Missing Children Family Reunion using Face Recognition

A Graduation Project Thesis
submitted to the Faculty of Engineering, Cairo University
in partial fulfillment of the requirements for the degree of Bachelor of Science in Electronics and Electrical Communication Engineering

By

Abdulrahman Elsayed Shaaban

Ahmed Radwan GadElRab

Ahmed Taha Abdulhaleem

Hussien Sayed Ahmed

Khaled Khaled Sabe

Mohamed Ismail Amer

Sponsored by NajahNow and ONELAB

under the supervision of

Dr. Hassan Mostafa

Cairo University – July 2022

Contents

Introduction 4
Problem Definition 4
Project Objectives and Scope 4
Face Recognition 6
Face Detection task 7
Face Recognition task 7
SphereFace 10
CosFace 11
ArcFace 11
Age invariant face recognition 12
MTCNN 14
Super Resolution 17
SRGAN 19
ESRGAN 22
Real-ESRGAN 25
CFR 30
Description 30
Algorithm 31
MTL 33
1. Feature extraction: 34
2. Attention-based feature decomposition: 36
3. Age-Estimation: 37
4. Face Recognition: 37
5. Identity Conditional Module (ICM) and Face Age Synthesis (FAS): 37
6. Datasets, Training Process, and results 43
7. Problems we faced with this paper 43
Datasets 44
Training datasets 44
Testing datasets 48
Experiments 50
Small Dataset Training 50
Large dataset training 50

False predictions of the model 51
Super Resolution Training 54
Face Rotation and De-occlusion 55
Problems with CFR model 56
Egyptian Dataset 57
Children Only Dataset 57
Pipeline 60
Description 60
Training models 61
Technologies used for the mobile and website application 62
Front End - Flutter 62
Databases 65
Relational database 65
Non-relational databases 67
MongoDB 69
Application running process 72
Splash Screen 72
Login Screen & Signup Screen 72
Home-Page Screen 73
New Searches and Reports 73
Summary Screen 74
Follow your searches and reports 74
Future Work 75
References 76

Introduction

Problem Definition

Missing children, a huge problem facing our society and not our society only but it's a problem everywhere, for example In the United States, an estimated 460,000 children are reported missing every year, and an estimated 100,000 missing children In Germany, and Egypt, there were 2,264 reports about missing children in 2018 and 2019. [40], [41]
We always try to make use of the technologies we have to help us with the problems we face people, some people created Facebook pages to report and help in finding missing children, and some printed posters of missing children and posted them on the streets, etc.

MISSING
CHILD
IF YOU HAVE ANY INFORMATION OR SEEN THIS PERSON, PLEASE CONTACT THE POLICE IMMEDIATLY BY CALLING 911
PLEASE - INFORMATION NEEDED
MGN

Figure 1

The advance in the Machine learning field specifically the Deep learning field helped in many fields like industry and medicine, and it can be used to assist in the problem we face every day which is finding missing children.
Almost all of us have already seen the power of AI in almost every field but it shines the most in computer vision applications because of the considerable advance in this field with minimal errors that can be as little as 0.5%, so it can be used in a critical problem like ours.

Project Objectives and Scope

In this project we used the power of AI and image processing to help in the problem which we've already discussed, so we decided to make a mobile application that can give anyone the ability to help find missing children with just little steps and using the power of AI to help in finding matches of children missing even if there's an age gap in the images, so we did the following:

1. Face Recognition:
A face recognition algorithm is an underlying component of any facial detection and recognition system or software, so it's the main component of our project.

And as we already mention our application is used in finding missing children so it's expected to receive images of the same person (child) at different ages so it was very important to have an age invariant face recognition.

2. Complete Face Recovery GAN:

We noticed that the images taken by the users will not be in an ideal environment for taking images so there could be an image of a child with glasses on or with something on their face, so we'll need to remove that to help us identify the child easily so we used Complete Face Recovery GAN which uses the power of GANs to generate images on the person without any occlusions.

3. Mobile Application:

We wanted to make an application that can be easily accessible by anyone and can also be easily used to encourage people to help, so we made sure that the application is simple and self-explanatory.

Face Recognition

Facial recognition has progressed to become the most appropriate and logical technique in human identification technology. Out of all techniques like signatures, hand maps, and voice identification, facial recognition is preferably used because of its contactless nature. Deep Learning technology is the foundation of the new applications that allow us to use facial recognition.

What deep learning is? It’s a field that operates under artificial intelligence that uses artificial neural networks. Neural networks can be described as a universal function approximator that can learn any mapping between some inputs and outputs in a given dataset.

Face recognition is a technology of identifying and verifying people by their faces from an image, video stream, or real-time.

A task that humans can easily perform, even if there are illumination changes or face changes with age, or if it is covered with accessories or a beard. But it has been a challenging computer vision problem for decades until recently.

Deep learning techniques can leverage huge datasets of faces and learn rich and compact representations of faces, so modern models work the same as the human level at first and later better than it.

In many cases, there is a need to automatically recognize the person in the photo. The reasons for that may be as follows:

- We may want to limit access to resources to one person, called facial recognition.
- We may need to make sure that a person matches an ID, called face verification.
- We may want to assign a name to a face, called face recognition.

Any of these tasks are commonly referred to as the problem of "face recognition" and can be associated with both still images and faces in video streams.

Humans can perform this task very easily. We can detect the face in the image and tell which identity it belongs to if they are known. We can do this very well, such as when the people have aged, are wearing sunglasses, have different colored hair, are looking in different directions, and so on. However, even after more than 60 years of research, this is a difficult problem to run automatically in software. Until very recently.

Process of automatic face recognition:

The face recognition process involves four steps; they are face detection, face alignment, feature extraction, and finally face recognition.

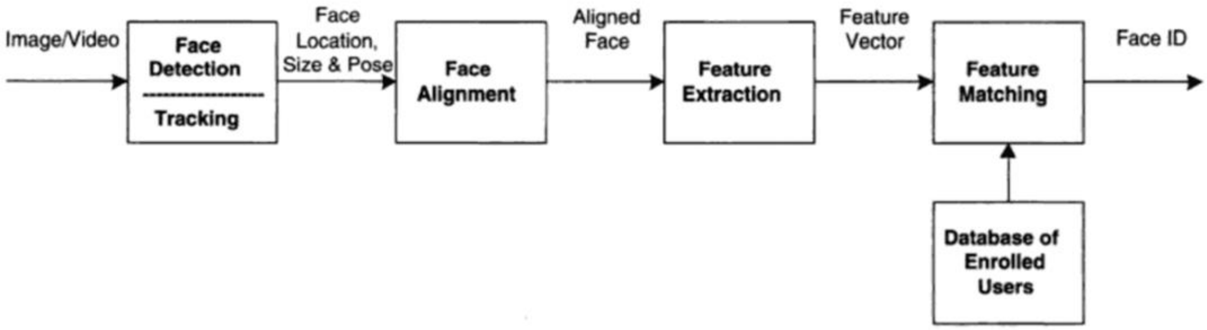


Figure 2. Face recognition process flow [1]

Face Detection task

It’s the problem that requires that both the location of each face in a photograph is identified (e.g., the position) and the extent of the face is localized (e.g., with a bounding box).
It’s a special case of the object detection problem, where the object here is not necessary a face but can be any other object. Therefore, researchers have been inspired by the solutions of object detection available and used them in face detection and innovated more tricks that belong to the faces’ case to increase the accuracy.
Face detection also needs to be reliable because it is the first stage of a more comprehensive face recognition system. For instance, if a face cannot be detected, it cannot be recognized. This means faces must be detected under all conditions of orientations, angles, light conditions, haircuts, hats, glasses, facial hair, makeup, ages, and so on.

Face detection methods can be broadly divided into two main groups:

- Feature-based
- Image-based

The feature-based face detection uses hand-crafted filters that are based on a deep knowledge of the domain. They can be very fast and very effective when the filters match the faces in the image, but they can fail dramatically when tested on the same faces with some variations, which makes them somewhat fragile.

Alternately, image-based face detection learns how to automatically locate and extract faces from the entire image. Neural networks fit very well into this class of methods as they can learn the mapping directly between the input (image of faces) and the output (location of the faces).

In this book, we will focus only on the recent object detection techniques that use deep learning. From R-CNN family [2], [3], and [4] that started all these advances, to finally RetinaNet [5]. And for the face detection case, we will focus on MTCNN [6].

Face Recognition task

There are two main modes of face recognition [1], there are:

- Face Verification. A one-to-one mapping of a given face against a known identity (e.g., is this the person?).
- Face Identification. A one-to-many mapping for a given face against a database of known faces (e.g., who is this person?).

Face recognition can be modeled as a supervised learning task using samples as inputs and outputs.

Input for all tasks is an image with at least one face in it, most likely a detected face from a previous face detection stage that may have been aligned from also a previous face alignment stage.

The output changes depending on the kind of prediction necessary for the task at hand; for instance:

In the case of a task requiring face verification, it can then be a binary class label or binary class probability.

For a task involving face identification, it might be a set of categorical class labels or probabilities.

In the case of a similarity-type task, it can be a similarity measure.

Face recognition has remained and continues to be an active area of research in computer vision. From the early classical techniques such as EigenFaces, to finally deep learning methods that are currently state of the art.

Thanks to the breakthrough of AlexNet in 2012 for the less complex problem of image classification, deep learning techniques for face recognition underwent a frenzy of research and publications in 2014 and 2015 and as a result, the performance quickly achieved near human-level performance on the standard face recognition tasks and then surpass the human-level performance within only three years [8]

In this book, we will only discuss deep learning-based approaches such as TBC.

But first, how can we use deep learning to solve the problem of face recognition?

Let’s say we have a set of labeled images, i.e., we know the identity of each person, we call this set the gallery set. And given a new test image, we call it the probe, we want to figure out the closest image from the gallery set to the probe image. Then, the identity of this closest image would be the identity of the test image.

First idea: compare the test image to every image in the gallery images to get the closest one. We can compare two images by subtracting them, dot product, or using any other metric. This is a simple and intuitive idea, but practically it’s impossible. If we compare one image to itself, we want the result to be close as possible, but if we shift one of them, flip it, rotate it, or any other translation, the comparison would tell us that those are two different images. And this is because we used a metric that's working directly on the feature space (the raw image).

Second idea: pass the two images into a backbone neural network like VGG, Inception, etc., and do the comparison on those outputs from that backbone (the feature space). The output of the backbone can be a vector feature of length 512 for an input image. Comparing the two images using those two vector features is simpler than comparing the raw pixels which can be of 1024*1024 dimensions and can be used practically.

But for this technique to work well, we need to train the backbone such that for the images of the same identity, it gives feature vectors close as possible to each other, and for images of different identities, it gives feature vectors far apart from each other in the feature space. See Figure 3. It's a toy experiment of mapping 8 identities with 2D feature vectors. Each point represents an image and each color represents an identity in the feature space. We can observe that the backbone can gather the images of the same identity close together and far away from other identities.

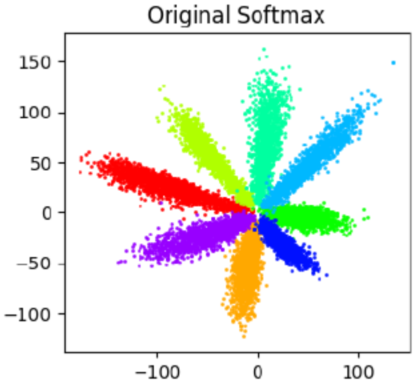


Figure 3. [9]

perhaps the first four papers that used deep learning for this problem are DeepFace [12], DeepID [13], VGGDFce [14], and FaceNet [15].

Now we know how to use neural networks to solve the face recognition problem, and the next question arrives, how to improve its accuracy?

From Figure 3, we can observe some overlapping between the points in the feature space, this leads to getting an image of one identity to be closer to another identity than to itself. And that can result in a wrong prediction.

Solution? We need something that can reduce this overlapping, i.e., compacting the points of the same identity more. From this idea, researchers have published a lot of papers that introduce innovative loss functions that express this demand to the neural network (making the points of the same identity very compacted in the feature space). The most famous papers in this area and what we are going to focus on in this book are CosFace [9], SphereFace [10], and ArcFace [11].

SphereFace

Problems with softmax loss:
Softmax loss doesn't attempt to keep the positive pairs closer and negative pairs farther as the features that it learns are only separable features and they aren't discriminative enough. The loss function can be defined as:

$$\mathbf{L} = -\frac{1}{M}\sum_{i=1}^{M} log(\frac{e^{W_{y_i}^T f_i}}{\sum_{k=1}^{C} e^{W_k^T f_k}})$$

Where:
W represents the weights of the last layer
f_i is the feature vector of the input image

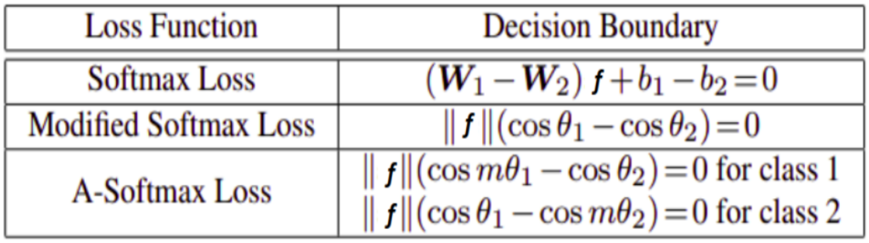
The boundaries of the softmax loss between any two classes can be described as:

$$||W_1||||f_i||cos(\theta_1) > ||W_2||||f_i||cos(\theta_2)$$

Where this is simply the dot product comparison between feature vectors and the weights of the last layer, θ_i is the angle between the feature vector and the weights. Coving normalized weights, we have $||W|| = 1$ and no bias, the loss function becomes more angularly distributed and this called modified softmax. Now the boundary depends only on the angle between the features and the weights.

SphereFace adds to the modified softmax an angular multiplicative margin m where the new form of the boundary is:
$C1: \cos(m\theta_1) > \cos(m\theta_2)$
$C2: \cos(m\theta_2) > \cos(m\theta_1)$

This adds more difficulty to learning and makes the model learn to produce features of the same class that is close to each other as it adds an angular margin.



Loss Function	Decision Boundary
Softmax Loss	$(\boldsymbol{W}_1 - \boldsymbol{W}_2)\boldsymbol{f} + b_1 - b_2 = 0$
Modified Softmax Loss	$\|\boldsymbol{f}\|(\cos\theta_1 - \cos\theta_2) = 0$
A-Softmax Loss	$\|\boldsymbol{f}\|(\cos m\theta_1 - \cos\theta_2) = 0$ for class 1 $\|\boldsymbol{f}\|(\cos\theta_1 - \cos m\theta_2) = 0$ for class 2

And now the new loss function can be written as:

$$\mathbf{L_{ang}} = -\frac{1}{M}\sum_{i=1}^{M} log(\frac{e^{||f_i||cos(m\theta_{y_i,i})}}{e^{||f_i||cos(m\theta_{y_i,i})} + \sum_{k\neq y_i}^{C} e^{||f_k||cos(\theta_{k,i})}})$$

CosFace

Similar to SphereFace, CosFace modifies the softmax loss function by adding a margin but instead of adding an angular margin, it adds the margin to the cosine space making the loss more consistent with cosine similarity that is used at the evaluation stage to differentiate between different classes. It also removes the contribution of the magnitude of feature vector by setting it to a constant value $||f_i|| = s$ as it shouldn't have any contribution at the testing stage. The new loss function now is defined as:

$$\mathbf{L_{lmc}} = -\frac{1}{M}\sum_{i=1}^{M} log(\frac{e^{s(cos\theta_{y_i,i}-m)}}{e^{s(cos\theta_{y_i,i}-m)} + \sum_{k\neq y_i}^{C} e^{s.cos(\theta_{k,i})}})$$

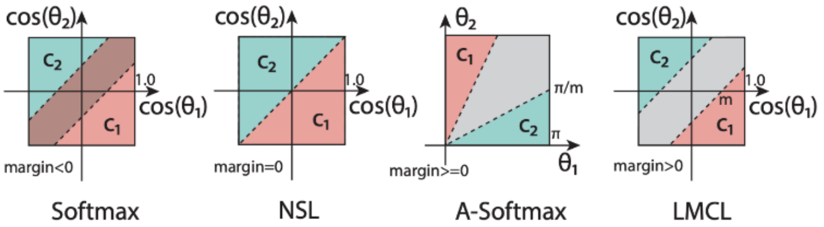


Figure 4 Comparison of feature spaces of loss functions

ArcFace

The authors of ArcFace propose using an additive angular margin in the softmax function to further improve the discriminative power of the model.

$$\mathbf{L_{arcface}} = -\frac{1}{M}\sum_{i=1}^{M} log(\frac{e^{s(cos(\theta_{y_i,i}+m))}}{e^{s(cos(\theta_{y_i,i}+m))} + \sum_{k\neq y_i}^{C} e^{s.cos(\theta_{k,i})}})$$

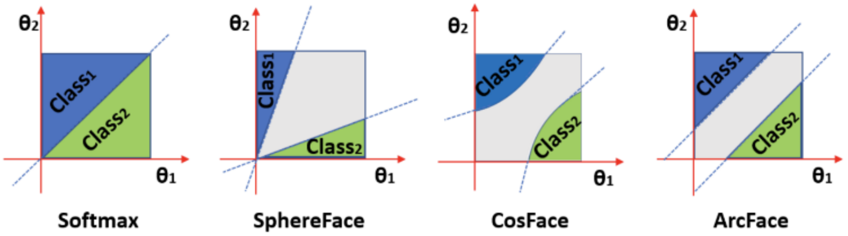


Figure 5 Comparison of angular feature spaces of loss functions

Age invariant face recognition

It's an extension of the normal face recognition problem but the catch here is that normal face recognition has problems with cross-age face recognition it's hard to distinguish between the same person at different ages especially if there's a huge age gap between these images, so it became a hot topic in deep learning research trying to achieve a reliable cross-age face recognition and in this part, we will talk about this topic and how it's important in many ways.

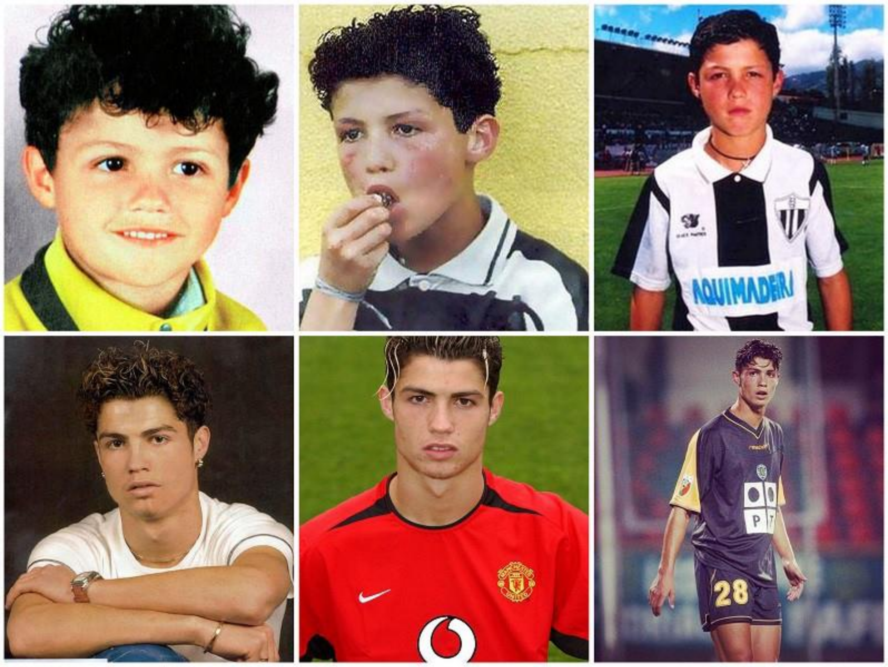


Figure 6 Image showing how facial features changes over age

Age invariant Face Recognition faces the same challenges as normal face recognition like Pose, lightning and image quality as shown in Figure 7

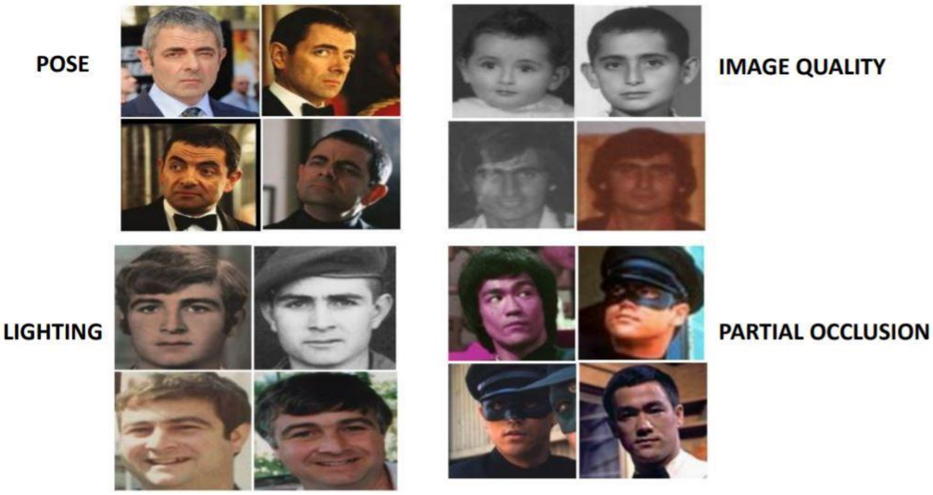


Figure 7

And here are some applications for Age invariant face recognition:

1. Biometrics: Face recognition plays an important role in this field so improving it by making it age invariant boosts the robustness of the model. During this period, a face may have undergone significant change. It is an example where the person's face will compare with the image taken long before.

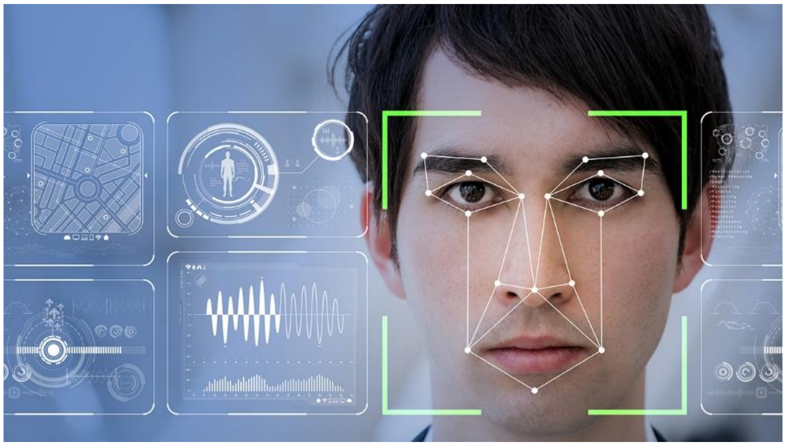


Figure 8

2. Forensics: AIFR is very important in forensics applications. it helps to recognize and identify wanted criminals by comparing suspects' faces to mugshots that could have been taken years before. There may be cases when the forensics experts require to change the age of the criminal's face. Hence, this demands a robust face recognition model in the presence of aging.
3. Medicine: It is used for diagnosing the disease by finding something unusual in the aging of a person. Identifying the disease early will help the treatment of a person effectively than discovering later.

MTCNN

MTCNN (Face Detection with a Multi-Task Cascaded Convolutional Neural Network) it's a network used in face detection and alignment, so it was very important to use in our problem as the images are not expected to have the faces already cut or aligned in a specific position, let's dive deep in this network and see how it works.

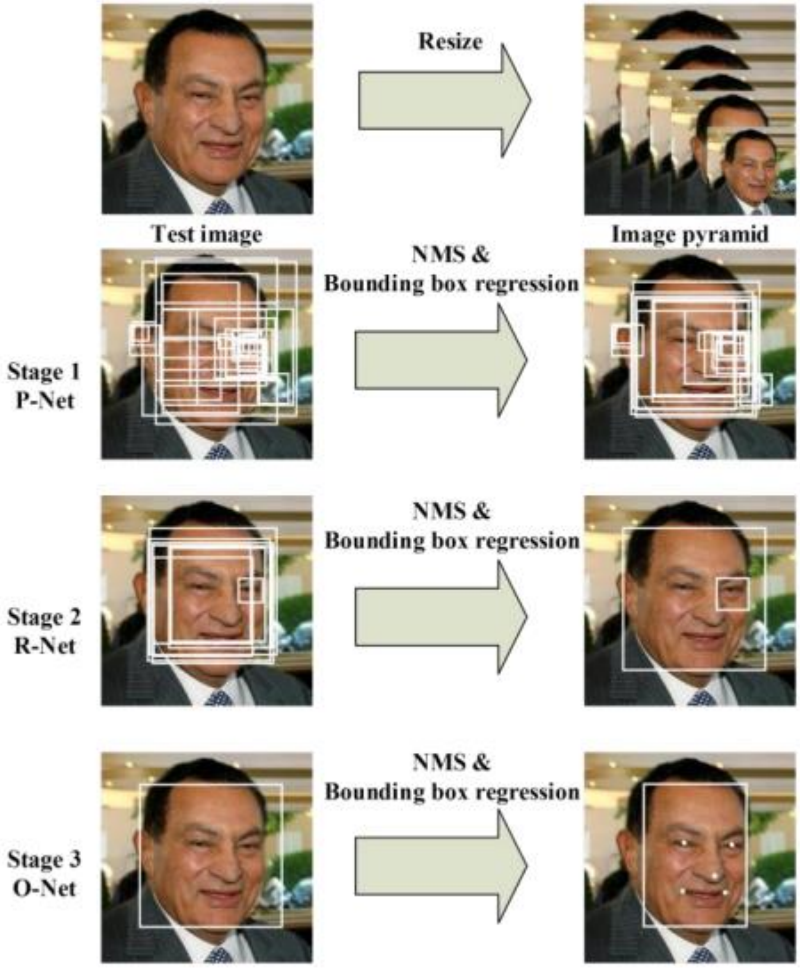


Figure 9

First, it resizes the image then it goes through 3 stages to finally detect any faces in the image,

Stage 1: in this stage, the network produces candidate windows using a shallow CNN called P-Net (Proposal Network), and this results in highly overlapping boxes.

Stage 2: in this stage, the network refines and filters out windows using a more complex CNN called R-Net (Refinement Network) and this merges highly overlapped boxes.

Stage 3: in this stage, the network refines and filters out more windows and also adds face features (like points on the eyes and nose) using a more complex CNN called O-Net (Output Network).

Figure 10 is the details of every network we talked about.

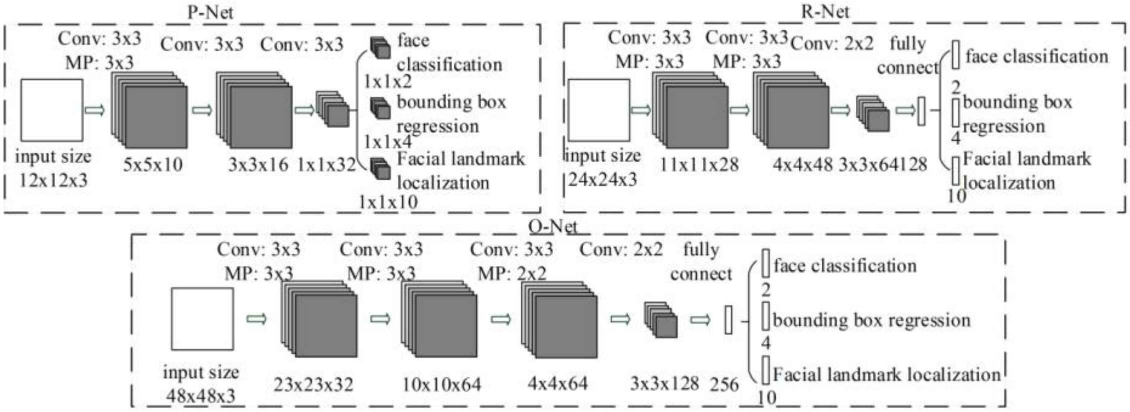


Figure 10

As we can see we have 3 errors to minimize

1. Face Classification:

The learning objective is formulated as a two-class classification problem. For each sample x_i, the cross-entropy loss:

$$L_i^{det} = -(y_i^{det}\log(p_i) + (1 - y_i^{det})(1 - \log(p_i)))$$

Where, p_i is the probability produced by the network that indicates a sample being a face, The notation y_i^{det} denotes the ground-truth label.

2. Bounding box regression:

For each candidate window, it predicts the offset between the box and the nearest ground truth (i.e., the bounding boxes' left top, height, and width). The learning objective is formulated as a regression problem and employed Euclidean loss for each sample x_i:

$$L_i^{box} = \left\| \hat{y}_i^{box} - y_i^{box} \right\|_2^2$$

where, $\hat{y}_i^{box}$ is the regression target obtained from the network and y_i^{box} is the ground-truth coordinate. There are four coordinates, including left top, height, and width so it's a 4-dimensional vector.

3. Facial landmark localization:

Similar to the bounding box regression task, facial landmark detection is formulated as a regression problem and we minimize the Euclidean loss:

$$L_i^{landmark} = \left\|\hat{y}_i^{landmark} - y_i^{landmark}\right\|_2^2$$

Where, $\hat{y}_i^{landmark}$ is the facial landmark's coordinate obtained from the network and $y_i^{landmark}$ is the ground-truth coordinate. There are five facial landmarks, including left eye, right eye, nose, left mouth corner, and right mouth corner so it's a 10-dimensional vector.

Now after we defined each loss function, we now will see how it's all trained together, using the following function

$$\min \sum_{i=1}^{N} \sum_{j\in\{det,box,landmark\}} \alpha_j \beta_i^j L_i^j$$

N: total number of samples
i: the number of the sample
j: is either (Facial landmark localization, Bounding box regression or Face Classification)
α: is a weighting for each loss because sometimes we don't train all losses equally for example in P-Net we use ($\alpha_{det} = 1, \alpha_{box} = 0.5, \alpha_{landmark}$ =0.5).
β: is either 1 or 0 and it's is the sample type indicator so if we don't want use a sample with specific loss, we can do that.
L: is the loss for each problem.

We didn't train this network ourselves as there are many pretrained models so it's reasonable to use a pretrained version that is trained with enough data.

Super Resolution

Image Super Resolution is the process of increasing an image's resolution from low resolution (LR) to high resolution (HR). it’s an important class of image processing and recently deep learning. The following are some common uses for it:

- Surveillance: detecting and identifying the faces in images from the low-resolution images obtained from the security cameras.
- Medical: generating high-resolution MRI images from the obtained low-resolution ones, is a lot easier than capturing the high-resolution images directly.
- Media: media can be sent at a low resolution and then get enhanced using SR at the client-side which reduces the server costs.

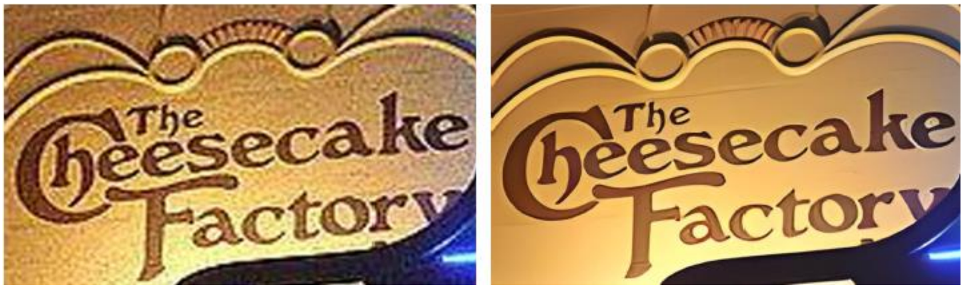


Figure 11

We can model the problem as if the high-resolution image is the input, which goes into some unknown degradation function with some noise, and the output of this function is the low-resolution image.

$$I_{LR} = D(I_{HR}; \sigma)$$

Where D is the degradation function, σ is the noise, I_{LR} is the input image, and I_{SR} is the output image.

Only the low- and high-resolution images are provided, but the degradation function and the noise are unknown. If we know the degradation function and noise, we can find the inverse degradation function and use it to find the high-resolution image of any low-resolution input.

$$I_{HR} = D^{-1}(I_{LR}; \sigma)$$

We can use neural networks which are universal function approximators to learn this inverse function using the low resolution and the corresponding high-resolution images.

The traditional methods for solving this problem are the nearest neighbor, bilinear interpolation, and bicubic interpolation. They are very simple and fast than the more advanced methods but cannot model the complex degradation function and often suffer from artifacts and unpleasing effects.

So, researchers have been searching for new techniques which use deep learning to solve the problem. As neural networks are universal function approximators, we can use them in an end-to-end manner to learn the inverse function from the low-resolution and the corresponding high-resolution images

There are many techniques in this regard such as Pre-Upsampling Super Resolution, Post-Upsampling Super Resolution, Residual Networks, Multi-Stage Residual Networks, Recursive Networks, Progressive Reconstruction Networks, Multi-Branch Networks, Attention-Based Networks, and Generative Models.

In this book, we focus on the generative models, specifically, we study SRGAN [16], ESRGAN [17], and Real-ESRGAN [18].

But first, let’s get a high-level understanding of the GAN framework.

GANs are capable to produce fake data that looks like real data. Some GAN applications aim to enhance image quality.

GAN contains two main networks namely the generator network and the discriminator network.

The generator network tries to generate the fake data and the discriminator network tries to distinguish between real and fake data, hence helping the generator to generate more realistic data.

SRGAN

In this paper, the authors present a generative adversarial network (GAN) for image super-resolution (SR). it's a framework that for the first time, is capable of inferring photo-realistic natural images for 4X upscaling factors. To achieve that, they proposed a perceptual loss function which consists of an adversarial loss and a content loss. The adversarial loss pushes the solution to the natural image manifold using a discriminator network that is trained to differentiate between the super-resolved images and original photo-realistic images. And the content loss is motivated by perceptual similarity instead of similarity in pixel space. As the behavior deep learning-based super-resolution method is principally driven by the choice of the objective function, this paper is capable of recovering the finer texture details, where previous work couldn't.

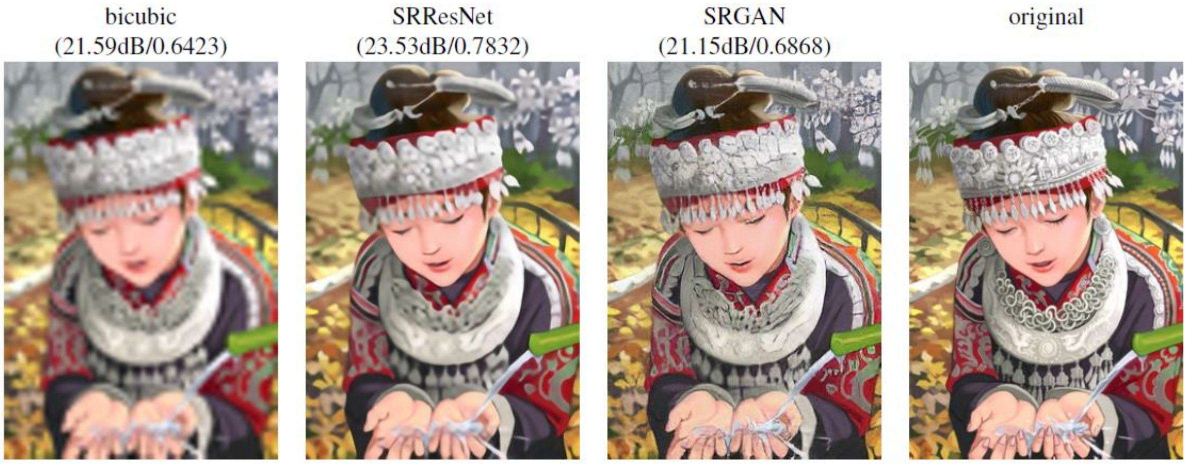


Figure 12

The common optimization target for supervised SR was the minimization of the mean squared error (MSE) between the recovered SR image and the ground truth. This is convenient as minimizing MSE also maximizes the peak signal-to-noise ratio (PSNR), and is also a common measure for evaluating and comparing different SR algorithms. However, MSE and PSNR cannot capture the perceptually relevant difference such as high texture detail that a human eye can capture. This is because the MSE is defined on the raw pixel space of the images.

This can be seen in Figure 12, where the SRGAN output is conceptually better than the SRResNet, though the PSNR is lower. Highest PSNR does not necessarily reflect the perceptually better SR result.

This paper proposes a super-resolution generative adversarial network (SRGAN) architecture based on a novel perceptual loss using high-level feature maps of the VGG network that can capture the high texture details combined with a discriminator that encourages the generated solution to be close to the ground truth images.

In training, a low-resolution image (ILR) is obtained by applying a gaussian filter to a high-resolution image (IHR) followed by a down-sampling operation.

1. Generator function G estimates for a given input LR image its corresponding HR image which is a super-resolved image SR.

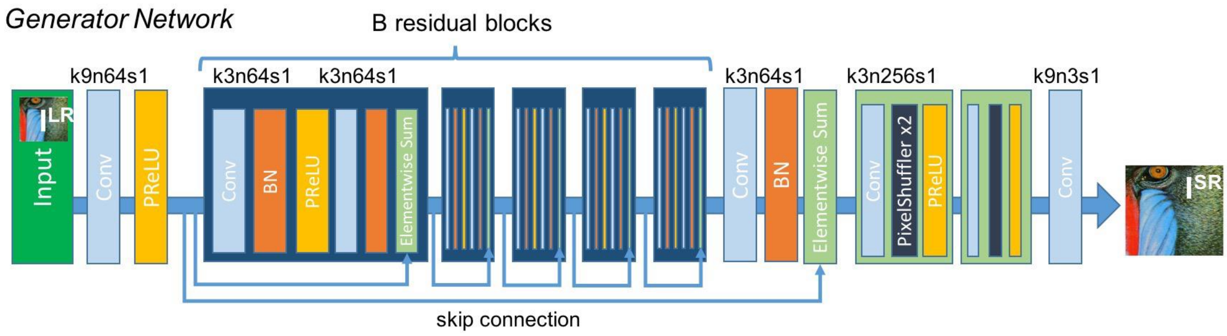


Figure 13

$$\hat{\theta}_G = \arg\min_{\theta_G} \frac{1}{N} \sum_{n=1}^{N} l^{SR}(G_{\theta_G}(I_n^{LR}), I_n^{HR})$$

2. Discriminator D is trained to distinguish super-resolved images and real images.

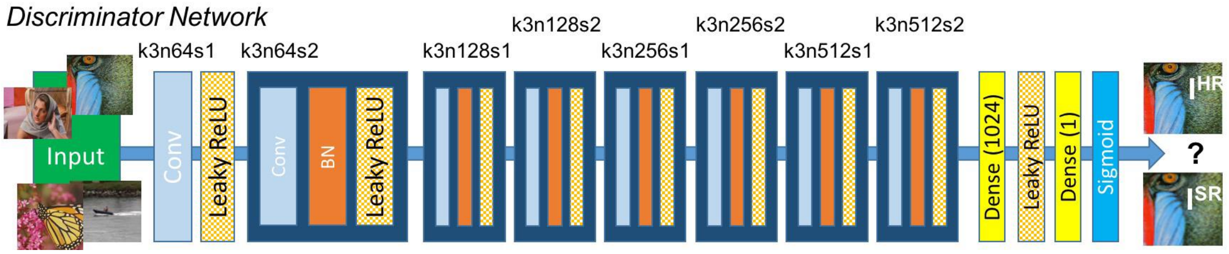


Figure 14

$$\min_{\theta_G} \max_{\theta_D} \; \mathbb{E}_{I^{HR} \sim p_{\text{train}}(I^{HR})}[\log D_{\theta_D}(I^{HR})] + \mathbb{E}_{I^{LR} \sim p_G(I^{LR})}[\log(1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))]$$

Loss function

The loss function is formulated as the weighted sum of a content loss and an adversarial loss as:

$$l^{SR} = \underbrace{\underbrace{l^{SR}_{\mathrm{X}}}_{\text{content loss}} + \underbrace{10^{-3} l^{SR}_{Gen}}_{\text{adversarial loss}}}_{\text{perceptual loss (for VGG based content losses)}}$$

Content loss

Instead of relying on MSE on the raw pixels, the loss is calculated on the output of a pre-trained 19 layers VGG network. With $\text{Ø}_{i,j}$ indicate the feature map obtained by the j-th convolution (after activation) before the i-th maxpooling layer within the VGG19 network. The VGG loss is then defined as the Euclidean distance between the feature representations of a reconstructed image and the ground truth image:

$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

Adversarial loss

This encourages our network to favor solutions that reside on the manifold of natural images, by trying to fool the discriminator network.

$$l^{SR}_{Gen} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

Results

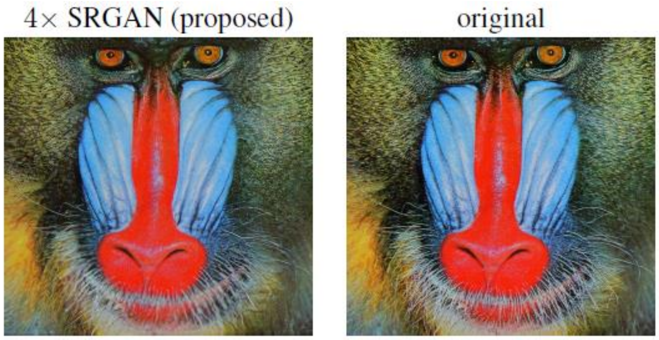


Figure 15

Super-resolved image (left) is almost indistinguishable from original (right).

ESRGAN

As the name depicts, this is an enhanced version of the previous Super-Resolution Generative Adversarial Network (SRGAN) paper. This method aims to further enhance the visual quality and reduce the unpleasant artifacts generated in SRGAN

The network's main high-level architecture is the same, but a few new ideas are added and some are tweaked, ultimately leading to an efficiency improvement.

> *"To further enhance the visual quality, we thoroughly study three key components of SRGAN — network architecture, adversarial loss, and perceptual loss, and improve each of them to derive an Enhanced SRGAN (ESRGAN)." [17]*

Network architecture

The architecture of the ESRGAN generator follows the baseline of SRGAN but replaces the residual block with the RRDB block and removes the batch normalization (BN) layers. The RRDB block is inspired by the DenseNet architecture and connects all layers within the residual block directly with each other.

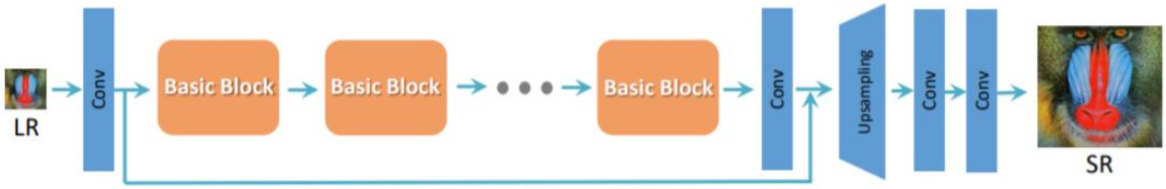


Figure 16

Figure 16 shows the generator architecture where Basic Block is actually the RRDB.

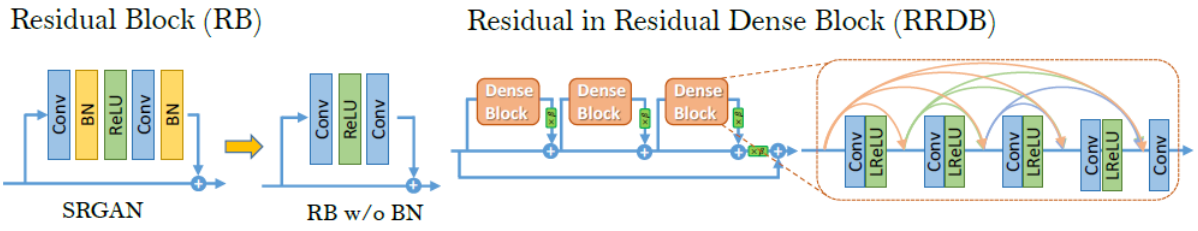


Figure 17

From Figure 17, it can be seen that the (BN) layer is removed.

In many network architectures, it has been found that removing BN layers improves speed while lowering computation complexity and memory usage. It also improves the performance; this is because the statistics of each layer are very different for every image and also for test images and thus violating the assumption of BN that the features for train and test images are similar.

Meanwhile, the RRDB increases the network complexity and gives it more capacity so that it can generate more complex images and ultimately boosts the performance.

Adversarial loss

The second improvement is using relativistic average GAN (RaGAN) instead of the standard GAN. Instead of the standard discriminator which outputs the probability that an image is real or fake, the relativistic discriminator outputs the probability that a real image is relatively more realistic than a fake image.

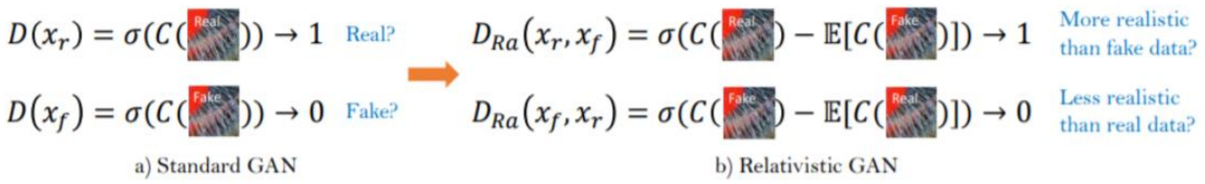

$D(x_r) = \sigma(C(\text{Real})) \to 1$
Real?
$D(x_f) = \sigma(C(\text{Fake})) \to 0$
Fake?
a) Standard GAN
$D_{Ra}(x_r, x_f) = \sigma(C(\text{Real}) - \mathbb{E}[C(\text{Fake})]) \to 1$
More realistic than fake data?
$D_{Ra}(x_f, x_r) = \sigma(C(\text{Fake}) - \mathbb{E}[C(\text{Real})]) \to 0$
Less realistic than real data?
b) Relativistic GAN

Figure 18

The authors found that this modification helps the generator recover more realistic texture details.

Perceptual loss

The third improvement modifies the perceptual loss of the previous SRGAN paper by using the features before activation instead of after activation as in SRGAN. This is because first; the authors found that the activated features are very sparse, especially after a very deep network, and second; the activated features cause inconsistent brightness in comparison to the ground-truth images. and thus, provide weak supervision and lead to inferior performance.

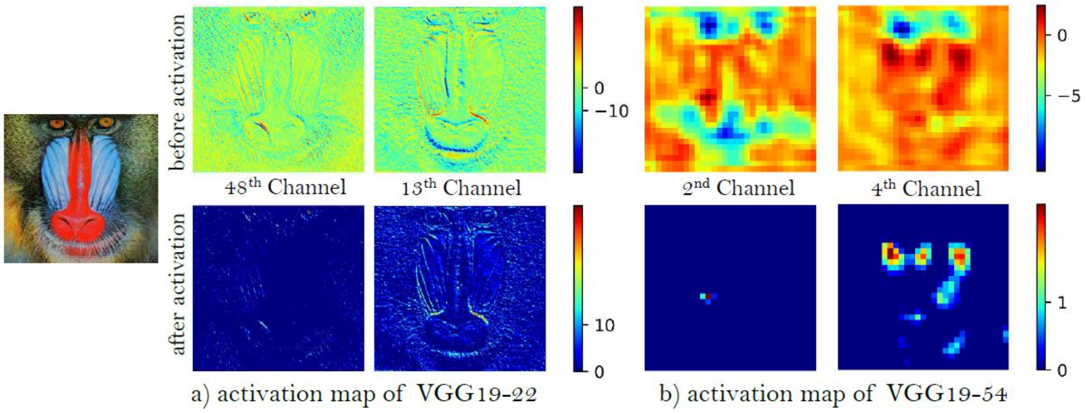


Figure 19

This adjustment of the perceptual loss provides sharper edges and more visually pleasing results.

Therefore, the total loss for the generator is:

$$L_G = L_{\text{percep}} + \lambda L_G^{Ra} + \eta L_1$$

Where L_{percep} is the perceptual loss, L_G is the relativistic generator loss, and L_1 1-norm distance between the recovered image and the ground-truth.

Results

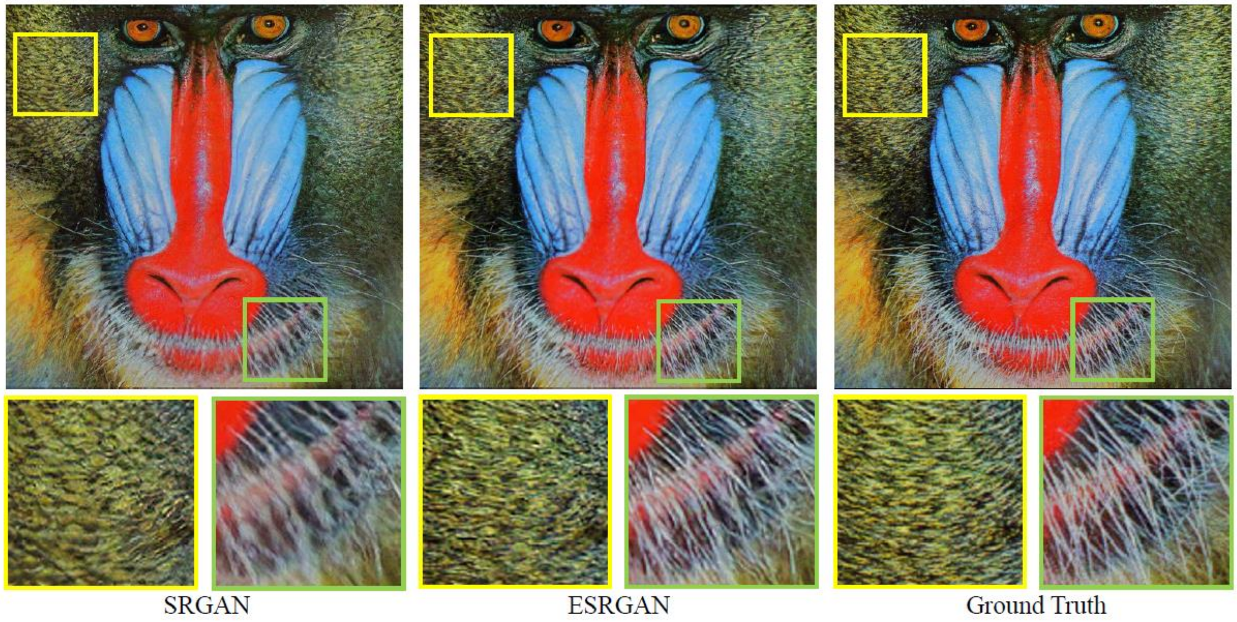


Figure 20

ESRGAN outperforms SRGAN in sharpness and details.

Real-ESRGAN

The third and final paper we are going to discuss is the Real-ESRGAN paper, which adds to ESRGAN new innovations that make it capable of restoring most real-world images and achieving better visual performance.

The main contributions of this paper are:

1. High-order degradation process to model practical degradations.
2. Modification of the discriminator to use U-Net discriminator with spectral normalization to increase discriminator capability and stabilize the training dynamics.
3. Training with pure synthetic data.

Degradation model

To be able to use deep learning in the super-resolution problem, the data is required to be collected as pairs of low-resolution images and the corresponding high-resolution ones. Initially, we have the high-resolution image, and then, it’s passed through some down-sampling techniques (degradation model) to acquire the low-resolution copy. But this degradation model is not what happens in real life. In reality, the degradation model is more complex. it usually comes from complicated combinations of different degradation processes, such as imaging systems of cameras, image editing, and Internet transmission.

> *"For example, when we take a photo with our cellphones, the photos may have several degradations, such as camera blur, sensor noise, sharpening artifacts, and JPEG compression. We then do some editing and upload it to a social media app, which introduces further compression and unpredictable noises. The above process becomes more complicated when the image is shared several times on the Internet." [18]*

However, restoring these photos after such degradation is incredibly difficult, largely because the degradation process is unknown and different for each image. It’s not easy to learn the inverse of the undefined degradation model.

Let's take a quick look at previous solutions to this issue:

Based on the underlying degradation process, existing techniques can be loosely divided into implicit modeling and explicit modeling

Implicit modeling methods utilize the data distribution learning of the GAN framework to obtain the degradation model, however, they are restricted to the degradations found in training datasets and do not apply well to out-of-distribution images.

Explicit modeling methods directly use the classical degradation model which consists of blurring, down sampling, noise, and JPEG compression. However, the degradations that occur in the real world are typically too complicated to be represented by a straightforward mixture of different degradations.

Thus, in real-world samples, these approaches are likely to be unsuccessful.

Let’s see those degradation operations in more detail:

Blurring

Blurring operation is used to approximate real camera blurring and it’s simulated as a convolution with a linear Gaussian filter.

Noise

Noise is applied by adding a Gaussian noise or Poisson noise. Poisson noise is sometimes used to model the sensor noise caused by statistical quantum fluctuations, i.e., variation in the number of photons sensed at a given exposure level.

Downsampling

Downsampling is a basic operation for producing low-resolution images from high-resolution ones. There are several algorithms for downsampling such as area resize, bilinear interpolation, and bicubic interpolation. Each one of them produces a unique effect, some produce blurry effects and others produce over-sharp images.

JPEG compression

JPEG compression is a popular lossy compression method for digital images. When applied to an image, the image loses some information for the benefit of storage size reduction. The more reduction in size, the more loss of information.

To synthesize the low-resolution images to build the training pairs, the author started with a typical degradation model which consists of the traditional degradation operations discussed above; blurring, adding noise, down sampling, and JPEG compression. For example, given the ground-truth image y, the degradation process D could be a combination of a blur kernel k, down sampling with scale factor r, adding noise n, and JPEG compression. We could express this process as the equation below:

$$\boldsymbol{x} = \mathcal{D}(\boldsymbol{y}) = [(\boldsymbol{y} \circledast \boldsymbol{k}) \downarrow_r + \boldsymbol{n}]_{\mathtt{JPEG}},$$

However, this trained model still cannot resolve some complicated degradations in the real world, as shown in Figure 21

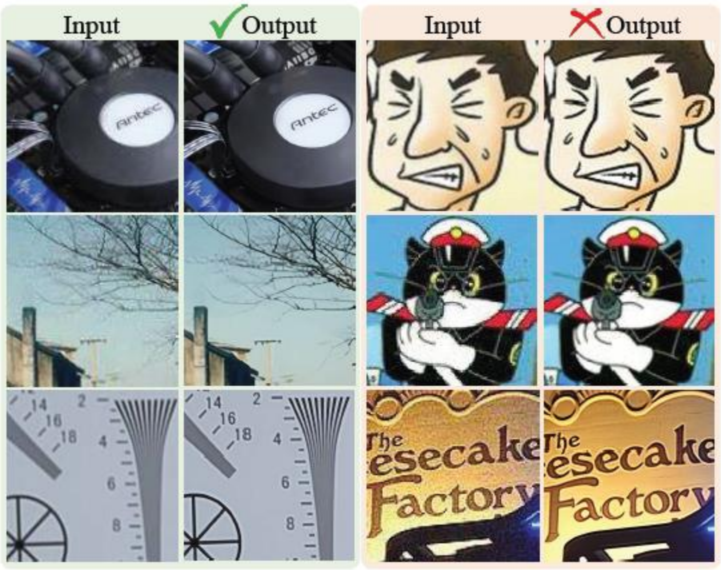


Figure 21

It's because there is a huge difference between the simulated low-resolution images and the simulated low-resolution images. For example, the classical degradation model only includes a fixed number of the above classical degradation operations in a fixed order, which is considered a first-order model. But the actual degradation processes are fairly varied and typically involve a number of actions such as imaging system of cameras, image editing, Internet transmission, etc. For instance, when we have a low-resolution image that we want to resolve, and it originally was in high-resolution but got into some degradation process, the degradation process may include a complicated combination of different degradation processes. Specifically, the image may have been taken many years ago with a cellphone that contains degradations such as camera blur, sensor noise, low resolution, and JPEG compression. The image may have also been edited which introduces some artifacts. And after that, it has uploaded through some social media apps which introduce transmission losses and different compressions. Not to mention that an image is could have been sent many times on the internet.

Therefore, such a complicated degradation process could not be modeled with the classical first-order model described above. Thus, the authors propose using a high-order degradation model to better simulate real-world degradation. An n-order model involves n repeated degradation processes as shown in the following equation:

$$\boldsymbol{x} = \mathcal{D}^n(\boldsymbol{y}) = (\mathcal{D}_n \circ \cdots \circ \mathcal{D}_2 \circ \mathcal{D}_1)(\boldsymbol{y}).$$

Where here we repeat the degradation process D n times.

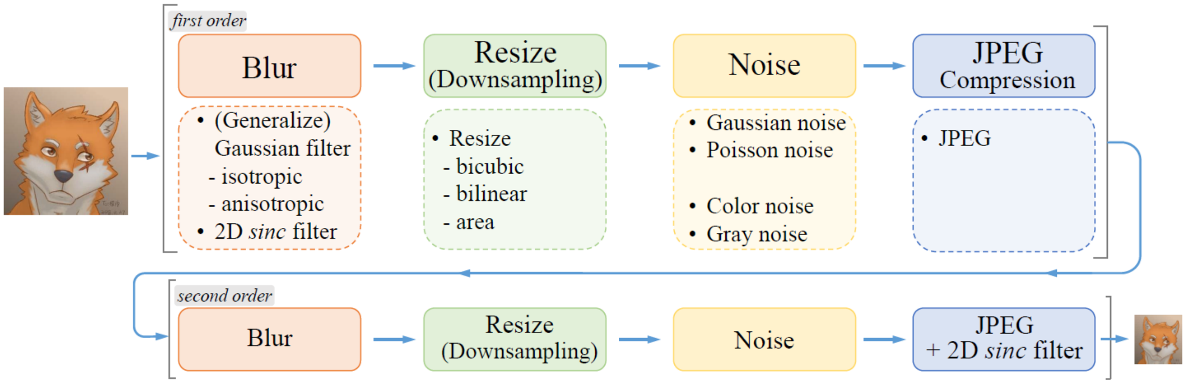


Figure 22

Figure 22 is an overview of the pure synthetic data generation pipeline. It's a more realistic degradation that is modeled using a second-order degradation process, where each degradation process follows the conventional degradation model such as blur, downsampling, noise, and JPEG compression. The input is a high-resolution image and the output is a low-resolution version of the input. We repeat this process for every high-resolution image in the available data, and hence, we end up we pairs of low- and high-resolution images suitable for the super-resolution model.

Network and training

The generator is the same as the ESRGAN generator, i.e., a deep network with several residual-in-residual dense blocks (RRDB), as shown in Figure 23.

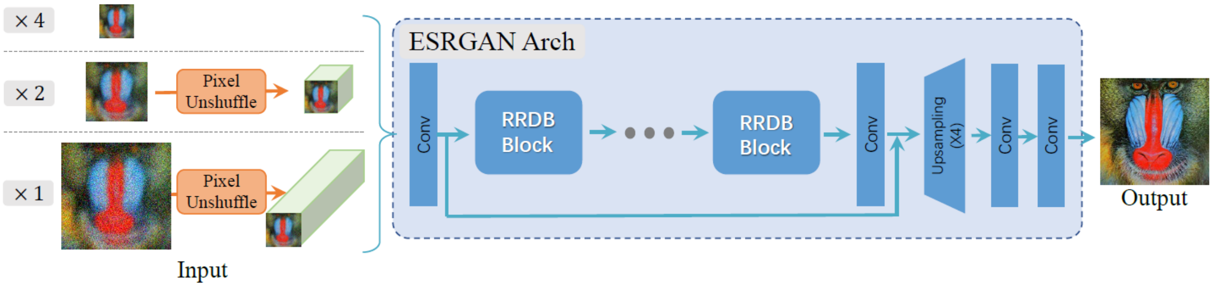


Figure 23

It's also extended to work with scale factors of x2 and x1, as the ESRGAN was only working with a scale factor of x4. As the ESRGAN generator is a heavy network, the image in cases of x1 and x2 goes first through a pixel unshuffle operation (the inverse of pixel shuffle) to reduce the spatial size and enlarge the channel size before feeding inputs into the ESRGAN generator. Thus, most calculations are performed in a smaller resolution space which can reduce the computational resources consumption.

U-Net discriminator

The discriminator here aims to judge on a much larger degradation space than ESRGAN and requires greater discriminative power. So, the original ESRGAN discriminator is replaced with a much more powerful one which is the U-Net.

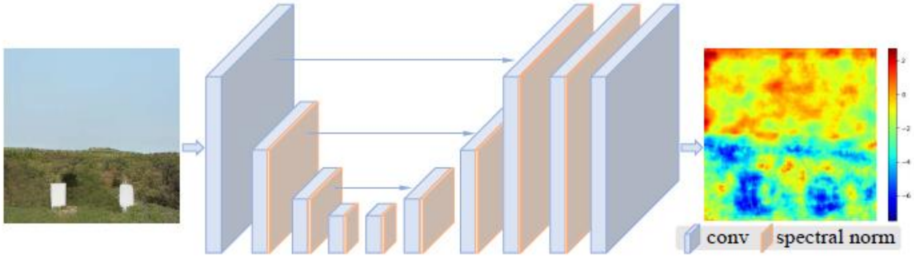


Figure 24

From Figure 24, the U-Net outputs realness values for each pixel and can provide detailed per-pixel feedback to the generator.

Result

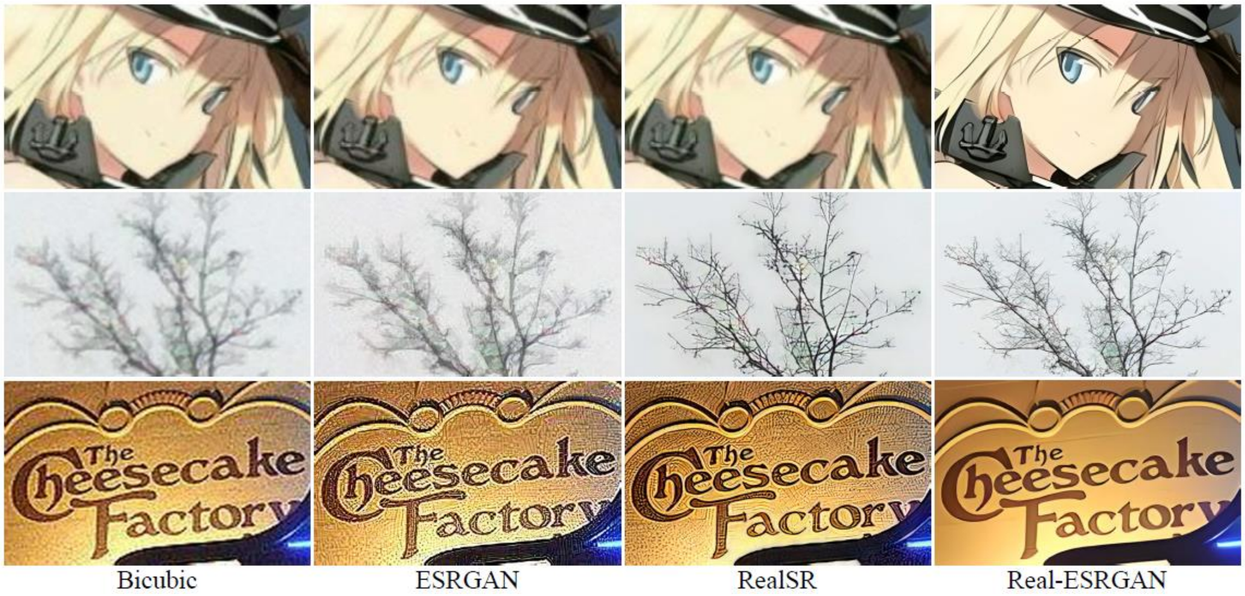


Figure 25

The Real-ESRGAN model trained with pure synthetic data is capable of enhancing details while removing annoying artifacts for common real-world images. [18]

CFR

In our project, we used CFR to enhance images so we can get a better result generally, and as we working on the problem where there are missing children and the images that are taken for them are highly possible to be in various poses and with occlusions, so we used this model to mitigate this problem.

By noticing the MTL-Face and ArcFace we found that it mainly fails when there is a severe pose in the image and if the images have occlusions that make the textures of the face different for images of the same person and this confirms our approach to use CFR.

Various studies have been conducted on face-related tasks, including facial recognition, expression recognition, and re-identification with progress in a deep neural network. Despite recent improvements, extreme pose and occlusion remain obstacles to the above tasks. Face rotation and de-occlusion can alleviate these problems but are challenging tasks because of the lack of high-quality training data.

Description

What is CFR?

Complete Face Recovery (CFR) is a fully unsupervised method for joint face rotation and de-occlusion that can reach a very good point in resolving this problem and helping the face-related tasks like our project on face recognition.

CFR results in a high-quality output image preserving the identity and the textures of the image as shown in Figure 26.

Qualitative results on CelebA-HQ and FFHQ datasets.

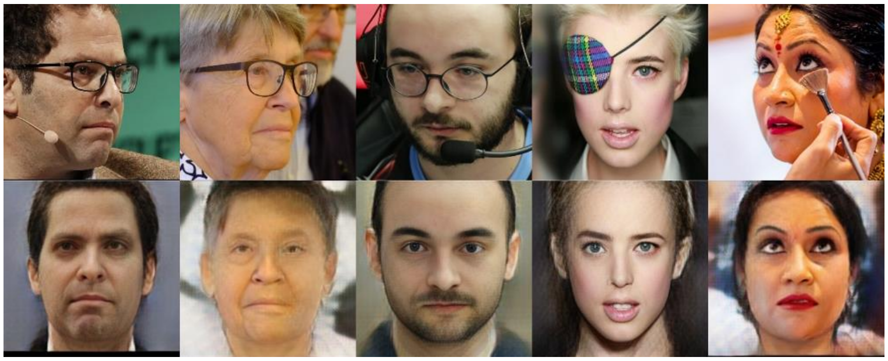


Figure 26

Algorithm

This method covers two challenging tasks:

1. Estimating the mask for the occlusion area that can provide 3D face-based guidance to naturally restore the texture of the occlusion area.
2. Providing strong self-supervision for joint face rotation and de-occlusion by proposing a Swap-R&R strategy that transfers occlusions from an image to the estimated 3D face and rotates it twice, as if rotated from a different pose.

First, two 3D faces are generated from the input image using a 3D face reconstruction model. One 3D face is created by estimating the 3DMM parameters, and the other 3D face is created by projecting the texture of the input image onto the estimated 3D shape.

The rendered image Re from the 3D face with the estimated texture is an occlusion-free facial image owing to the limited representation power of 3DMM model and the rendered image Rp from the 3D face with the projected texture is a facial image that includes occlusion.

Then the CFR applies a strong self-supervision with a Swap-R&R strategy. Specifically, the mask for the occlusion area is coarsely calculated based on the color and structural differences between the two face images.

Then, the occlusion areas are exchanged between two rendered images by utilizing the calculated occlusion mask. Thus, Re and Rp become the occluded and occlusion-free images, respectively.

Next, the model obtains a damaged facial image through two rotate-and- render operations. The process rotates a face in the 3D space back and forth, and re-renders it onto the 2D plane. Rp becomes a rendered facial image, with any random pose through a first rotate-and-render operation. A second rotate-and-render operation creates a facial image with the original pose. Finally, our generator learns to restore the original input image from two images, Re and Rp. The generator is designed to provide structural and textural information from Re for the recovery of Rp.

In addition, there is an occlusion parsing path to focus on occluded and damaged regions so that more natural images can be recovered. On the other hand, in the inference process, the generator creates an image without occlusion from the rendered images of the two 3D faces.

Swap-R&R strategy to generate training pairs for self-supervision

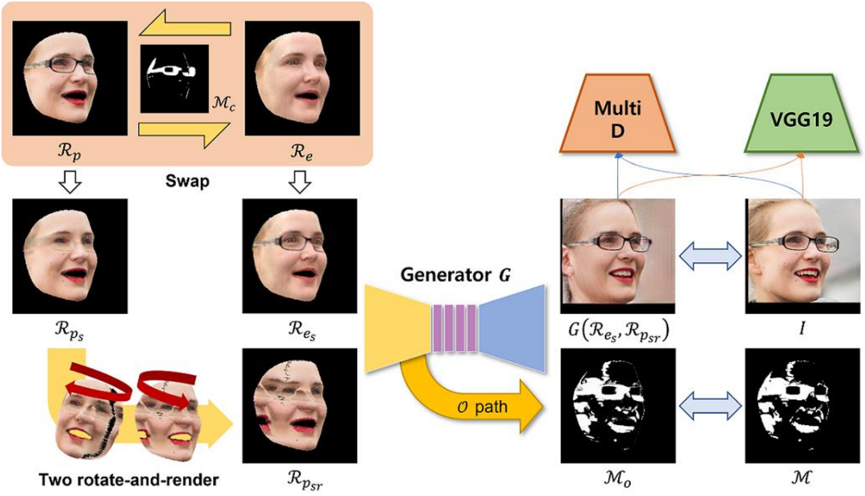


Figure 27

The overall framework of the CFR_GAN

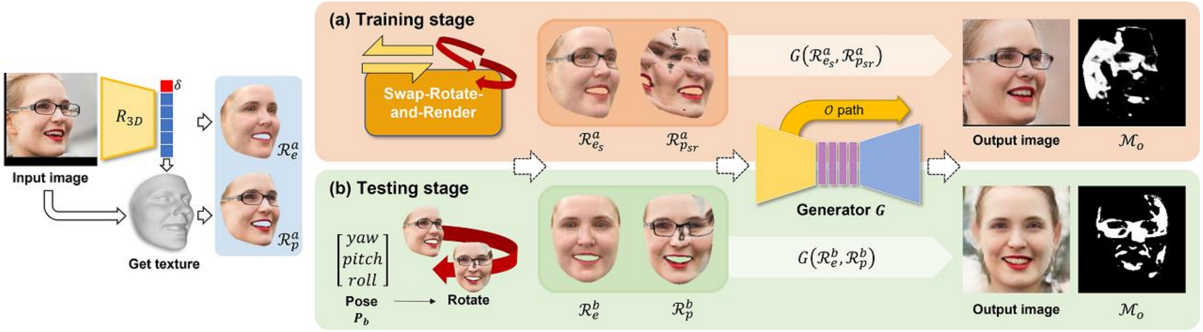


Figure 28

MTL

MTL (When Age-Invariant Face Recognition Meets Face Age Synthesis: A Multi-Task Learning Framework) is the core of our pipeline, it's the part that is used in face recognition.

What's important in our problem is that the face recognition should be age-invariant (i.e., it should consider a person at the age of 5 and 25 the same even though there are some changes in this person's face over the years), that's why it's not an easy problem.

This model has two parts, one part recognizes the face and outputs an embedding for the image (AFIR) and the other part generates images for the person at different ages (FAS), we used the part that recognizes faces only and we will explain why after we talk about the model in details.

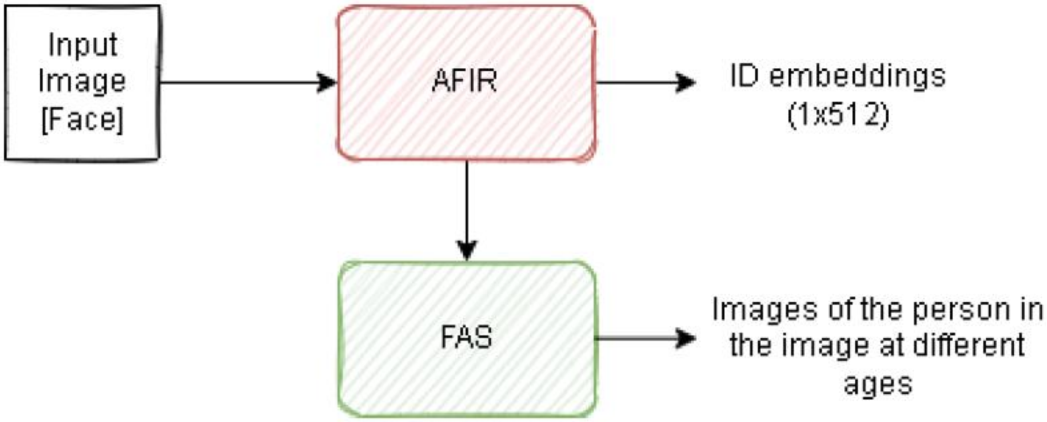


Figure 29. Brief illustration of the MTL

Now let's dive deep in the details of the MTL,

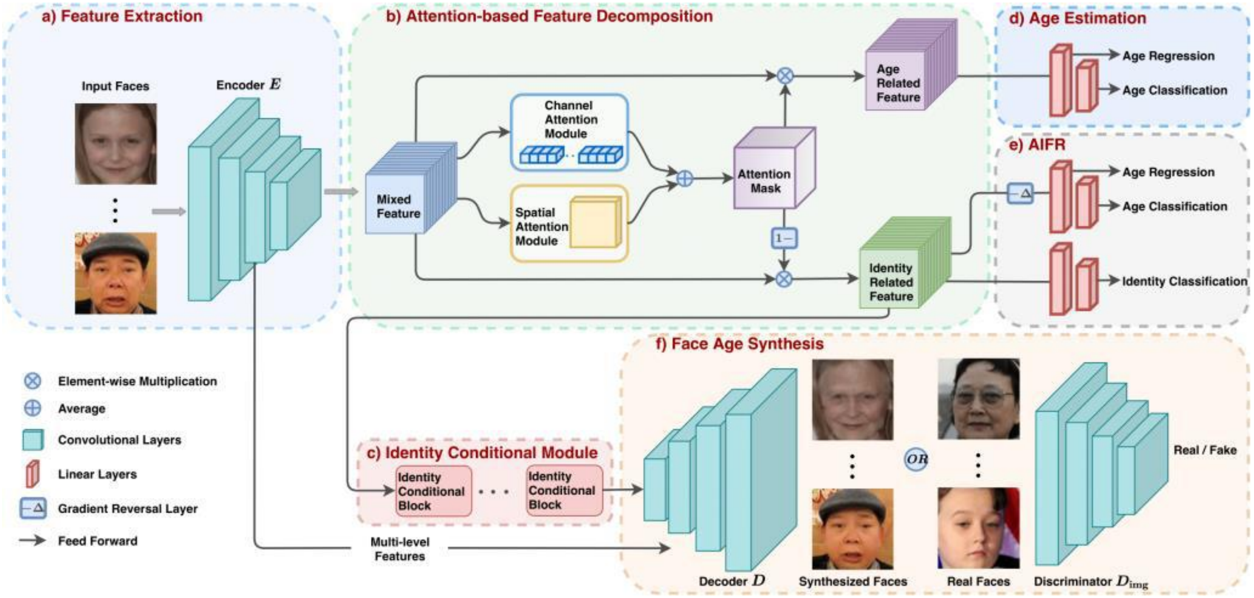


Figure 30. MTL model

1. Feature extraction:

It extracts the mixed feature maps from input faces which are sent to the AFIR and FAS.
It's done by Encoder E and it's a ResNet network and can use Squeeze and excitation network with it as an option.
What is **ResNet**? its basic block is something called Residual block

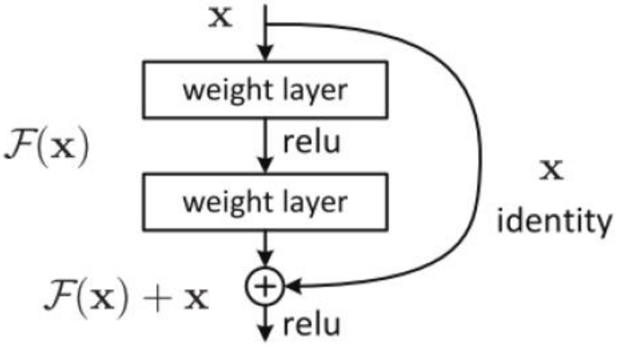

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 31. residual block

They noticed with the normal CNN networks when we increase the number of layers and go deeper this worsens the network performance because of a problem called gradient vanishing that the gradients get so small the deeper the model gets.
So, they tried to add what is called "Skip connection" in each residual block, and by doing that we can go deeper than a normal CNN without suffering from the gradient vanishing problem.

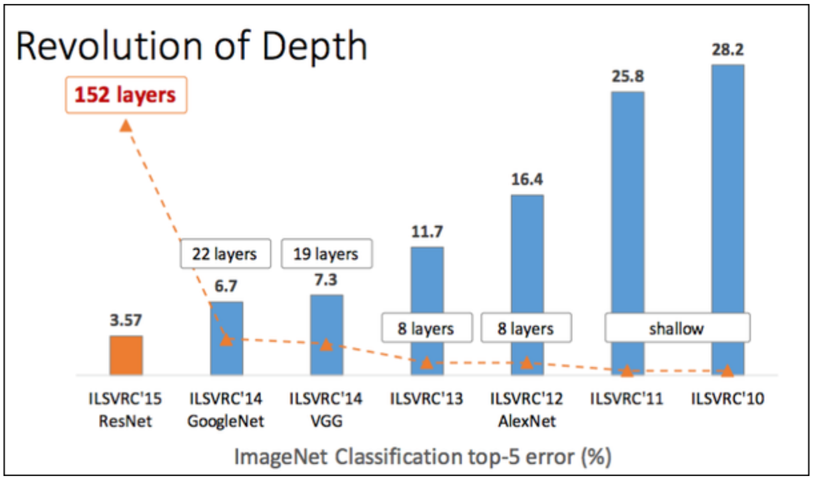


Figure 32. Comparison between different networks

From Figure 32, we can see that ResNet allowed us to have deeper networks and of course an increase in accuracy.
Also, in this paper (MTL) there are many variants for the ResNet (like 152, 101, etc.) and it's the number of layers in the network.

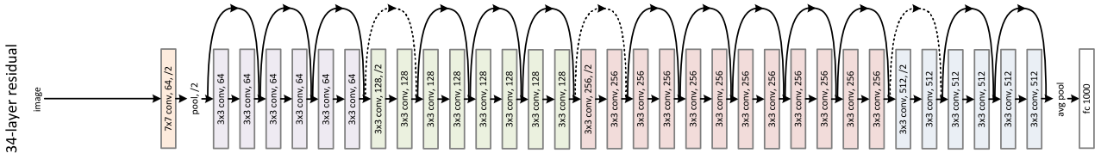


Figure 33. An example of a ResNet network (ResNet-34)

So, it was reasonable to use it as the Encoder.

What is **squeeze and excitation**? the problem with a normal CNN is that the network weights each of its channels equally when creating the output feature maps, so the squeeze and excitation idea is that it weights the channels by its importance and they found out that this works better for example they found that ResNet-50 with Squeeze and Excitation gives almost the same accuracy as ResNet-101 with no Squeeze and Excitation, also it adds almost no computational cost (only about 1%).

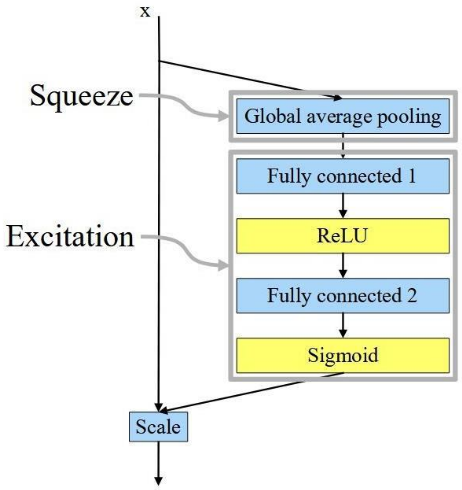


Figure 34. Squeeze and Excitation network

So, a squeeze and excitation network works as follows:
1. The function is given an input convolutional block and the current number of channels it has
2. We squeeze each channel to a single numeric value using average pooling A fully connected layer followed by a ReLU function adds the necessary nonlinearity.
3. Its output channel complexity is also reduced by a certain ratio.
4. A second fully connected layer followed by a Sigmoid activation gives each channel a smooth gating function.
5. At last, we weigh each feature map of the convolutional block based on the result of our side network.
So, it was reasonable to use it with the ResNet in the Encoder.

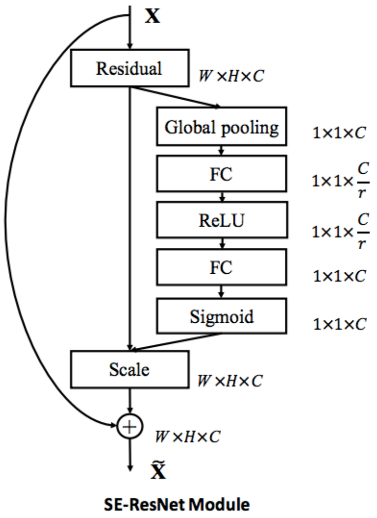


Figure 35. Squeeze and Excitation network with a residual block

2. Attention-based feature decomposition:

After extracting mixed features from the image, they're then decomposed to age and identity features using the attention model.

$$X = X_{id} + X_{age}$$

It consists of a Channel attention module and a Spatial attention module; we will talk briefly about both parts.

Channel attention:

is a module for channel-based attention in convolutional neural networks. We produce a channel attention map by exploiting the inter-channel relationship of features. As each channel of a feature map is considered as a feature detector, channel attention focuses on 'what' is meaningful given an input image. To compute the channel attention efficiently, we squeeze the spatial dimension of the input feature map, we first aggregate spatial information of a feature map by using both average-pooling and max-pooling operations, generating two different spatial context descriptors, which denote average-pooled features and max-pooled features respectively, both descriptors are then forwarded to a shared network to produce our channel attention map.

Spatial attention:

It's a module for spatial attention in convolutional neural networks. It generates a spatial attention map by utilizing the inter-spatial relationship of features. Different from channel attention, spatial attention focuses on where is an informative part, which is complementary to channel attention. To compute spatial attention, we first apply average-pooling and max-pooling operations along the channel axis and concatenate them to generate an efficient feature descriptor. On the concatenated feature descriptor, we apply a convolution layer to generate a spatial attention map that encodes to emphasize or suppress.

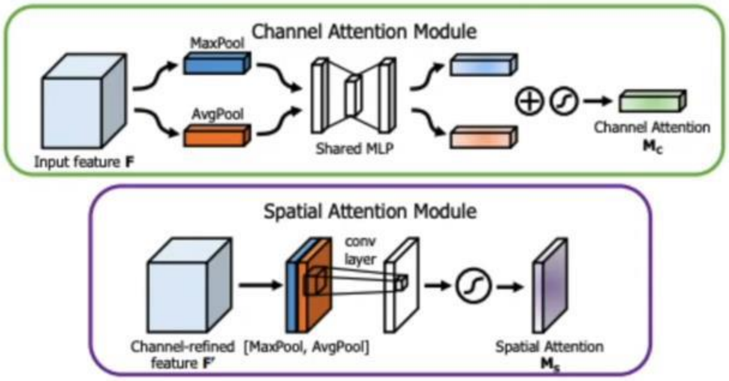


Figure 36

This attention module is supervised by:

1- Age estimator that helps it extract age-related features.
2-Face recognition that that helps it extract identity-related features.

3. Age-Estimation:

It's a convolution network and outputs age and age group, in our project we used 7 age groups which split ages into groups as follows 0→10, 10→20, 20→30, 30→40, 40→50, 50→60, 60→, so we get a total of 7 age groups, splitting ages into groups helps with FAS since the changes over time are minor with a small age gap.

4. Face Recognition:

It consists of 2 parts:

Age-Estimation with gradient reversal layer:

Using the same age estimator module used in the Age estimation part, but before it, there's a gradient reversal layer added which increases the error of the age estimator so that when the error backpropagates to the attention module it helps it extract identity features better.

Identity Classification:

A CosFace model is used in this part to recognize Faces and increase distances between identities but before that, the identity-related features (coming from the attention module) are then passed through a network to decrease the size of the feature vector to 512.

5. Identity Conditional Module (ICM) and Face Age Synthesis (FAS):

As already explained the FAS job is to generate images of the person (input image) at different ages.

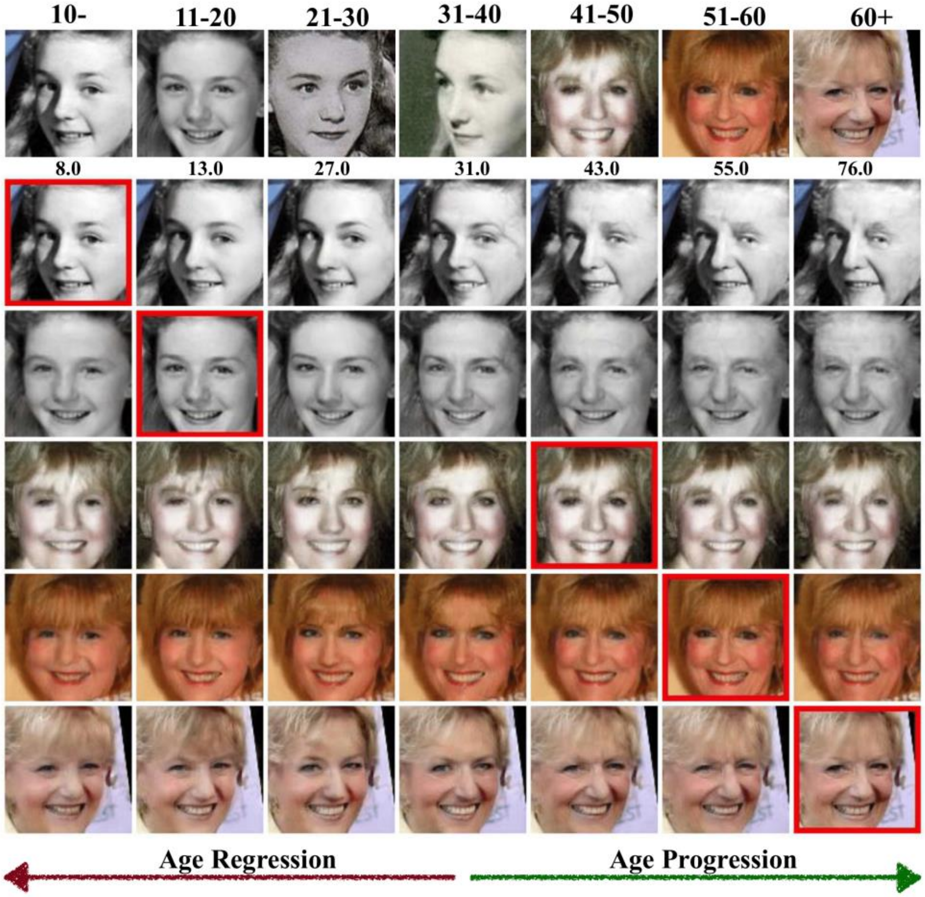


Figure 37

Figure 37 is the sample results of MTL Model. First row: the real faces of the same person at different ages with estimated age labels underneath (Face recognition part). Remaining rows: the synthesized faces when given input faces in the red boxes (FAS part).

So, from the idea of this FAS, it needs the identity of the person as input to generate the images, we could use one-hot encoding for each identity but it has the following drawbacks:

1) one-hot encoding represents the age group-level aging/rejuvenation pattern, ignoring identity-level personalized patterns, particularly for different genders and races.
2) one-hot encoding may not ensure the age smoothness of the synthesized face.

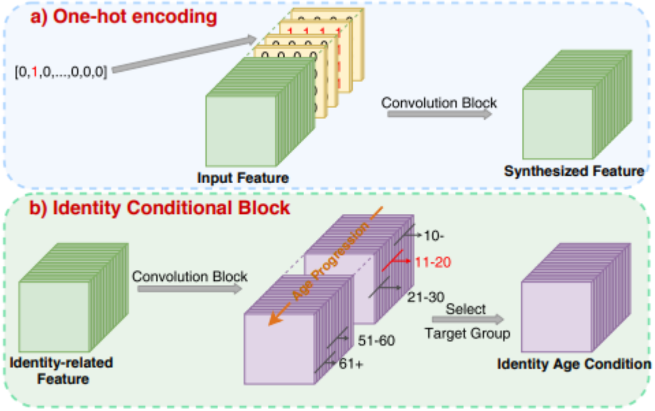


Figure 38

So, they decided to use what is called ICM which consists of 4 Identity conditional block (ICB) to achieve an identity-level aging/rejuvenation pattern, with a weight-sharing strategy to improve the age smoothness of synthesized faces. Specifically, the ICB takes the identity-related feature from the Attention module as input to learn an identity-level aging/rejuvenation pattern, also a weights-sharing strategy is used to improve the age smoothness of synthesized faces so that some convolutional filters are shared across adjacent age groups. The rationale behind this idea is that faces are gradually changed over time, where the shared filters can learn some common aging/rejuvenation patterns between adjacent age groups.

After that, the ICM output is then passed to a GANS network to generate the aged faces, but because the attention model makes us lose information from the original image-like backgrounds, so the Encoder E output is also passed to the Generator to keep the details of the input image like the background.

GANS and cGANS:

Figure 39 This is an image of a person who doesn't exist generated by SyleGANs

GANs (generative adversarial networks) are networks that generate fake images and can be used in other domains like creating new music or new videos, so it has an interesting potential to be helpful in many fields.

A GAN consists of 2 networks a Generator and a Discriminator, the generator gets a noise as an input (no information) and generates an image then the Discriminator works as a classifier to tell if the image generated by the generator is fake or not,

And it has what is called the Minimax Loss function:

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

The generator tries to minimize this loss and the discriminator tries to maximize it, that's why it's called the Minimax loss function.

D(x): is the discriminator's estimate of the probability that the input of real images x is real.
Ex: is the expected value over all real data instances.
G(z): is the generator's output with a noise input z.
D(G(z)): is the discriminator's estimate of the probability that the generator output is real.
Ez: is the expected value over all random inputs to the generator.

That was the basic GAN network after that there have been many variants and enhancements to the network to increase the quality of the output images and use it in different applications, and here are some types of GANs:

1. Conditional GAN (cGAN): it's a type of GAN that involves a condition as an input to the generator to generate an image and it's the type used in the MTL model as we want to keep the identity in the generation of the image.

2. Super Resolution GAN (SRGAN): Its main focus is to increase the resolution of images.

3. Cycle GAN: performs the task of Image Translation. Suppose we have trained it on a horse image dataset and we can translate it into zebra images.

And there are many more types and applications for the GANs

Now let's talk about the cGANs:

Its only difference from normal GANs is that it gets a condition as input, not just noise like in normal GANs, so knowing that we can explain why its loss functions look like this,

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

It's almost the same as the normal GANs loss function but we added the condition y so that we minimize/maximize the error given certain conditions.

This GANs Network used in the paper consists of Generator and Discriminator as follows:

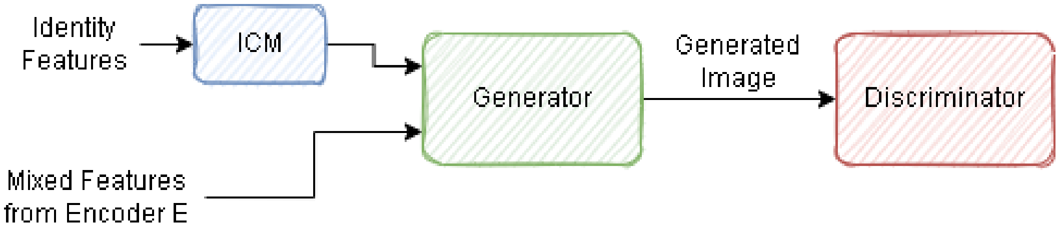


Figure 40

The generator is a CNN network with multiple Up sample layers and it generates the images and the discriminator is Patch-Discriminator that penalizes the framework for better visual quality.

The GANs in this paper is adapted from [40].

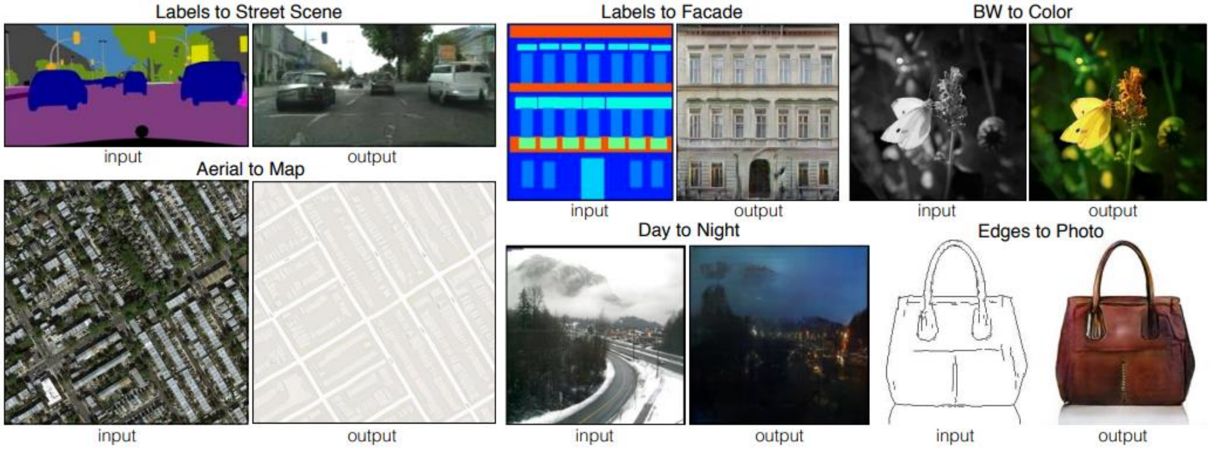


Figure 41 Examples of Image-to-Image translation using cGANS

And in this paper, it implements the normal cGANs loss function but they add an L1 loss for the generator so that it not just tries to trick the discriminator, it also tries to create an image that is close to a Ground truth so this increases the image quality and the new loss function for the image-to-image translation looks like this:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G).$$

where,

$$\begin{aligned}\mathcal{L}_{cGAN}(G, D) =&\mathbb{E}_{x,y}[\log D(x, y)]+\\ &\mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))]\end{aligned}$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1].$$

And that's it for the FAS part.

As mentioned before, we only used the Face recognition part, and here's why:

1. This feature isn't important for your problem (Finding missing children) as we don't care much about seeing the missing person at different ages.

2. It's Computationally expensive to train bot Face Recognition and Face Age Synthesis.

3. Even though this FAS error backpropagates to the Face recognition part (Age Estimation part) we found out that it didn't affect the accuracies of the Face recognition so there was no need to train the whole FAS just for that.

6. Datasets, Training Process and results

1. **Datasets**

We used SCAF and LCAF datasets to train the MTL model, they're both a subset of a larger dataset called m1sm which contains many celebrity images with their age and ID.

SCAF size is about 500k and LCAF 1.7M. And for testing, we used LFW, Age-dB, FG-NET, and CACD-VS

2. **Training Process**

We Trained the model on an Nvidia 2080TI GPU and it took about 20 hours for the SCAF dataset to finish training.

3. Results

	LFW	Age-dB	FG-NET	CACD-VS
SCAF (Our Training)	99.47%	95.83%	93.7%	99.05%
LCAF (Our Training)	99.58%	97.2%	95.4%	99.2%
SCAF(Paper)	99.52%	96.23%	94.78%	99.38%

As we can see from the results our training results are almost the same as the papers even though we only trained the MTL's Face recognition part only.

7. Problems we faced with this paper

1. We didn't know if this paper was suitable for our problem and can actually give the accuracies it promised in the paper, so we had to train it and test it ourselves which consumed too much time for many reasons which we'll discuss in the other problems.

2. Datasets used in this paper weren’t available, so we had to extract the datasets used in this paper (SCAF and LCAF) from the bigger dataset which is ms1m, and we also contacted the author of this paper to send the notation of the datasets used but he responded a bit late, at this time we finished training using the dataset we extracted from ms1m, this problem is discussed in details in Datasets Section.

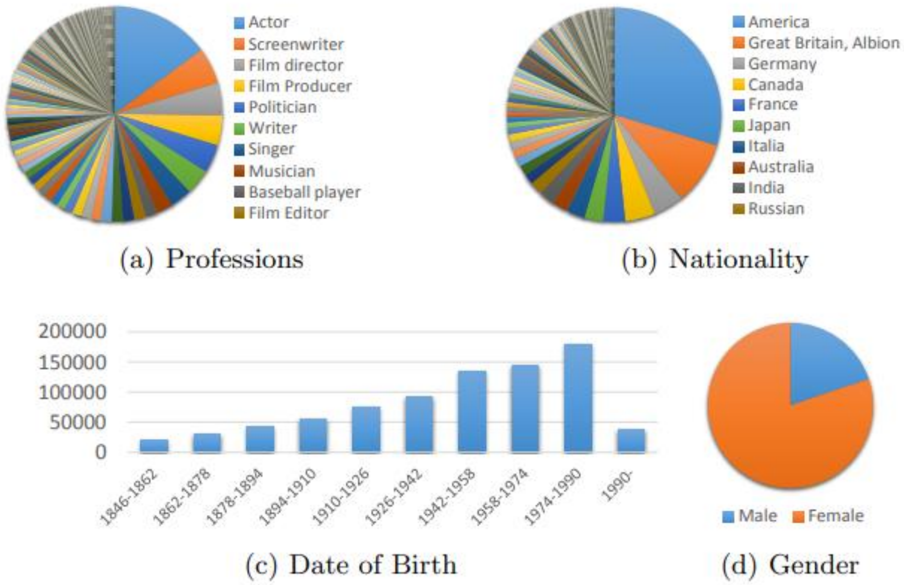
3. Code wasn't documented well, the author had written the model and shared it on GitHub so we tried to use it to train the model but there was no comment or anything to help us understand the code, so this consumed much time.

Datasets

Training datasets

MS-Celeb-1M

A dataset introduced by Microsoft [26] has 10 million images of the top 100 thousand celebrities selected from a list of one million celebrities depending on the appearance frequency. As shown in Figure 42, the data has gender imbalance which might be correlated with the professions' distribution. The data comes from more than 200 different countries where most celebrities are American.



(a) Professions (b) Nationality

(c) Date of Birth (d) Gender

Figure 42: MS-Celeb-1M statistics

MS1MV2

A semi-automatic refined version of the MS-Celeb-1M dataset was introduced by the authors of ArcFace [11] having 5.8 million images of 85 thousand Ids.

CAF (Cross-age face dataset)

A dataset introduced by the authors of OE-CNN [27] consists of more than 300 thousand images of 4.7 thousand Ids. The dataset was gathered by forming a list of celebrities from multiple sources like IMDB, Forbes celebrity list, and Wikipedia then iteratively using the Google search engine to collect images. The dataset has a large number of Asian celebrities to increase the diversity of the data. An age estimation module (DEX) [28] was utilized to add age annotations to the data.

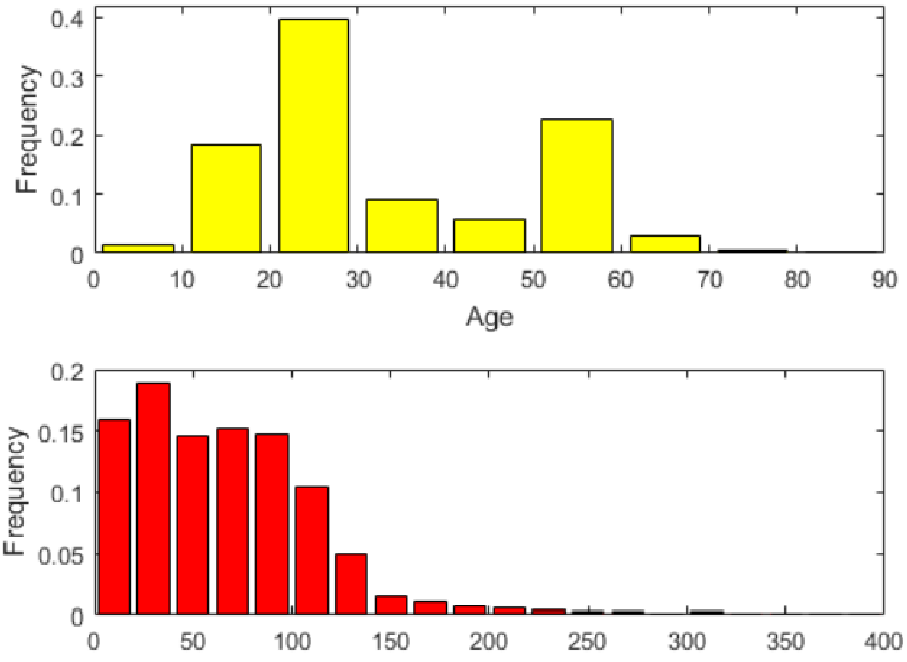
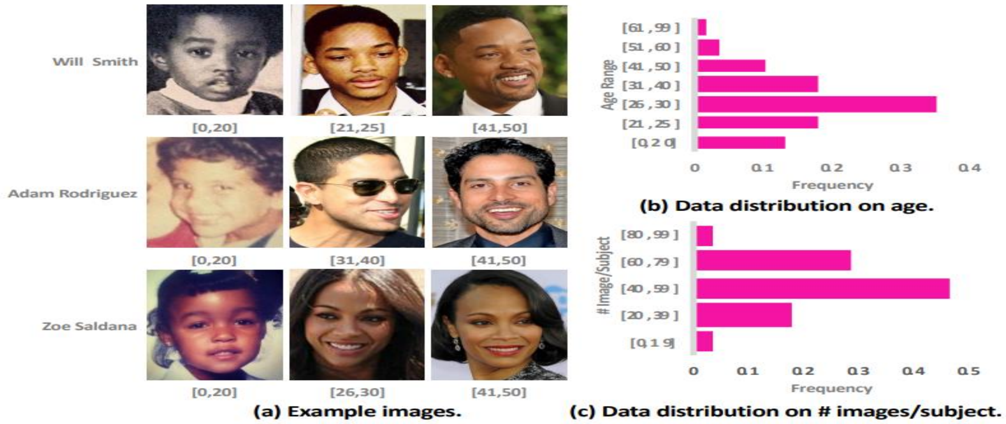


Figure 43: CAF statistics

CAFR (Cross-age face recognition dataset)

A recollection of the MS-Celeb-1M dataset by [30] using a subset from its list of names while taking into consideration gender balance, racial diversity, and that every individual has a lot of images at different ages. The data was passed to a face detector to remove images without faces. An age estimator and a landmark localization algorithm were used to add annotations to the data then these annotations were manually corrected by professional data annotators. The dataset contains 1.5 million images of 25 thousand Id.



(a) Example images.

(b) Data distribution on age.

(c) Data distribution on # images/subject.

Figure 44: CAFR statistics and examples

CASIA-Webface

Similar to the above datasets, the dataset was collected by forming a list of celebrities from the IMDb website then the images are collected [31]. The dataset contains 0.5 million images of 10.5 thousand Ids after cleaning.

VGGFace2

The dataset contains 3.3 million images of 9131 Ids with an average of 362 images per Id [32]. The dataset can be used for both training and testing having 500 Ids left out for testing. The data is approximately gender-balanced having 59.3% males. It also contains annotation of age and poses.

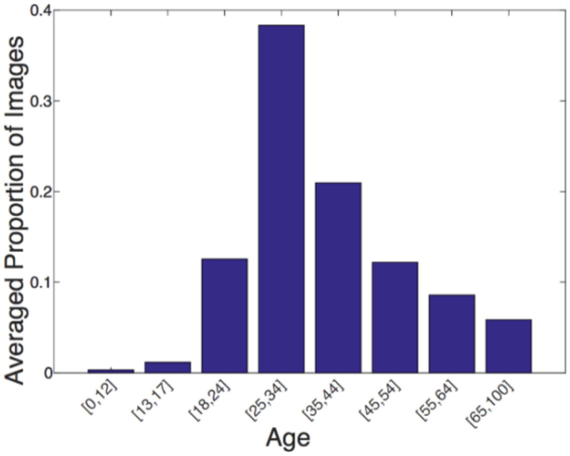


Figure 45: Age distribution of VGGFace2

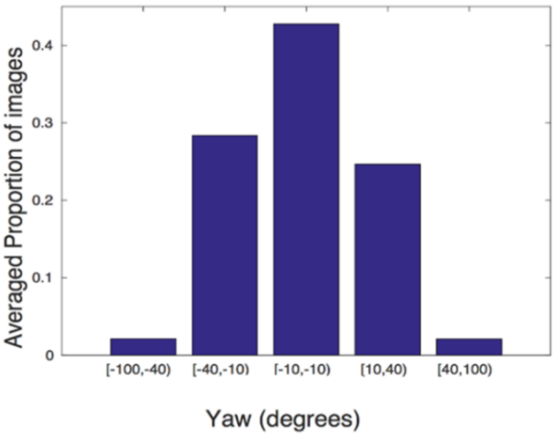


Figure 46: Pose distribution of VGGFace2

SCAF and LCAF (small and large cross age face dataset)

They are annotated subsets of the ms1mv2 dataset introduced by the authors of MTLFace [33]. The sizes of the datasets are 0.5 million and 1.7 million for SCAF and LCAF respectively. Age and gender annotations were added using the public Azure facial API then they were manually corrected by the authors. The Age distribution of the SCAF dataset shows that it has more images for small ages compared to other datasets used in other works which are very helpful for the model.

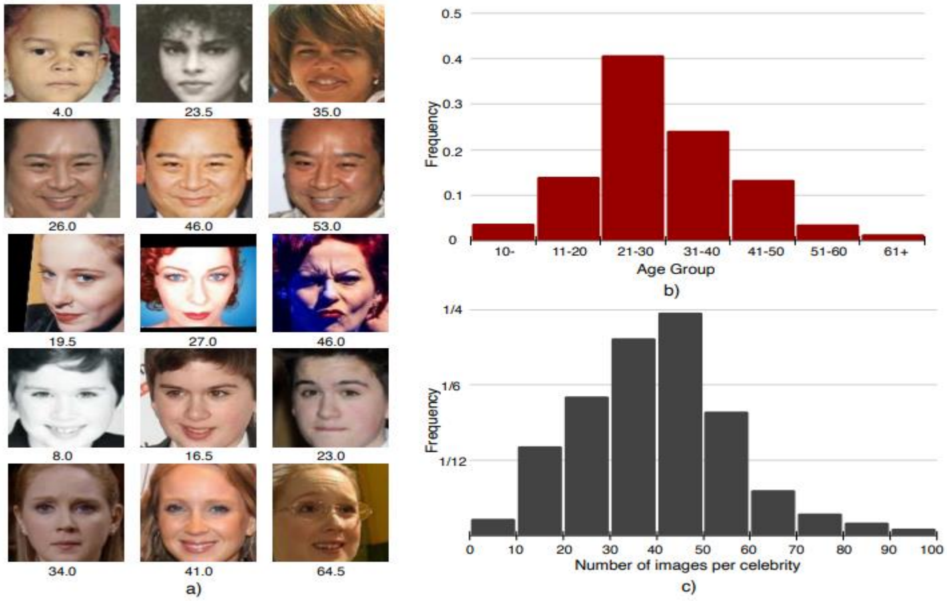


Figure 47: SCAF statistics and examples

Other datasets

There are also other non-popular datasets used in other works like VGGFace, Deepglint, Celebrity+, and Google.

The following table summarizes information about training sets:

Dataset	#images	#subjects	Average #images/subject	Used by	Main advantage	Public
MS-Celeb-1M	10M	100K	100	-	-	Yes
MS1MV2	5.8M	85K	-	ArcFace	-	Yes
LCAF	1.7M	24K	-	MTLFace	SOTA	Yes
SCAF	0.5M	12K	40	MTLFace	SOTA	Yes

CAFR	1.5M	25K	57.86	AIM	Gender-balance	No
CAF	0.3M	4.7K	80	DAL,OE-CNN	Racial diversity	No
CASIA-Webface	0.5M	10.5K	46.8	DAL,OE-CNN	-	Yes
VGGFace	2.6M	2.6K	1000	DAL,OE-CNN	Bigger #images /subject	Yes
VGGFace2	3.3M	9.1K	362.6	ArcFace	Variation of poses	Yes
Celebrity+	0.2M	10K	20	DAL,OE-CNN	-	Yes
Deepglint	2.8M	94K	-	-	Asian data	Yes
Google	200M	8M	-	Google	Largest dataset	No

Testing datasets

There are two types of testing in facial recognition:

Face verification

Pairs of faces are constructed having the number of positive pairs equal to the number of negative pairs. The accuracy is measured by the percentage of correct predictions of whether the pairs are positive or negative.

Face identification

The test is done by constructing probe and gallery sets where a loop over every image in the data is considered a probe set and other images are the gallery set where a search happens over the gallery to find the top match/matches to the probe image. Rank-k accuracy is the percentage of probe images that had a correct match in the galley in the top-k matches found by the model. All the following works report the rank-1 accuracy over the test sets. Some tests add some external data to the gallery set to make the searching process more challenging. The external data is called a distractor set in this case and the more we increase the distractors the more the identification accuracy drops. Some examples of using distractors are FG-NET and Facescrub MegaFace [35] challenges where MegaFace is used as a distractor and added to the galleries of FG-NET and Facescrub datasets.

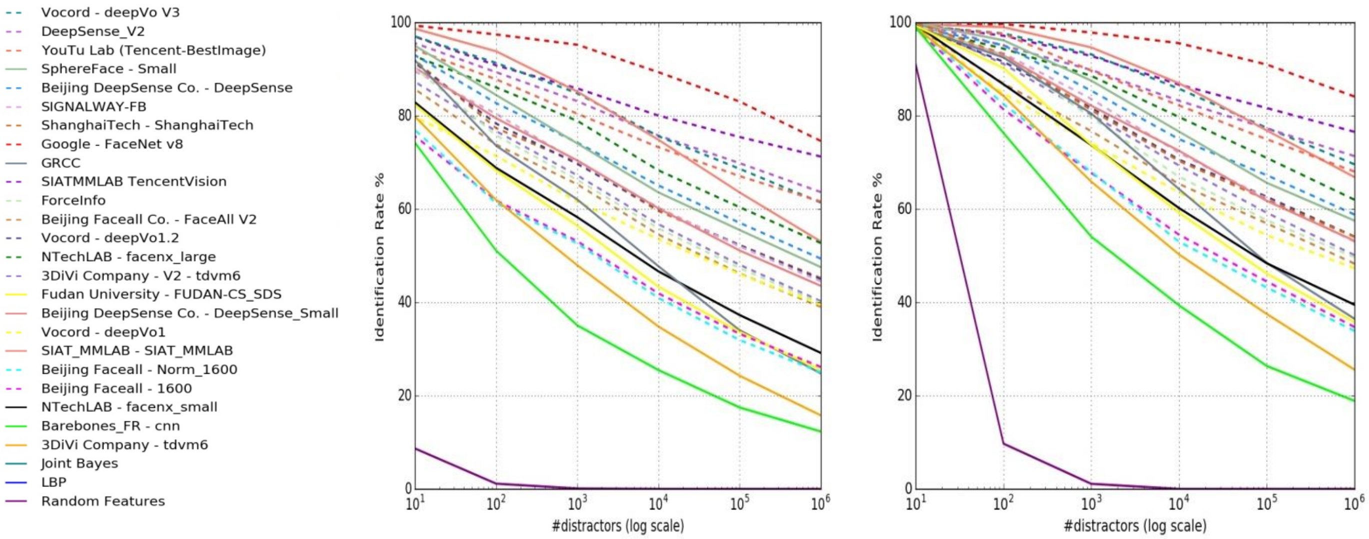


Figure 48: Accuracy on FG-NET MegaFace Challenge I against number of distractors

In the following table we summarize information about test sets:

Dataset	SOTA score (%)	SOTA paper	Training set size	Pair-size	#distractors	Public
CACD-VS	99.55	MTLFace	Large	2000	-	Yes
Age-dB	96.23	MTLFace	Small	3000	-	Yes
CALFW	95.62	MTLFace	Large	3000	-	Yes
LFW	99.52	MTLFace	Small	3000	-	Yes
Morph II	99.4	DAL	Large	-	0	No
MF1-Facescrub	77.58	DAL	Small	-	1M	No
FG-NET (leave one out)	94.78	MTLFace	Small	-	0	Yes
FG-NET (MF1)	57.92	DAL	Small	-	1M	No
FG-NET (MF2)	60.01	DAL	Large	-	1M	No

Experiments

Small Dataset Training

The model was trained on the SCAF dataset which has 0.5 million images. The accuracy of the model improved over some stages where every stage adds some enhancement to the accuracy and these stages were:

- Disabling Automatic mixed precision in training.
- Using a manually corrected annotation file used by the author.
- Using gradient accumulation to achieve an equivalently large batch size (512) as used in the paper.

Accuracies over these stages are reported along with the accuracy achieved by the author on the two datasets LFW and Age-dB in the following graph:

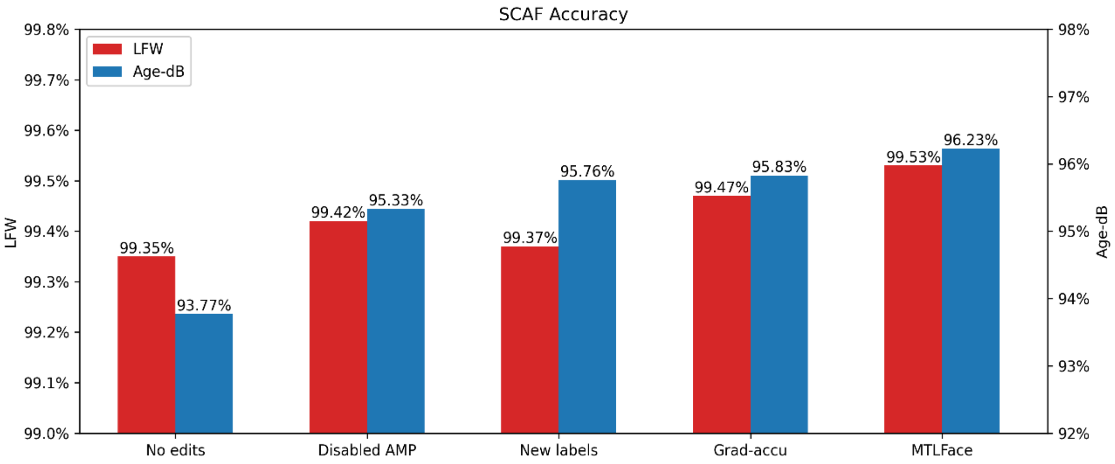


Figure 49: Accuracy of SCAF in different stages

The accuracy is at its closest to the authors at the final stage. The difference between the two accuracies is due to the author's reporting of the best accuracy over each test set individually while we report the accuracy of an intermediate model that achieves a balanced score on the test sets.

Large dataset training

Figure 50 shows the improvement added by increasing the training set size using the LCAF dataset has 1.7 million images. The accuracies are reported on LFW, Age-dB, FG-NET, and CACD-VS test sets.

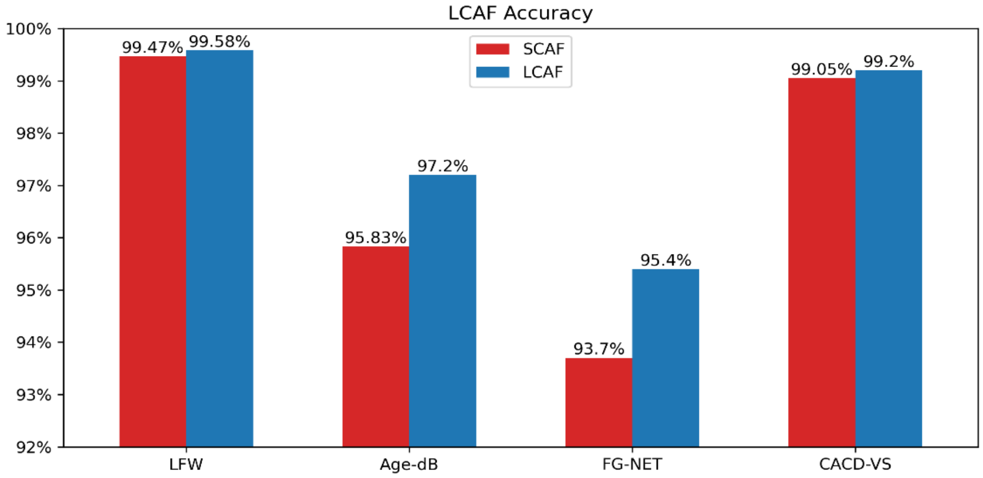


Figure 50: Accuracy of LCAF Vs. SCAF

False predictions of the model

In the following figures, we show a sample of the error of the model on each test set

LFW

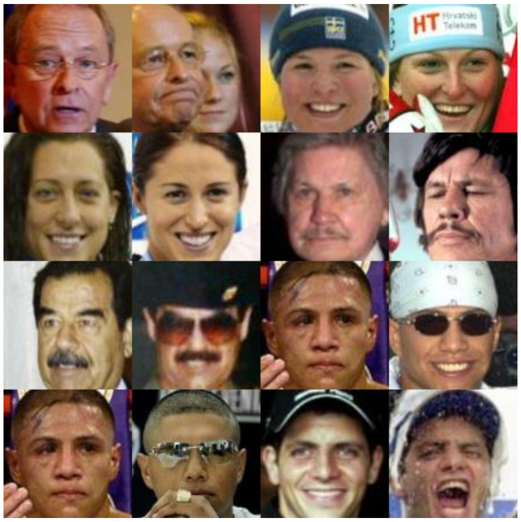


Figure 51: False negatives of LFW

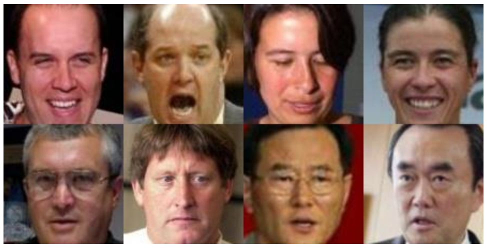


Figure 52: False positives of LFW

FG-NET

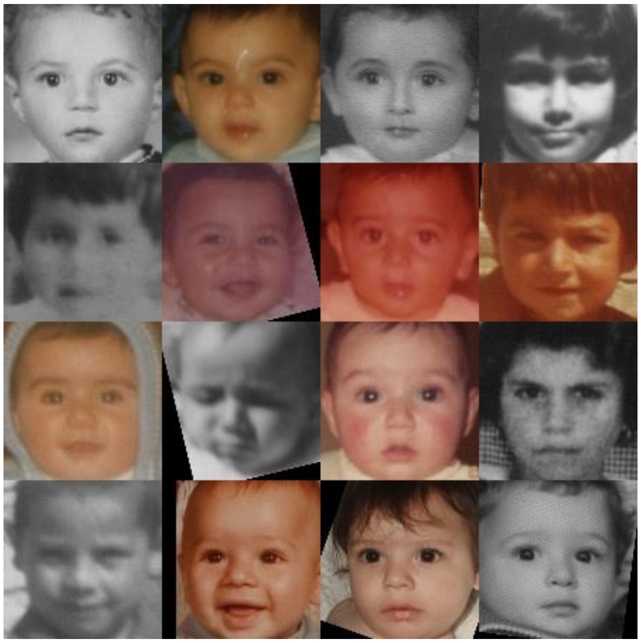


Figure 53: Failed probe identification of FG-NET

Age-dB

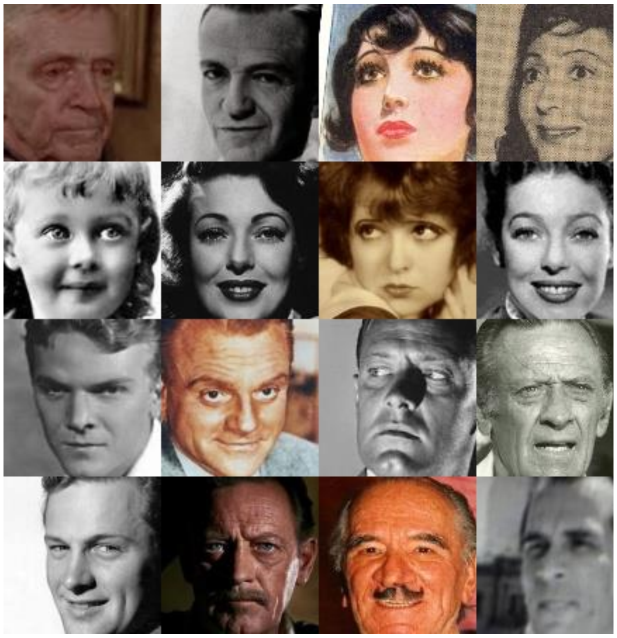


Figure 54: False negatives of Age-dB

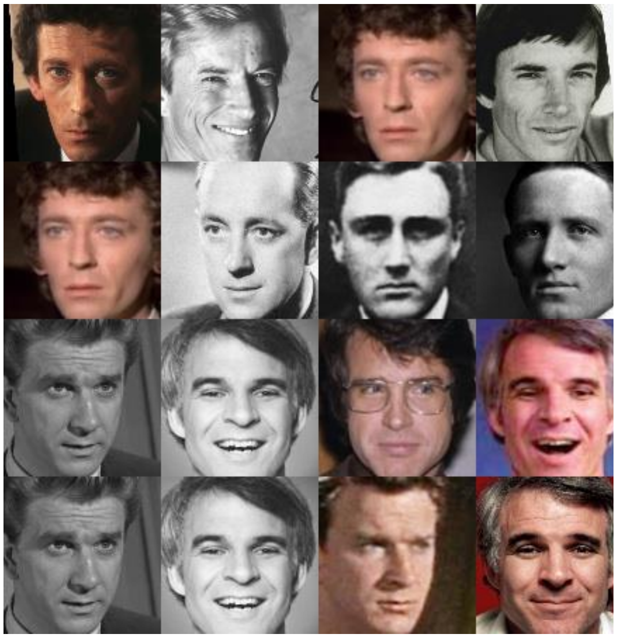


Figure 55: False positives of Age-dB

CACD-VS

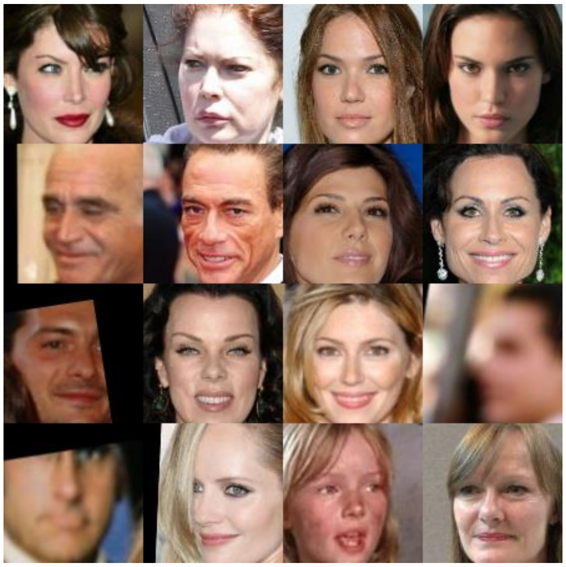


Figure 56: False negatives of CACD-VS

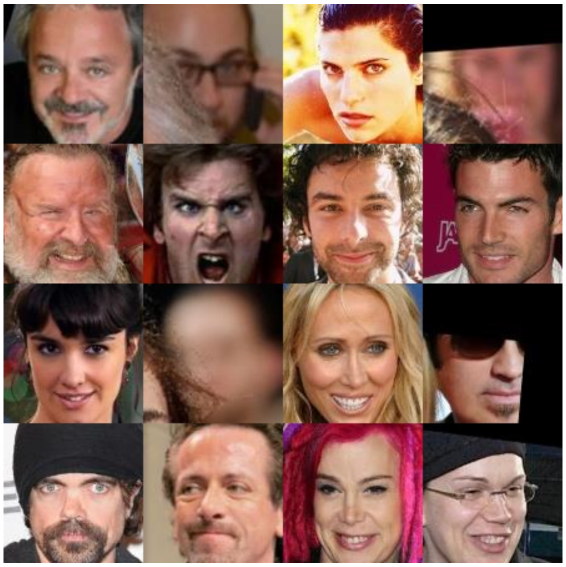


Figure 57: False positives of CACD-VS

As we can see there are four significant problems that cause the model not to work very well:

1. Face occlusion
That's more clearly seen in LFW false negatives where the face in the image has an occlusion such as a hat, glasses, or sunglasses.

2. Young children
As FG-Net is the only test set that contains small children, it shows that it's hard to do face recognition for young ages, more commonly ages of less than two years.

3. Face pose variation, which is more common in Age-dB errors.

4. Unclear faces
Having blurred or low-quality face which is clearer in CACD-VS errors.

Super Resolution Training

To help model performance on low-quality and blurred images, we used a super-resolution GAN on the test set images. However, accuracy doesn't improve as the super-resolution GAN doesn't preserve the identity of the recovered face.

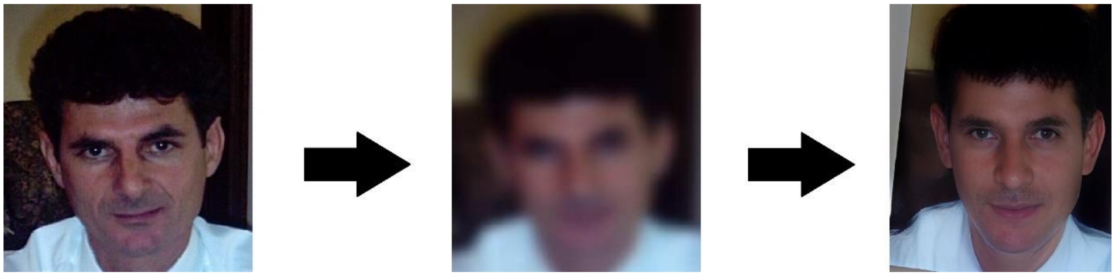


Figure 58: Identity preservation of super-resolution GAN

We also tried to train a model on images enhanced by super-resolution to force the training dataset to have the same features as the test set. Accuracy improved on most test sets. However, it was still worse than the non-enhanced images.

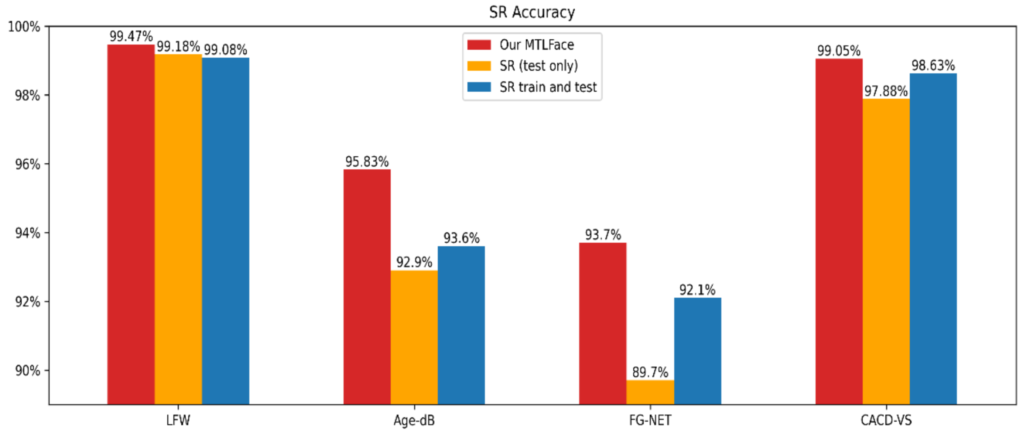


Figure 59: SR accuracy

Face Rotation and De-occlusion

To enhance the model's performance on images of faces with occlusion or pose problems, we use a face rotation and de-occlusion GAN named CFR (complete face recovery). CFR can't be used directly with the model as it adds some distortions to the generated face which makes it harder to recognize it and makes the accuracy worse.

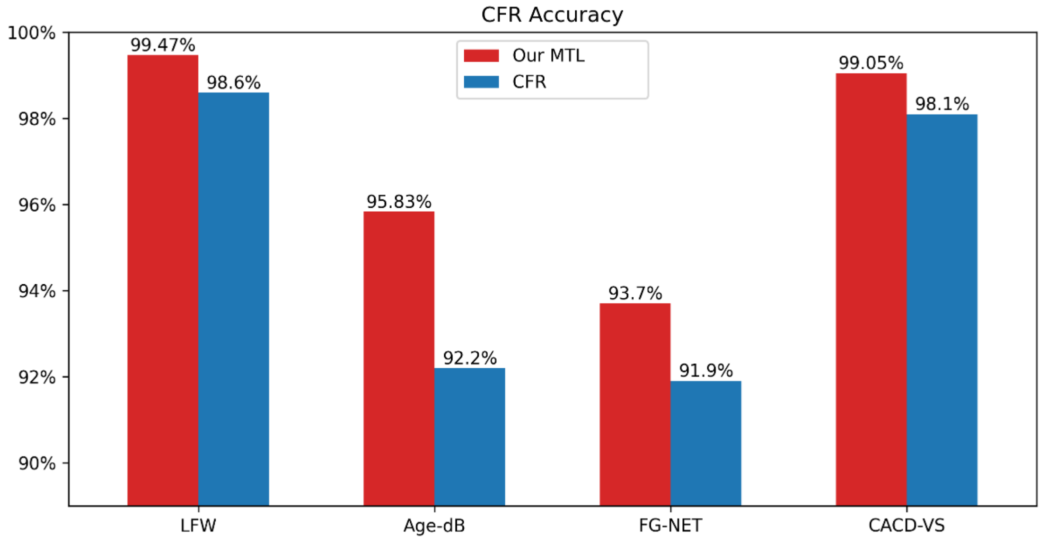


Figure 60: CFR accuracy

Instead, it's used in one path of the model while another path uses normal images. This helps add some improvement and the percentage of removed error is reported in the following graph.

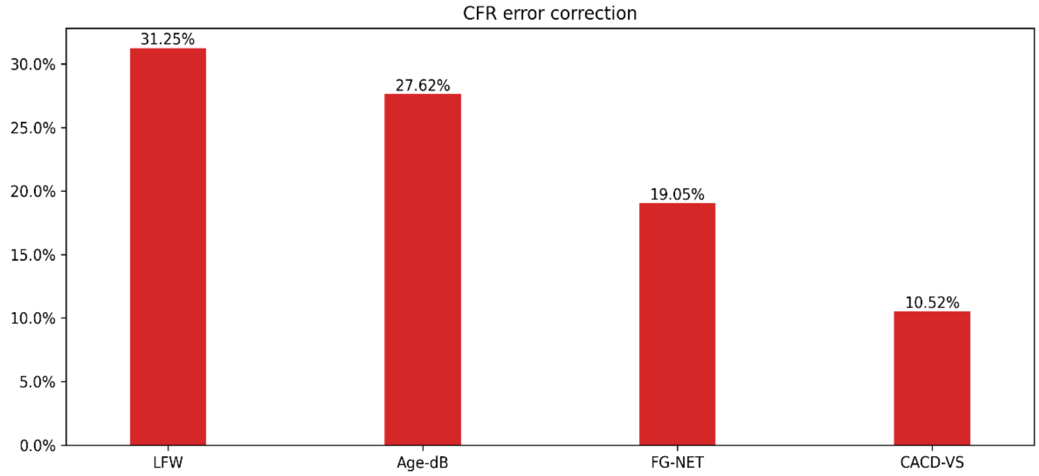


Figure 61: Percentage of images corrected by CFR

Problems with CFR model

As seen in Figure 62, the model can deal with face pose and occlusion problems but the model doesn't perform well on faces with sunglasses as shown in Figure 63.

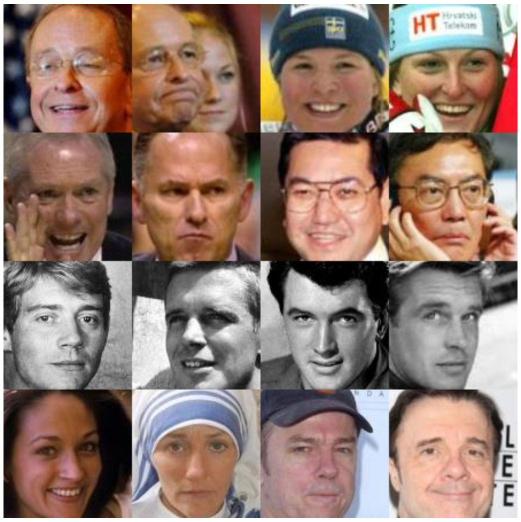


Figure 62: Sample of images corrected by CFR

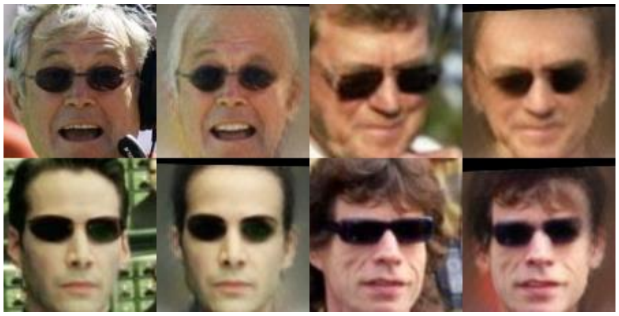


Figure 63: CFR performance on sunglasses

Egyptian Dataset

We collect an Egyptian dataset for testing the model's performance on our race. The dataset was tested while being collected where it had 231 Images of 42 Ids labeled as "set 1". After data collection was finished the number of young children increased by a large amount which caused the accuracy to drop. The data has 400 images of 84 persons labeled as "set 2". We report the identification accuracy of ranks 1 and 5. We also report the verification accuracy of random pairs from the dataset. The accuracies of the two sets show that there is no significant dependence of race on the model's performance and the low accuracies obtained in set 2 are due to having a larger ratio of young children with respect to other face recognition test sets.

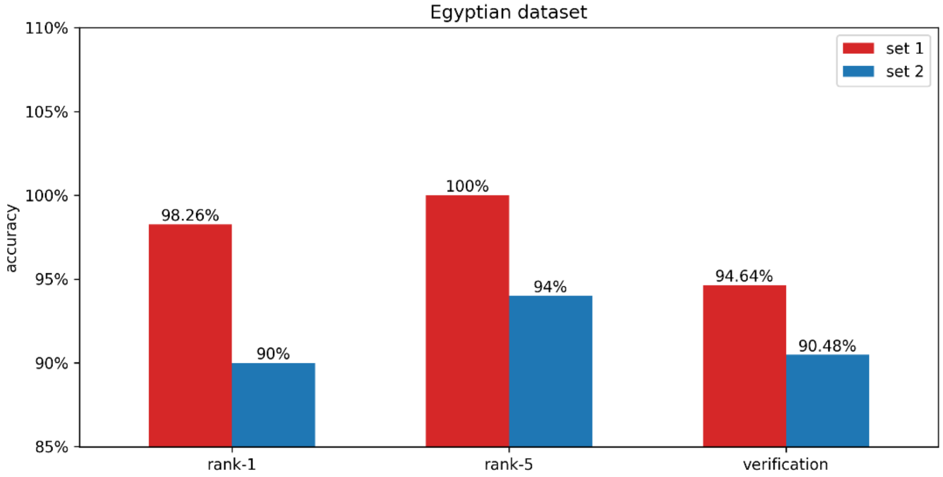


Figure 64: Egyptian dataset accuracy

Children Only Dataset

We needed a good representation of how the model performs on children, so we made a subset of the FGNET dataset consisting of 708 images of 81 children. We report the accuracy of the models trained on SCAF and LCAF datasets. We also report the accuracy of a model trained only on children’s images consisting of 0.5 million images with ages of less than 25.

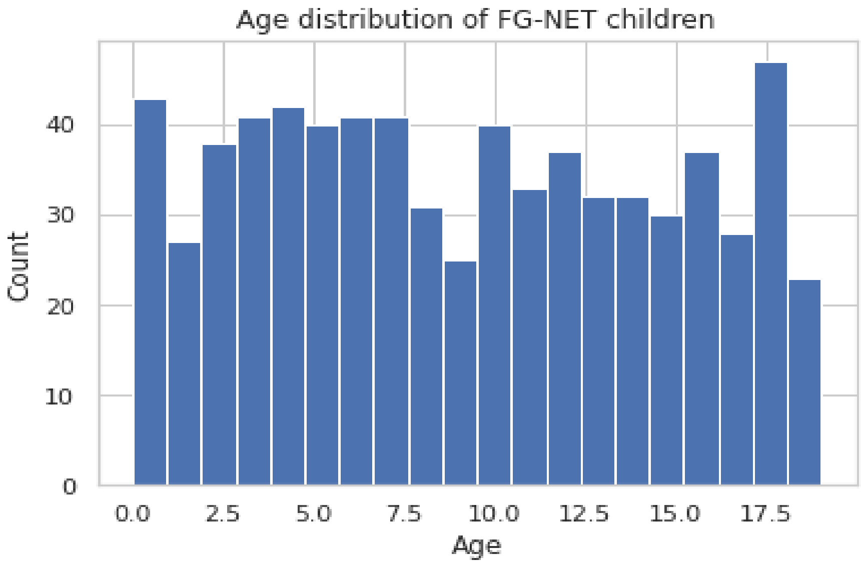


Figure 65: Age distribution of FG-NET children

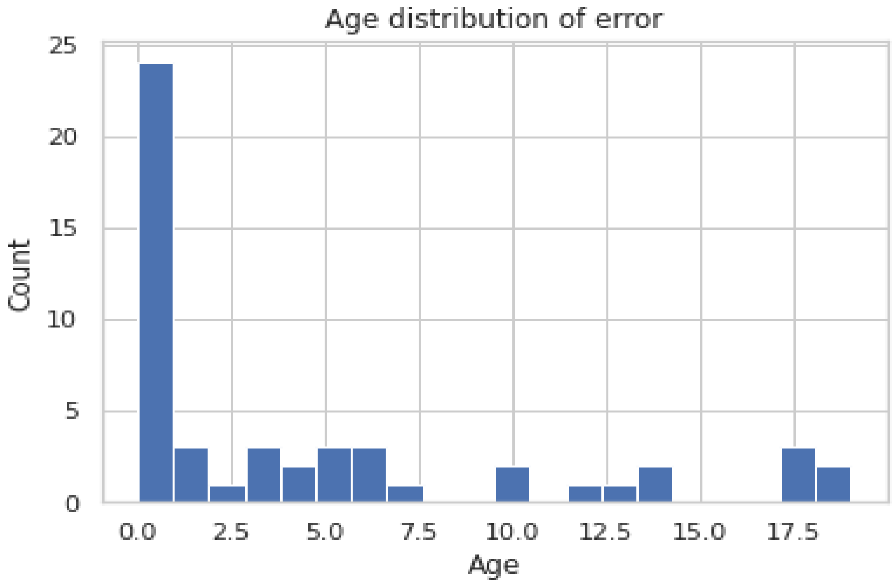


Figure 66: Age distribution of error of FG-NET children

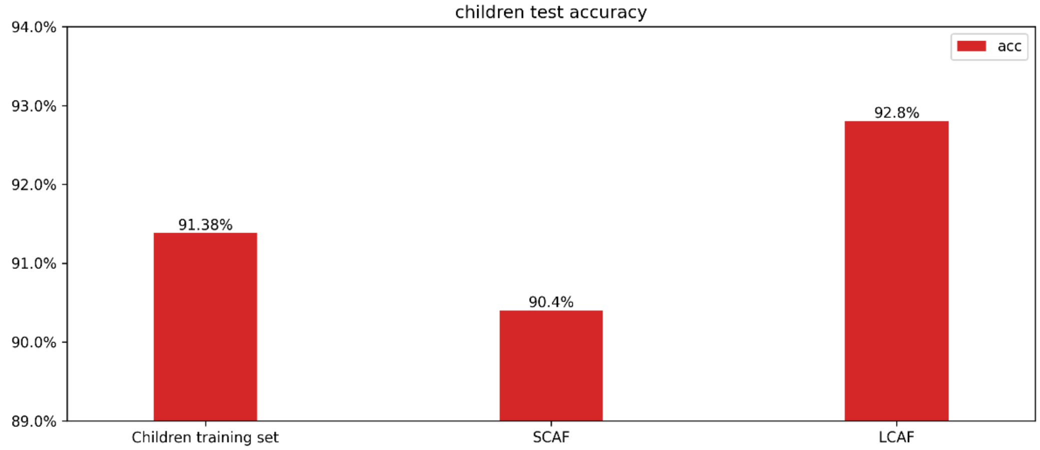


Figure 67: Accuracies on FG-NET children by models trained on SCAF, LCAF and LCAF children

We notice that the model trained on children only outperforms SCAF which has the same size which means that increasing the percentage of children on the training set improves the model's performance. We also notice from the age distribution that the most error happens at young ages (less than 2 years).

Pipeline

In this part, we will talk about the final pipeline for our project, and the following is a diagram explaining the idea and then we'll show how we implemented it using the models we talked about before.

Description

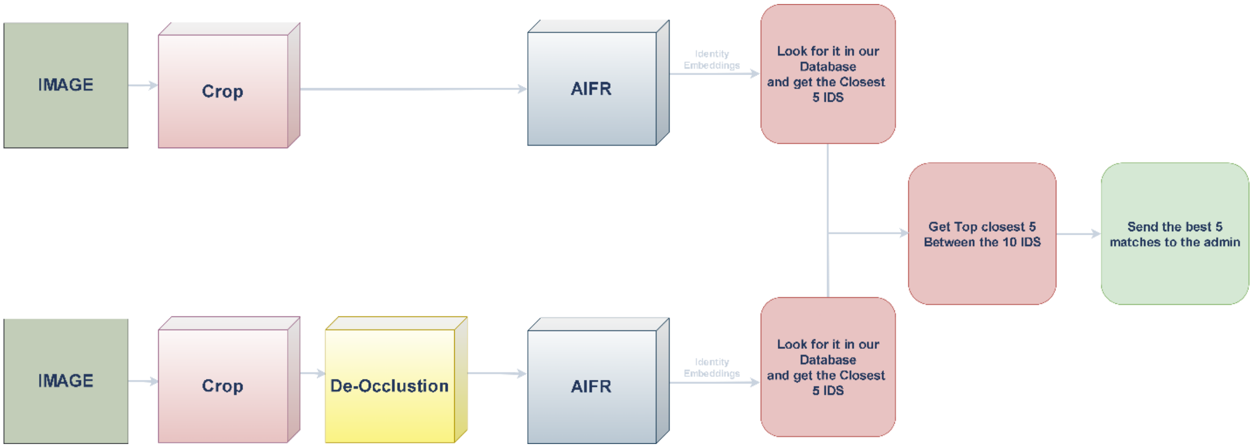


IMAGE
Crop
AIFR
Identity Embeddings
Look for it in our Database and get the Closest 5 IDS
Get Top closest 5 Between the 10 IDS
Send the best 5 matches to the admin
IMAGE
Crop
De-Occlustion
AIFR
Identity Embeddings
Look for it in our Database and get the Closest 5 IDS

Figure 68

As shown in Figure 68, it's consisting of 2 branches working together as an ensemble the only main difference is the De-Occlusion in the bottom branch and its job is to remove Occlusion from faces (like glasses or anything on the face).

Starting from the top branch, it does the following:

1. Get an image from the application (a missing person/or suspected to be missing).

2. Crop the face from the image using **MTCNN**.

3. Input it to the AFIR model (**MTL**) that's outputting an embedding for this person and this MTL is trained on the LCAF dataset.

4. Find the closest 5 Identities to this person using the embedding we generated for this person and compare it to the embeddings we have in the database.

And that's it for the first branch, for the second it does the following:

1. Get an image from the application (a missing person/or suspected to be missing).

2. Crop the face from the image using **MTCNN**.

3. Input it to the De-Occlusion model (**CFR**) to remove any occlusions on the face.

4. Input it to the AFIR model (**MTL**) that's outputting an embedding for this person but the difference here is that this MTL with different weights than the one in the top branch as it's trained on the SCAF dataset passed through the CFR model.

5. Find the closest 5 Identities to this person using the embedding we generated for this person and compare it to the embeddings we have in the database.

Now after returning the 5 closest identities from each branch, we then find the closest 5 from these 10 identities then we send it to the admin to choose and confirm the closest identity to the input image (person).

Also, we can see that there's a human involved in this process as this is a critical problem and we need to make sure that the matches (the input person and the closest identity in the database) are accurate 100% before taking any actions.

The pipeline after replacing each function with the model is shown in Figure 69.

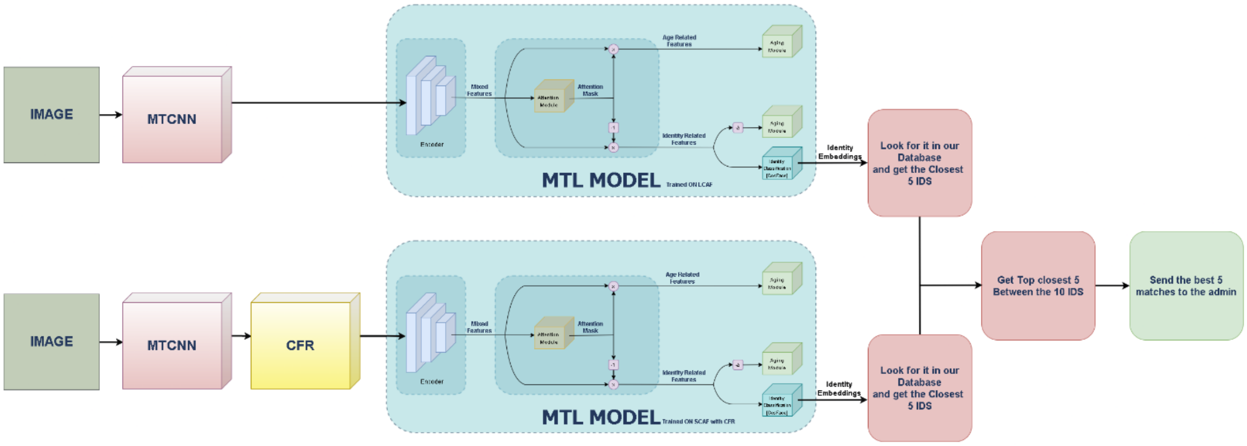


Figure 69

Training models

1. **MTCNN**, we used a pre-trained model.

2. **CFR**, we used a pre-trained model.

3. **MTL in the top branch**, we trained it on LCAF as it gave us the best accuracies when we were testing different datasets on it.

4. **MTL in the bottom branch**, we trained it on SCAF but we passed It through the CFR so that it gives better accuracies when using CFR.

Technologies used for the mobile and website application

Front End - Flutter

Flutter is an open-source UI software development kit (SDK) created by Google. It is used to develop cross-platform applications for Android, iOS, Linux, macOS, Windows, Google Fuchsia, and the web from a single platform-agnostic codebase. And It's one of the simpler, faster, and cheaper ways to get a product to market.

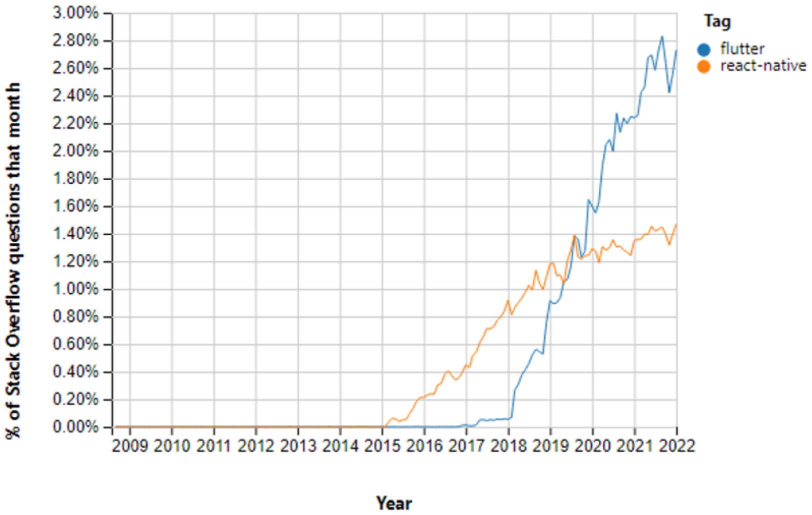


Figure 70

Since its release in 2017, it quickly became the developers' go-to framework and has recently surpassed React Native to be on the top of the list of mobile application development frameworks.

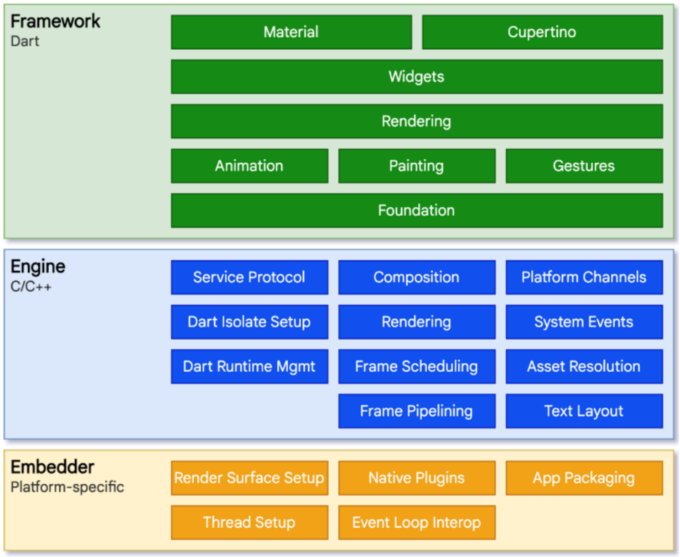


Figure 71

As shown in Figure 71, Flutter has three layers that comprise its structure, they are the framework that's built using Google Dart programming language and it's also the interface for the developers, the underline engine that's built using C/C++, and the embedder that's specific to the platform running the application and it uses the Skia library for the graphic capabilities.

There are a lot of reasons why Flutter has become the number one choice for developing a new mobile application, here are the majors 10 advantages for which we use Flutter

1. Open source
It allows for issue posting and easy access to the open documentation by the open developer forums and enables Flutter newbies to learn from and evolve alongside the community of developers who actively support the platform. And thus, making the programmers more effective and productive, which reduces the project's overall time and expense.

2. Single codebase
As the framework is cross-platform, developers can write the code once and use it for all the platforms, instead of writing a specific program for each platform as with native frameworks. As a result, the overall cost of creating and releasing the software is significantly reduced.

3. Dart as a programming language
Flutter is built using the Dart programming language. Dart, which is an object-oriented language, is a lot like Java and uses a lot of popular features of other languages too such as rich standard libraries, garbage collection, strong typing, generics, and async-awaits. This all helps developers build the application with ease.

4. Hot reload and development
With hot reloading, developers can inject new file versions and updates to the code while the app is running. This way, developers and designers don't need to wait much to see the changes and therefore they can act accordingly immediately which boosts productivity and reduces the time of development.

5. Native app-like performance
Flutter applications can run fast on all platforms compared to other app development platforms. This is because it is built using the Dart programming language which is fast, simple, and compiles into native code easily.

6. Tech community
There is a strong development community that consistently works to improve Flutter. They facilitate framework learning for newcomers. Anyone can easily begin creating an app with enormous community resources.

7. Use of custom widgets
Flutter provides many ready-to-use widgets that developers can use while building the application. Those widgets can even be wrapped inside another to enable more complex functions.

8. Attracts More Investors
As flutter enables only a single codebase to build the application on multiple platforms with a high-quality user experience and performance, this can bring investors from all platforms quickly and easily to fund the project.

9. Creating an application for different platforms
Using Flutter can allow hiring only one developer instead of many developers for the many platforms which reduce the cost dramatically.

10. Less testing
With Flutter, all the testing needed is done on one platform just once Instead of testing the software implementations for every platform.

Databases

A database is a collection of data stored in a computer that can be accessible in various manners. Usually, this data is structured in a way that makes the data easily accessible.

All projects that deal with data need a database system that can store all types of data, access required data quickly, and get instant insights to make strategic business decisions.

There are mainly two types of databases: relational databases and non-relational databases.

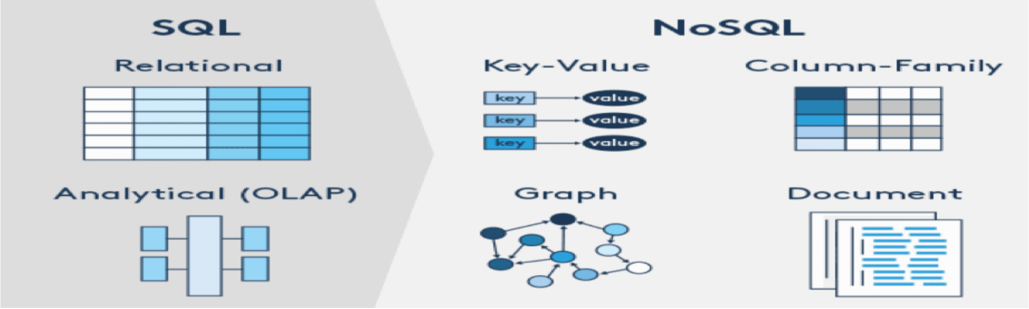
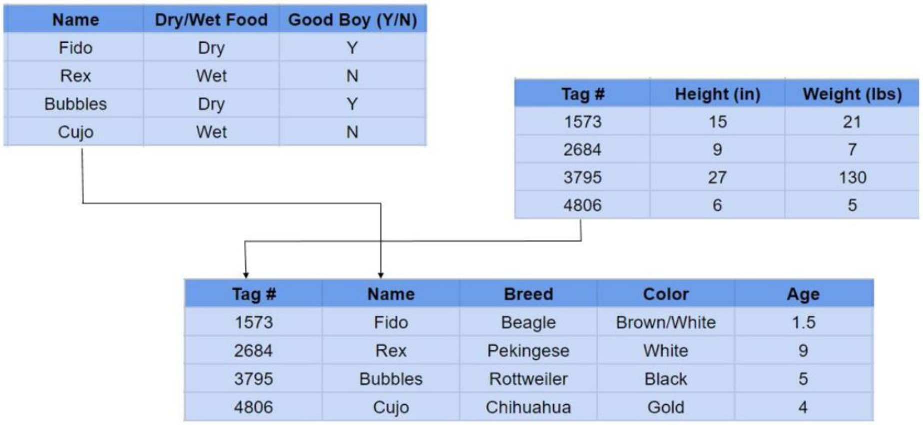


Figure 72

Relational database

In a relational database, or a relational database management system (RDBMS), information is stored in tables. These tables often have shared information with one another, which results in the establishment of relationships between them. From this, a relational database derives its name. Each table is concerned with one class we gather information about. And each column of the table describes a feature of that class and often is called a field. And each row represents the actual data for the different instances of that class and often is called a record.

Figure 73 shows an example of a relational database. It consists of three tables, each one has a unique name and is concerned with specific information, and each one is connected to the other hance form a relation.



Name	Dry/Wet Food	Good Boy (Y/N)
Fido	Dry	Y
Rex	Wet	N
Bubbles	Dry	Y
Cujo	Wet	N

Tag #	Height (in)	Weight (lbs)
1573	15	21
2684	9	7
3795	27	130
4806	6	5

Tag #	Name	Breed	Color	Age
1573	Fido	Beagle	Brown/White	1.5
2684	Rex	Pekingese	White	9
3795	Bubbles	Rottweiler	Black	5
4806	Cujo	Chihuahua	Gold	4

Figure 73 [23]

The relationship between the tables and the fields defined in the records is called a schema. For relational databases, the schema must be precisely defined and fixed, i.e., each record that needs to be recorded in the table must follow the defined schema.

Relational databases are also called Structure Query Language (SQL) databases. As SQL is the most common way of interacting with relational database systems. Developers can write SQL queries to retrieve data, and edit data by updating, deleting, or creating new records.

Popular relational/SQL databases

SQL Server

Developed by Microsoft and offered by multiple editions with varying features to target different users.

MySQL

Free, open-source, and one of the most popular RDBMS worldwide. It's used by many high-traffic websites

PostgreSQL

Another free and open-source RDBMS. It can handle complicated data workloads due to Its wide range of extension functions

Advantages of relational databases

1. Data accuracy
The relations between tables follow some rules that ensure there is no duplicated information. This, in turn, simplifies the queries and reduces storage costs.

2. Normalization
Data is organized in such a way that ensures that data anomalies are reduced or eliminated. This consequently also lowers the cost of storage.

3. Simplicity
There are many tools and resources available to help get started as RDMS has been around for so long.

Disadvantages of relational databases

1. Scalability
It scales vertically, as historically it was intended to be run on a single machine. This means that if the data size or the number of accesses increases, we will have to improve the hardware of the machine, which of course has some celling and can be incredibly expensive.

2. Flexibility
As the schema is fixed (the relationships between tables and the fields and the records and their restrictions), making changes to the structure of the data is very complex. To add a record with a new field or a new data type, all the existing records must be edited, which takes time and makes the database offline temporarily.

3. Performance
The performance is highly linked with the complexity of the database. i.e., the number of tables and the records and fields stored in them.

Non-relational databases

Non-relational database, also known as NoSQL (Not Only Structure Query Language) as it can SQL and other types of query languages, is any kind of database that does not use the tables, records, and fields in a structured way like relational databases. Instead, it has a storage model that is tailored to the kind of data it is storing. It also has now schema, meaning there is flexibility to insert data with different shapes and fields.

The four different types of NoSQL databases.

1. Key-value databases
This is the most basic type of database, where information is stored in two parts: key and value. The key represents the location where data is stored and then is used to retrieve the value which holds the actual data.

Think of it as a dictionary. Its simplicity is its advantage as everything is stored as a unique key and a value, which allows for fast reading and writing. However, its simplicity restricts the data type stored in the value and hence cannot deal with complex data.

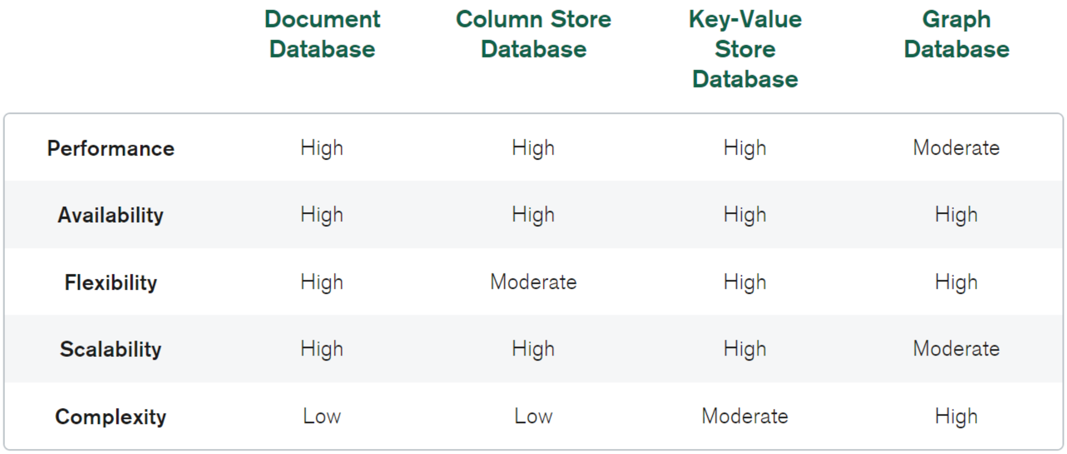
2. Document databases
Document databases store data in documents, which are usually JSON-like structures called Binary JSON (BSON) that support a variety of data types such as strings, numbers like int, float, and long, dates, objects, arrays, and even nested documents. The data is stored in pairs, similar to key/value pairs.

To query the data, we use the Mongo Query API (instead of SQL as in the relational databases). Because the document is represented in a JSON-like manner, they can nicely map to objects in object-oriented programming languages making them much easier for the user to read and work with.

3. Wide-column databases
This database uses tables, records, and fields, but unlike a relational database, the names and types of the fields can vary from record to record in the same table.

4. Graph databases

This database is the most specialized of the non-relational database types. They use nodes to store data, and the edges between the nodes to specify the relationship.



	Document Database	Column Store Database	Key-Value Store Database	Graph Database
Performance	High	High	High	Moderate
Availability	High	High	High	High
Flexibility	High	Moderate	High	High
Scalability	High	High	High	Moderate
Complexity	Low	Low	Moderate	High

Figure 74

Popular non-relational/NoSQL databases

MongoDB

MongoDB is a source-available cross-platform document-oriented database developed by MongoDB Inc.

We use this database in our project, so, we will discuss it in detail later.

Redis

Remote Dictionary Server (Redis) is an open-source key-value database that supports many kinds of data structures such as strings, lists, maps, sets, sorted sets, and more.

The distinction between relational and non-relational databases can be summed up as follows: relational databases store data in rows and columns like a spreadsheet while non-relational databases use a storage model (one of four) that is best suited for the type of data it is storing.

When to use a relational database

Relational databases remain the ideal option for developing a project where the data is predictable in terms of its structure, size, and frequency of access, and not likely to change.

When to use a non-relational database

If the data is not predictable and cannot naturally fit in a table structure and need to have flexibility in terms of shape and size maybe change in the future. It’s also a good choice for AI- and IoT-based applications that deal with a lot of data and therefore need superior performance and scaling capability.

MongoDB

Why do we use MongoDB and non-relational databases in general?

1. Scale Cheaper
There are two types of scaling a database; vertical scaling, and horizontal scaling.
In vertical scaling, as in SQL, we add more hardware to the same server. And when the server reaches its limits, the database is migrated to another server. As shown in Figure 75.

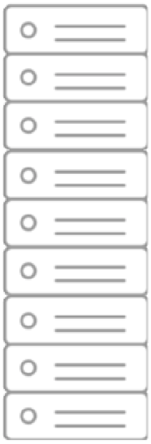


Figure 75

Figure 76

Problems arise with this approach

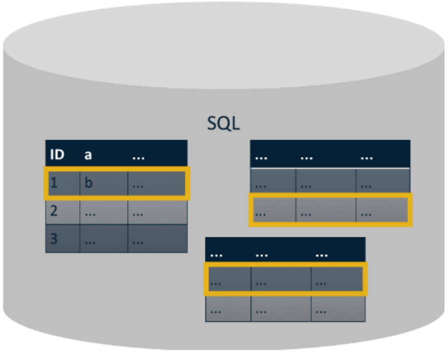
- One large server tends to be more expensive than two smaller servers with the same total capacity.
- Large servers may not be available due to cost limitations, cloud provider limitations, and technology limitations.
- Migrating to a larger server may require application downtime.

But MongoDB does not have this limitation as it is designed to scale horizontally through sharding. Sharding is a technique that split the data into small partitions and distributes those partitions over many small servers. As the database continues to grow, we can distribute it again over more servers. The benefit is that these new servers don't have to be bulky, expensive machines, which in turn, are cheaper. Also, there is no need for downtime.

2. Query Faster

Queries with MongoDB are typically faster than with SQL. As SQL stores the data into tables that are connected through relations as shown in Figure 77. Whenever we want to query a record, the tables must join together first which is very expensive for complex data. But when using MongoDB, data that’s related and accessed together is stored together

But when using MongoDB, data that are related and accessed together are stored together. And hence, most queries will not require joining.

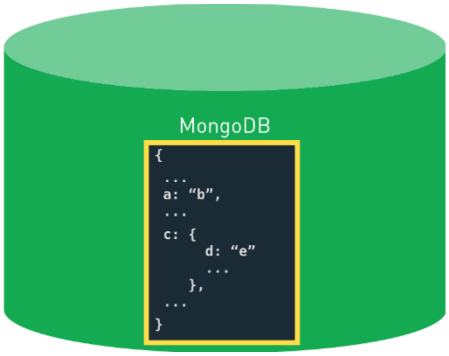


ID	a	...
1	b	...
2	...	...
3	...	...

...	...	...
...	...	...
...	...	...

...	...	...
...	...	...
...	...	...

Figure 77



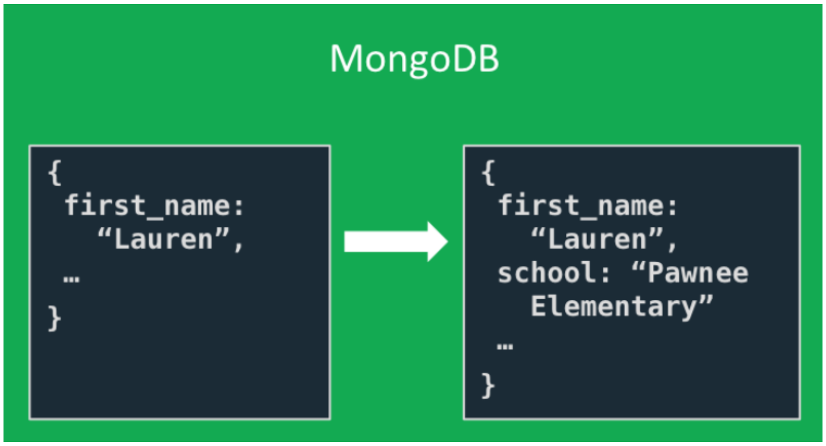
```
{
 ...
 a: “b”,
 ...
 c: {
      d: “e”
      ...
    },
 ...
}
```

Figure 78

3. Dynamic schema

Many applications change their requirements after the project has been designed and even after being deployed. So, the underline database management system must respond to and support those changes quickly. Adding one more field for the newly inserted records in SQL requires all the stored records to be updated. Which depends on the size of the data, may take hours, and require application downtime.

But in MongoDB, we can easily change the shape of the data as the app evolves without the need to change the previously stored records. Figure 79 shows an example of requirement change at runtime.



```
{
 first_name:
   “Lauren”,
 …
}
```

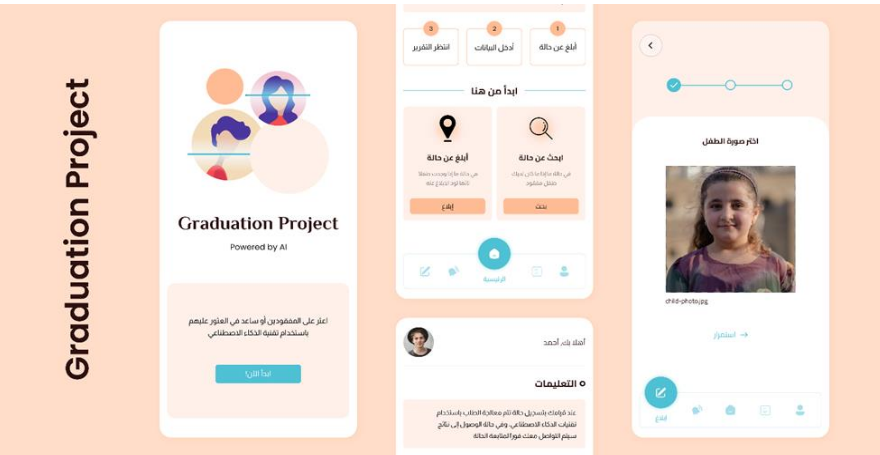
```
{
 first_name:
   “Lauren”,
 school: “Pawnee
   Elementary”
 …
}
```

Figure 79

4. Program Faster
MongoDB has support for the following languages: C, C++, C#, Go, Java, Node.js, PHP, Python, Ruby, Rust, Scala, and Swift. [25]

And as the documents are stored in a JSON-like format, they can map easily to objects in object-oriented programming languages which makes the interface very friendly and requires a few lines of code.

Application running process



اعثر على المفقودين أو ساعد في العثور عليهم باستخدام تقنية الذكاء الاصطناعي

عند قيامك بتسجيل حالة تتم معالجة الطلب باستخدام تقنيات الذكاء الاصطناعي. وفي حالة الوصول إلى نتائج سيتم التواصل معك فورا لمتابعة الحالة

Figure 80

Splash Screen

- A splash screen is a screen that displays while the application or other item is loading. After the load is complete, the user is generally taken to another more functional screen. The splash screen is generally just a display screen to orient users and gives them something to look at while the hardware is working to present the software to them.
- A splash screen is also known as a start screen or startup screen.

Login Screen & Signup Screen

- Login Screen is the next screen that appears after clicking Start now button. it has two fields email and password. If a user tries to log in without entering a valid email and password they will not be allowed to log in and will send a message handled by the backend according to the error if some of that error “please enter a valid email”, “password is not correct”, and “please enter all fields”. Also, this screen has a signup button which navigators to sign up and create a new user.
- Signup Screen has 5 text input fields following name, email, phone, password, and password. The error is also handled by the backend and a message is sent according to those errors.
- In signup when the user enters valid data those data are stored in a database to be checked when this user tries to log in with has created an account.

Home-Page Screen

- This Screen mainly consists of two main parts creating a new report and creating a new search. Clicking any of those two will route to the next screen which will be discussed later in the next section.
- Search means we have a missed child.
- Report means that we saw someone in the street and have a doubt that this child may be missed.

New Searches and Reports

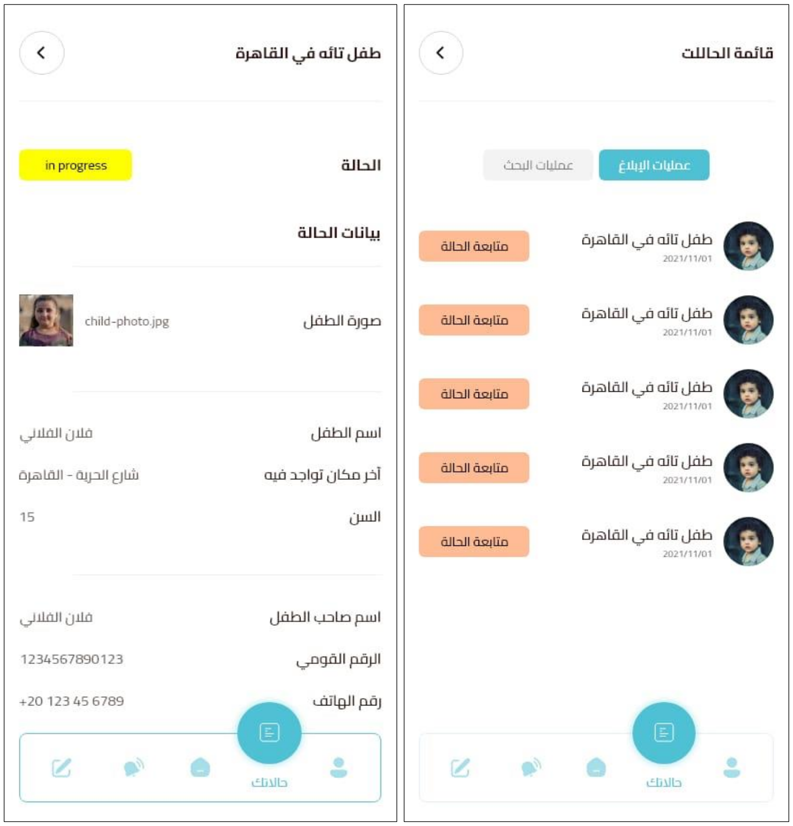
- As said above these two screens are meant for creating new searches and reports. Users fill a form (stepper form, to be user-friendly), and then those data are stored in the database.
- A script checks, if there's new data, is added to the database, and if there it'll either of search or report type.
- If the added data from the search script will download the image in save it in a search folder. Then run the model with this image and compare it with the corresponding report images in a report folder.
- If there's a match or more than one match results are sent to the admin (a web dashboard made using Reactjs to control the status of the report and search). If the admin sees that there's a match result will show in the application.
- There are 3 statuses:
    - In Progress: This means that model still running
    - Not Found: This means that model finishes its work and matches are found either due to there being no matches already or the admin sees that model was wrong about this case.
    - Matched: This means that we have a match for this case and the admin agrees on this. For this case, there will be an organization responsible for getting the child back to his family.

Summary Screen

Shows the summary info of the reported case as shown in Figure 81 (left).

Follow your searches and reports

- This screen contains two buttons one to follow report cases and the other one to follow search cases.
- When a button is clicked route to the screen corresponding to this screen is made and navigation occurs.
- follow the report cases screen, or follow the search screen to fetch the data from the API and show the missing reported address, the image of the child, and the status. As shown in Figure 81 (right).



طفل تائه في القاهرة

الحالة	in progress

بيانات الحالة

صورة الطفل	child-photo.jpg
اسم الطفل	فلان الفلاني
آخر مكان تواجد فيه	شارع الحرية - القاهرة
السن	15
اسم صاحب الطفل	فلان الفلاني
الرقم القومي	1234567890123
رقم الهاتف	+20 123 45 6789

قائمة الحالت

عمليات البحث | عمليات الإبلاغ

طفل تائه في القاهرة 2021/11/01	متابعة الحالة
طفل تائه في القاهرة 2021/11/01	متابعة الحالة
طفل تائه في القاهرة 2021/11/01	متابعة الحالة
طفل تائه في القاهرة 2021/11/01	متابعة الحالة
طفل تائه في القاهرة 2021/11/01	متابعة الحالة

Figure 81

Future Work

1. Using CycleGANs to make datasets more diverse and have all races equally as white people are significantly more represented than black people and Europeans are significantly more represented in data sets than Indians, this difference in ethnic representation is largely reflected in the quality and accuracy of the findings of the model.

2. Using the loss function from the paper "Improving Face Recognition with Large Age Gaps by Learning to Distinguish Children" that focuses on identifying the children’s identity out of the gallery of adults.

3. Using the loss function from the paper "AdaFace: Quality Adaptive Margin for Face Recognition" that deals with low-quality images. We tried to use super-resolution but that degraded the accuracy, therefore, it seems that modifying the loss function is the solution.

References

[1] Handbook of Face Recognition, Second Edition, 2011.

[2] Rich feature hierarchies for accurate object detection and semantic segmentation, 2013.

[3] Fast R-CNN, 2015.

[4] Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, 2015.

[5] Focal Loss for Dense Object Detection, 2017.

[6] Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks, 2016.

[7] RetinaFace: Single-stage Dense Face Localisation in the Wild, 2019.

[8] A Gentle Introduction to Deep Learning for Face Recognition, 2019.

[9] CosFace: Large Margin Cosine Loss for Deep Face Recognition, 2018.

[10] SphereFace: Deep Hypersphere Embedding for Face Recognition, 2017.

[11] ArcFace: Additive Angular Margin Loss for Deep Face Recognition, 2018.

[12] DeepFace: Closing the Gap to Human-Level Performance in Face Verification, 2014.

[13] Deep Learning Face Representation from Predicting 10,000 Classes, 2014.

[14] Deep Face Recognition, 2015.

[15] FaceNet: A Unified Embedding for Face Recognition and Clustering, 2015.

[16] Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, 2016.

[17] ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks, 2018.

[18] Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data, 2021.

[19] https://blog.paperspace.com/image-super-resolution

[20] https://stackoverflow.blog/2022/02/21/why-flutter-is-the-most-popular-cross-platform-mobile-sdk

[21] https://medium.com/flutter-community/top-10-reasons-flutter-is-better-for-your-app-development-30d50e345b29

[22] https://www.mongodb.com/compare/relational-vs-non-relational-databases

[23] https://insightsoftware.com/blog/whats-the-difference-relational-vs-non-relational-databases

[24] https://www.mongodb.com/developer/products/mongodb/top-4-reasons-to-use-mongodb/

[25] https://www.mongodb.com/languages

[26] MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition, 2016.

[27] Orthogonal Deep Features Decomposition for Age-Invariant Face Recognition, 2018.

[28] Dex: Deep expectation of apparent age from a single

[29] image. In: International Conference on Computer Vision Workshops (ICCVW), 2015

[30] Look Across Elapse: Disentangled Representation Learning and Photorealistic Cross-Age Face Synthesis for Age-Invariant Face Recognition, 2018.

[31] Learning Face Representation from Scratch, 2014.

[32] VGGFace2: A dataset for recognizing faces across pose and age, 2018.

[33] When Age-Invariant Face Recognition Meets Face Age Synthesis:

[34] A Multi-Task Learning Framework, 2021.

[35] http://megaface.cs.washington.edu
[36] Deep Residual Learning for Image Recognition,2015

[37] Squeeze-and-Excitation Networks ,2017

[38] Attention Is All You Need,2017

[39] Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks,2016

[40] Image-to-Image Translation with Conditional Adversarial Networks paper

[41] https://globalmissingkids.org/awareness/missing-children-statistics

[42] https://www.bbc.com/news/world-middle-east-53564935