognise only one person (owner of the device).

5. Brightness corrected mask application is not available in this work.

Figure 16 : Brightness corrected mask application

References - 5 related items (textbooks, journal, conference)

[1] Potuck, M., & Michael Potuck@michaelpotuck Michael is an editor for 9to5Mac. Since joining in 2016 he has written more than 3. (2021, April 27). *iPhone: How to use Face ID with a mask*. 9to5Mac. https://9to5mac.com/2021/04/27/iphone-how-to-use-face-id-with-mask/.

[2] *Your Own Face On A Mask*. Maskalike. (n.d.). https://maskalike.com/.

[3] Cao, Qiong & Shen, Li & Xie, Weidi & Parkhi, Omkar & Zisserman, Andrew. (2018). VGGFace2: A Dataset for Recognising Faces across Pose and Age. 67-74. 10.1109/FG.2018.00020.

[4] Aqeelanwar. (n.d.). aqeelanwar/MaskTheFace. GitHub. https://github.com/aqeelanwar/MaskTheFace/tree/master/datasets.

[5] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters, 2016. 1

[6]Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In ECCV,2016. 2, 6, 7

[7]W. Liu, Y. Wen, Z. Yu, and M. Yang. Large-margin softmax loss for convolutional neural networks. In ICML, 2016. 2, 3

[8]W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In CVPR, 2017. 1, 2, 3, 4, 5, 6, 7

[9]F. Wang, X. Xiang, J. Cheng, and A. L. Yuille. Normface: l 2 hypersphere embedding for face verification. arXiv:1704.06369, 2017. 2

[10] Yang, J., Bulat, A., & Tzimiropoulos, G. (2020). FAN-Face: a Simple Orthogonal Improvement to Deep Face Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(07), 12621-12628. https://doi.org/10.1609/aaai.v34i07.6953

[11] Deng, Jiankang & Guo, Jia & Xue, Niannan & Zafeiriou, Stefanos. (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition. 4685-4694. 10.1109/CVPR.2019.00482.

[12] 1adrianb. (n.d.). *1adrianb/face-alignment*. GitHub. https://github.com/1adrianb/face-alignment.

[13](Steven), C.-H. L. (2020, July 11). *[Paper Summary] ArcFace: Additive Angular Margin Loss for Deep Face Recognition*. Medium. https://medium.com/@steven413d/arcface-additive-angular-margin-loss-for-deep-face-recognition-3955dca8b469

[14] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. arXiv:1512.01274, 2015. 2, 5

Link to program codes in GitHub or Google Drive, etc.

GitHub:
https://github.com/meiyihTan/WID3008_Image_Processing
Testing:
https://colab.research.google.com/drive/1CCleAFWFaD_8F3dTWJeCrCb_Wamrmoe1?usp=sharing
Training:
https://colab.research.google.com/drive/1zqF0Zt71xhiP-2u25SzkDmc5CMdx4t1b?usp=sharing