

From an Eye to a Face:

A Novel Approach to Surveillance and Identity Reconstruction

By Tse Yik Long, Ronnie Jok, Peter Ng, Isaac Lee

Abstract

The use of AI in fields both face generation and face recognition is becoming increasingly prominent nowadays. Despite the fact that fields of such are having dramatic developments, it is always the case that security cameras can't capture the whole face of the suspect, especially when they usually cover up their major facial features except their eyes. With the addition of the pandemic, people are wearing masks worldwide, hence in order to keep security systems up to date, it's crucial to have applications that can generate, or even recognise faces from one eye, as camera angles might not always capture both eyes in most of the situations. Inspired by a similar paper, we proposed a novel approach to solve the problem, using autoencoder as the generator, meanwhile utilising the VGG19 model for feature loss. We now also propose an improved version of feature loss by adjusting loss weights.

I. Introduction

Face generation has been a popular topic in machine learning for its easy data collection and is very suitable for the demonstration of new generative model architectures. Developments such as the CNN [1] architecture, the VAE architecture, the GAN [2] architecture have a profound impact on not only face generation, but on all real-life generation tasks, such as sound generation, text generation, etc. They show that deep neural networks are capable of understanding complicated real-life patterns which only humans are capable of recognising in the past.

Researches on face generation can be generally divided into two categories, unconditional face generation and conditional face generation. Unconditional face generation is a comparatively easier task, in which the model is only required to synthesize a photorealistic face with any prerequisites, such as requirements on skin colour, race, etc. Usually, a random gaussian noise is inputted into the model to ensure the variety of generated faces. The development of unconditional face generation has been rapid in recent years, with projects such as "This Face does not Exist" [3] using a StyleGAN architecture proving the possibility of the generation of detailed super high-resolution faces. In light of maturing technologies in unconditional face generation, projects in unconditional face generation start to appear. They are more useful in real life and can offer better computer-automated solutions to real-life problems such as smart surveillance. However, conditional face generation is much more challenging as it adds complication to the task of face generation, forcing the model to identify and utilize the correlation between faces and inputted information. One major example of conditional face generation is the recovery of faces from partial facial features, such as the generation of a full face from a side-angle face and eye-to-face synthesis, which inspired our project.

In the process of researching the topic, we have noticed that GAN seems to be the most popular if not the only solution to face generation. This can be easily understood as adversarial models are more capable of capturing small details which add to a realistic output and can constantly improve themselves by the competition of a generator and discriminator model. Rich developments in GAN and the vast variety of GAN models also allow it to be applicable to many generative tasks, not limited to face generation. In our case, conditional GAN models are proven effective to force the model to find the relationships between inputs and outputs, i.e. the inputted eye and output face. Although GAN could be a possible solution to our task, we are dissuaded by its disadvantages. To name one, GANs are unstable

as improvements in the generator model come from the competition between the generator and discriminator, not a direct fit of input and output data. The generator or the discriminator can easily become too powerful and would cause failure as one of the model cannot learn from the other. This means that for the GAN architecture to work, constant optimization of the learning rates and the ratio of learning of the generator and discriminator is required.

However, alternatives to face generation are few and are usually inferior to the GAN architecture. Solutions such as L1 loss always result in problems such as blurry images and mode collapse. In replacing GAN, we must be able to solve the two above problems. To increase the precedence of details in the generated face, we are inspired by the use of perceptual loss (i.e. indirect transfer learning of the VGG19 network) which focuses on textures and features but not individual pixels. To increase the effectiveness of feature loss, we propose a novel method to force the model to focus on details of a face while preserving its identity, which is by adjusting the loss weights of each layer of the VGG19 network. Compared to GAN, this methodology is much more stable as it does not have adversarial elements in training, meanwhile preserving the main advantages of GAN, i.e. details in the generated face.

In the following part of the paper, Section II would introduce some related works to our project and explaining some inspirations and references we have taken when doing this project. Section III would mainly focus on our data collection and preparation process. Section IV would be an in-depth explanation and analysis of the methodologies we have incorporated into our project. Finally, Section V would be a demonstration of our results and some evaluation of them.

II. Related Works

Surveillance and Identification Technologies

Since the creation of facial recognition systems, they have been widely used for surveillance purposes, as they can help law enforcement agencies to accurately identify suspects in cases. These systems had been enhanced over the years to what they are today.

In the 1960s, automated facial recognition was first developed, but the first-generation facial recognition technology required a human to establish the coordinates of the facial features in a photograph before the photograph can be used by the computer for recognition. This system recognizes human faces comparing 20 distances between the coordinates of facial features. [4]

In September of 1993, the Face Recognition Technology (FERET) programme was started, with Dr. P. Jonathon Phillips, Army Research Laboratory, Adelphi, Maryland, serving as technical agent, and sponsored by the Department of Defense Counterdrug Technology Development Program Office. The goal of the programme was to develop automatic face recognition capabilities that could be employed to assist security, intelligence, and law enforcement personnel in the performance of their duties. [5]

In the early 1990s, principal component analysis (PCA) allowed facial recognition programmes to identify human faces in an image that contains other objects. [6] PCA reduces the number of variables in a dataset by collecting highly correlated data together. In human facial recognition programmes, PCA is usually used to produce a set of eigenfaces, which can be viewed as a set of standard face elements that is calculated by statistically analysing a large number of face images. As these eigenfaces are “standard” faces, every human face can be expressed as a combination

of these different eigenfaces. [7]

In 1997 the PCA Eigenface method of face recognition was improved upon using linear discriminant analysis (LDA) to produce Fisherfaces. [8] This method works on the same principle as PCA, and it performs dimensionality reduction while preserving as much of the class discriminatory information as possible. [9]

In the late 1990s, the Bochum system [10], in which a Gabor filter was used to record and link the face features [11], overtaking purely feature-based approaches. It can also often see through hinderance, leading to higher positional accuracy and reliability with the aid of landmark finding. [10]

In 2001, with the aid of the Viola-Jones object detection framework, real-time face detection in video footage became possible. [12] AdaBoost, the first ever real-time frontal-view face detector, was launched by combining the Viola-Jones object detection framework with the Haar-like feature approach, applying in object recognition of digital images. [13] The Viola-Jones algorithm is also used in the data collection stage of our project, which we will introduce in the data section of our paper.

Tasks related to Face Generation

Eyes to face synthesis can be considered as a generalized problem of face generation with partial facial features, for which images are reconstructed from partially available information, i.e., conditions. One of the popular solutions to this is by using Generative adversarial networks. As for unconditional face generations, they can be treated as face inpainting with random noises by using training for GANs. [14] produces high resolution (1024x1024) face images with a relatively stable training instance by taking in randomly initialized latent vectors as inputs and constructs a never-before-seen face image. [15] produces state-of-the-art images with great clarity detail and is further implemented in the well-known “this person does not exist” project [16].

Meanwhile, ace generation with conditions requires more manual control to integrate computational face generation into real-life applications. [17][18][19] Adding an extra parameter “condition” to the formula of generation, conditional GAN, or CGAN, originated from [20]. In [21], faces are reconstructed from incomplete face images. DeMeshNet reconstructs detailed ID photos that are corrupted by mesh-like lines and watermarks through a 3-step feature extraction process. With breakthroughs in GAN and its variants, [22] reconstructed faces from corresponding eyes as an attempt to improve surveillance abilities. [18] proposed a Unet-structured GAN that maps an image with only a pair of eyes to an image of the corresponding face.

Development of Generative Networks

Simple Neuron Networks

When face generation is first introduced, simple neural networks were being used. In [1], dense 3D face and single face image are reconstructed from monocular video, especially on the usage of image-based 3D face reconstruction in face recognition [23][24] and face animation [25][26] In early networks of such, L1 loss (Manhattan distance loss) and L2 loss (Euclidian distance loss) were being implied to minimise the error. However, the results of CNN with L1 and L2 losses are still not satisfactory because the images tend to get blurry and fail to grasp details, as well as having artefacts [27].

Generative Adversarial Nets

[2] processed improved networks, by proposing a new network called the Generative Adversarial Nets. Based on a double model involving a generator and a discriminator, as the two trains each other to solve more complex problems, including style transfer [18], super-resolution, [18] and image inpainting [28]. However, GANs are unstable in training, leading to poor results and problematic tuning. [29] used the geometric metrics regulariser and the mode regulariser to solve the problem based on the original GAN model to create MRGAN, and [30] created a new algorithm from the traditional GAN model, by replacing the JS-Divergence with 1-Wasserstein distance to measure the difference between the model and target distributions, leading to better results and more efficient training [30]. [31] used Conditional Generative Adversarial Nets (CGAN), which fed the original input to both the generator and critic, for the discriminator to recognize the relevance between the input and output, avoiding mode collapse and increasing correlation between the result and the original data.

[22] designed an end-to-end network based on conditional generative adversarial networks (GAN) to generate the face information based only on the available data of the eye region. Despite having significant improvements, the GANs still have the issue of unstable training process which is rooted in their architecture. To solve the problem [32] used a thorough evaluation of networks of increasing depth using an architecture with very small (3×3) convolution filters. Being a totally different architecture, there is no adversarial loss within the network, showing a significant impact on eliminating earlier mentioned problems.

Our approach

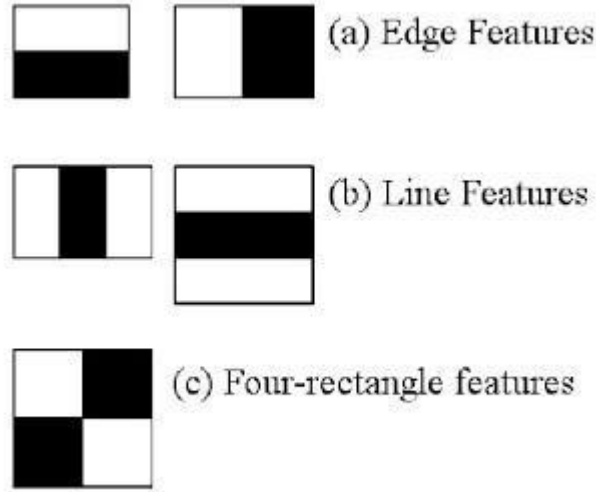
With the rapid development of face recognition and reprinting technology, GAN and its variants are being regarded as typical models for image inpainting and models of such. After our attempt to recreate the architecture in [33], we found GANs are unstable in training [34][30][20]. Referring to [32], its pretrained, deep convolution layers of the VGG model does a good job of grasping the details of the image, hence feature loss is considered our alternative for image processing. Feature loss functions are used when comparing two different images that look similar, and was used in [35] for Real-Time Style Transfer and Super-Resolution by a per-pixel loss between the output and ground-truth images. This gives similar qualitative results comparing to the optimization-based method but is three orders of magnitude faster, as well as giving visually pleasing results while a per-pixel loss is being replaced by a perceptual loss.

III. Data

In our project, the 7500 faces from the CelebA dataset [36] (in a total of 202599) were used for training, while the FFHQ face dataset is used for the evaluation of the model.

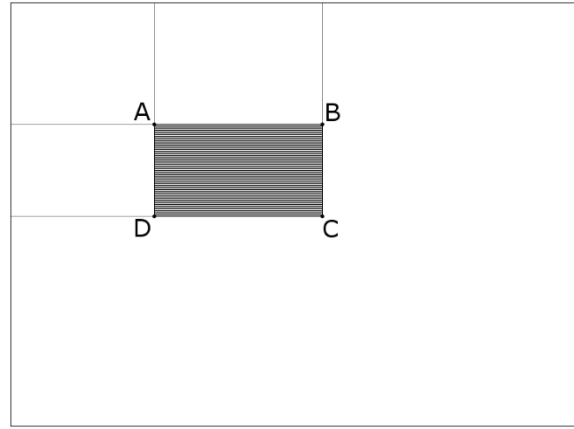
Using the OpenCV module (Open Source Computer Vision Library), we generate 2 pairs of eye-face data from each image from the dataset, given the image contains 1 face and 2 eyes, using haar-feature based cascade classifiers [37]. The rationale behind this requirement is that if an image contains more than 1 face or less than 2 eyes, it is probably the case that the face retrieved is only a partial face. On the other hand, if an image contains more than 2 eyes, the classifier may wrongly a non-eye object as an eye.

A Haar feature refers to the value representing a section of an image, which indicates the existence (or absence) of certain characteristics (such as edges or changes in texture) by calculating the differences between sections.



圖表 1: Examples of Haar features (Taken from [40])

A rectangular Haar feature can be defined as filters that are calculated by taking weighted sums of integrated areas of the images. [37] The equation is as follows:



圖表 2: Calculation of Haar feature (Taken from [40])

$$Sum = I(A) + I(C) - I(B) - I(D)$$

To figure out the best features out of all thousands of them, Adaboost was used, as we applied each and every feature on all the training images. With features of minimum error rate selected, they are sorted to most accurately classify the face and non-face images.

Note that classifiers are also categorised into strong and weak ones, as reflected from their error rates. Weak classifiers can be combined into a strong classifier by taking the weighted sum of their results over an identical feature.

After the cropping process, faces are resized to 128x128 and eyes are resized to 32x32 by a bilinear interpolation, which works by taking four known neighbouring pixels as reference. Finally, every pixel value in each of the training image is normalized from the range of (0,255) to (0,1).

The normalization process can be summarized by the following:

$$N(r, g, b) = (\frac{r}{255}, \frac{g}{255}, \frac{b}{255})$$

, where r, g, b represent the three colour channels of a pixel.

IV. Methodology

The aim of face synthesis from a single eye is to supplement the missing information of a face in the image of an eye. The problem of eye-to-face synthesis can be summarized in a function:

$$M(l_{eye}) = l_{face}$$

, where M represents the model, l_{eye} is an eye (the input) and l_{face} an image of a face (the output).

Specifically, we used an autoencoder network in our model, combined with feature loss and standard deviation loss during training.

Autoencoder

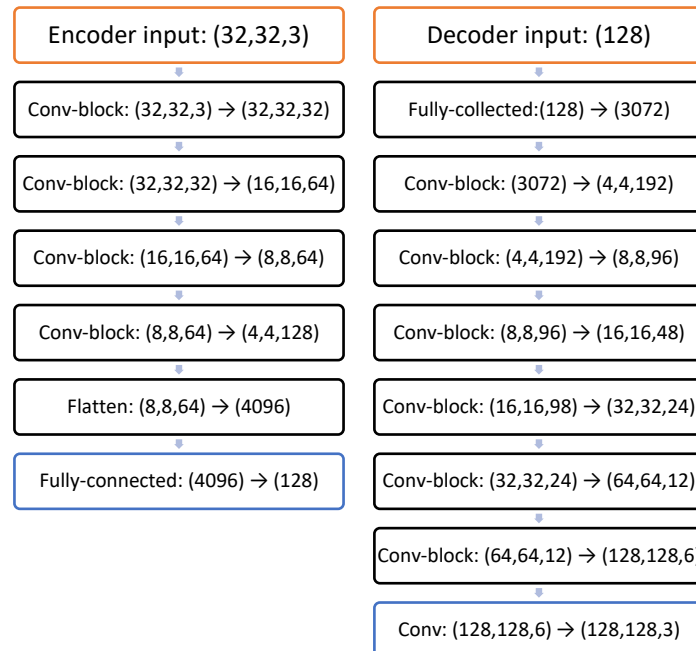
Our autoencoder can be divided into two parts, the encoder and the decoder. The aim of the encoder is to convert the image of an eye to a latent vector that represents the important features of the inputted eye, while the decoder turns the latent vector into an image of a face. It can be summarized by the following equations:

$$f_e(l_{eye}) = z$$

$$f_d(z) = l_{face}$$

, where f_e represents the encoder, z represents the encoded vector, f_d represents the decoder.

The implementation details of the autoencoder model can be illustrated as follows, where each Conv-block contains two groups of layer, each containing a convolutional layer, a batch normalization layer and a leaky-relu activation function. The convolutional layer in the first group has a stride of 2 or a transpose stride of 2 which reduces and enlarges the dimension by 2 respectively. Note that the first conv-block in the encoder has only 1 group of layer, and the activation function for the last convolutional layer in the decoder is sigmoid to bound the output pixel value in $[0,1]$.



Mean Absolute Error (L1 Loss)

L1 loss is commonly utilized in computer models as it is the simplest most intuitive method of calculating loss. Its objective is to minimize the differences in the pixels between the generated image and the target image. L1 loss takes the mean of the absolute error of the generated image when compared to the target image, without considering their directions. The range of L1 loss is $[0, \infty]$. The formula for L1 loss is:

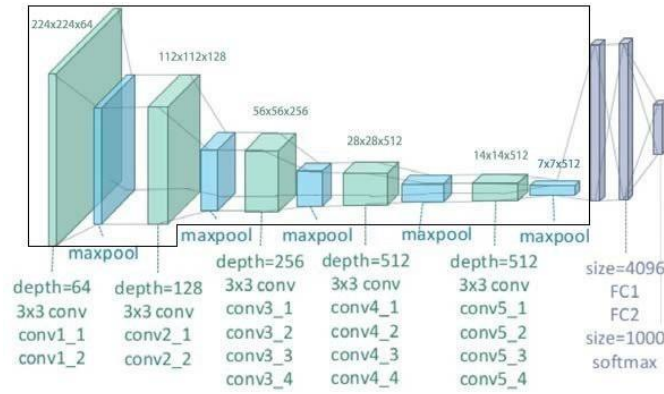
$$L_1 = \frac{\sum_{i=1}^n \|y_i - y_i^{pred}\|}{n}$$

, where i represents one pixel in an image and n represents the total number of images in a batch. [26]

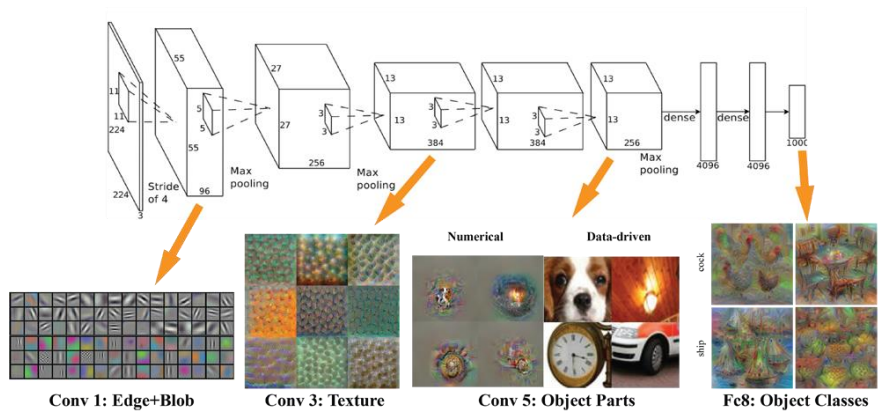
However, L1 loss is not used in our model as it creates artefacts and causes the image to be blurry [38]. L1 loss only measures the euclidean distance between the pixels of the generated image and target image, thus it is hard for a model using L1 loss to grasp concepts such as edges and textures. It also causes the problem of mode collapse as a model can achieve a low L1 loss even if it generates similar images for different inputs.

Feature Loss (Perceptual Loss)

Inspired by the paper *Perceptual Loss for Real-Time Style Transfer and Super-Resolution* [35], we decided to use feature loss to replace GAN loss and L1 loss in order to compensate for the shortcomings of them. By using feature loss, our model is more able to generate an image that is more similar to the target image, instead of constructing a whole new face. When compared to other kinds of losses, feature loss is more able to identify the details of features in an image, thus creating a more realistic image that is similar to the target image.



圖表 3: The architecture of VGG19 neural network, the layers included in the box is the part we utilized in our model (Modified from: https://www.researchgate.net/figure/illustration-of-the-network-architecture-ofVGG-19-model-conv-means-convolution-FC-means_fig)



圖表 4: The output of several layers of the VGG19 model (Taken from: http://cs.brown.edu/courses/cs143/2017_Spring/proj6a/)

In our model, we construct a model combining our generator and the VGG19 model. In training, the output of each layer of the VGG19 model in the combined model would be compared with the actual output when the target image is feed to the VGG19 model by the L1 loss.

However, there is a problem with this approach. As each layer in the VGG19 model has different dimensions, the Euclidean distance between pixels can vary to a large extent. In actual terms, during training, the L1 loss of earlier and

later layers, when compared to that of the middle layers, can differ for about 1000 times (from 10^{-1} to 10^2). When added together, the model will place too much importance on the intermediate layers. The original paper suggests solving this by dividing it by the dimension of each layer as follows:

$$Feature\ loss = \sum_i \frac{\|\phi_i(y) - \phi_i(y_{pred})\|}{C_i H_i W_i}$$

, where i represents a layer in the VGG19 model, ϕ_i represents the output of the VGG19 at layer i and $C_i H_i W_i$ represents the product of channel, height and width of layer i .

However, this solution is not optimal. In trial, there are still differences in the scale of losses, this time ranging from approximately 10^{-5} to 10^{-2} . Therefore, we propose a new method to balance the losses of each layer to ensure that every layer in the VGG19 model is considered during training. The method works by first evaluating a newly initiated generator and record the initial L1 loss in each layer in the VGG19 model. Our new feature loss can then be calculated as follows:

$$Feature\ loss = \sum_i \frac{\lambda_i \|\phi_i(y) - \phi_i(y_{pred})\|}{l_{initial}}$$

, where $l_{initial}$ represents the initial L1 loss for layer i of the VGG19 model and λ_i represents the loss weight for layer i , which will be explained in the following paragraph.

After balancing the loss, we can now modify the loss weights to make the generator focus on some layers in the VGG19 model as wish. As we want the generator to focus on the texture of the output to make the output image more detailed, we devise a weight loss system as follows:

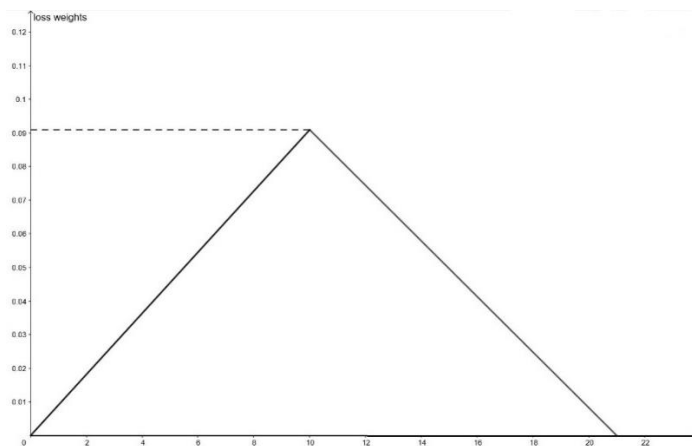
Let $x = \frac{2}{n}$, where n is the total number of layers of the VGG19 used in training

If $i \leq f, \lambda_i = \frac{x i}{f}$, where f is the position of the focus layer in the VGG19 model, with 0 being the first

If $i > f, \lambda_i = \frac{x(n - f - 1)}{n - 2}$

In our training, a setting of $f = 10$ and $n = 22$ is used.

When drawn in a graph, a linear progression and regression of loss weights can be seen:



圖表 5: Arrangement of loss weights in the VGG19 model plotted on a graph with loss weights against layer number
The reason behind such decision is to prevent drastic changes in loss weights. Under other alternatives, such as

geometric progression, the difference between the “focused” layers and the other layers would be too great and the model may neglect small details or global features. On the other hand, if one uses a weight progression that is too flat on top, the model may lose focus. Furthermore, our formula above keeps the sum of all the loss weights to be 1, so if other losses are to be added to the training, the weights for the added loss will be easier to set.

One final thing to note is that as the VGG19 model is trained on the imagenet competition dataset [39], images (that is normalized to 0 to 1) have to be preprocessed before inputting into the model. The preprocessing can be summarized by the following formula:

$$Preprocess(r, g, b) = 255 \times (r - 103.939, g - 116.779, b - 123.68)$$

During training, there was a minor mistake in which we mixed up the order of the values of RGB. However, as the values were similar, and both our generated batches and target batches had the same mistake, the overall effect of this mistake on our model is insignificant.

V. Results










Training

Our model was trained using the Adam optimizer using the suggested learning rate (10^{-3}) [40]. To prevent overfitting and to increase training efficiency, batch normalization is used [41]. Our training set consists of 15000 pairs of eyes and faces, which are fed into the model with a batch size of 64 during training. The model was trained for 150 (± 1) epochs with a GPU in Google Collaboratory.

Results and Evaluation

We will first analyse the performance of our model in several aspects, namely angle of face, skin colour, lighting, face shape, cosmetics and clarity. We will then present some failed and interesting results to show the shortcomings of our current methodology. The following images are all generated from eyes in test data.

Angle

Eye			
Real face			
Generated face			


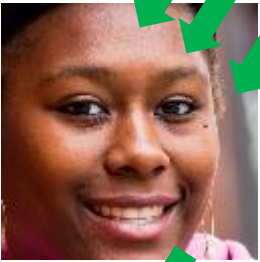
From the above results, it can be seen that the model is able to grasp the direction of faces from the subtle tilting of the inputted eyes.

Skin colour

Eye			
Real face			
Generated face			




From the above result, it can be seen that the model is able to identify the skin colour from the colour of the skin surrounding the eye.

Lighting

Eye			
Real face			
Generated face			

From the above results, it can be seen that the model is able to grasp the direction of the lighting from the shading on the skin around the eyes.


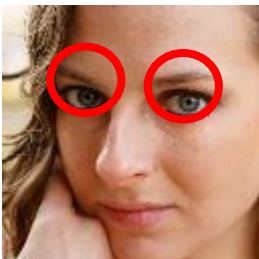

Face shape

Eye			
-----	---	---	---

Real face			
Generated face			

From the above results, it can be seen that the model is able to recognize the face shapes from the information provided by the eyes.

Cosmetics







Eye	Real face	Generated face
		



From the results above, it is obvious to us that the model is able to grasp the makeup of the person from the eyes, such as eyeshadows and even the rouge are portrayed in the generated image.

Clarity

Despite the model being able to grasp many important facial features from the inputted eye, the clarity of the generated results is not satisfactory. When the image is zoomed, it is easily observable that there are many white dots on the results, which seems to be a common problem of feature loss. Furthermore, as the VGG19 model only grasp details of the focused object, the background and the hair seem to be neglected during the training process.

Failed results

Eye			
Real face			

Generated face			
----------------	---	---	--

For inputs with special angles or face shapes, the face generated would have a rather special shape, meanwhile for some certain results they were being affected by the addition of glasses, as part of the eye is being covered, creating distortions in the generated face.

Comparison with GANs

Although we have GANs are the “state of the art” model for generative tasks, we fail to produce comparative results in this case, which is most likely because of the unstableness of GANs. The GAN model we used for fair testing (with a similar architecture and the same training settings) is a Wasserstein conditional GAN with gradient penalty [42]. Some results (generated from the training data) are shown below:

Eye			
Real face			
Generated face			

Although these results are definitely not the expected results from GANs, we can conclude that GANs are not the best choices when a quick and fast solution is needed.

VI. Conclusion and Future Works

As the quality of our results is still unsatisfactory compared to other similar face generation models using GAN, we may try a combination of feature loss and adversarial loss for better results. Additionally, other kinds of face recognition networks can replace the VGG network used in our model. In the future, we may create models that can generate a human face from other partial facial features, e.g. mouth and nose, and our ultimate goal is to create a model that can generate an image of a human face from any partial face features provided to it.

References

1. Guo, Y., Cai, J., Jiang, B., & Zheng, J. (2018). Cnn-based real-time dense face reconstruction with inverse rendered photo-realistic face images. *IEEE transactions on pattern analysis and machine intelligence*, 41(6), 1294-1307.
2. Goodfellow, I. (2016). NIPS 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160.
3. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8110-8119).
4. Nilsson, Nils J. (2009). *The Quest for Artificial Intelligence*. Cambridge University Press. ISBN 9781139642828.
5. National Institute of Standards and Technology. Face Recognition Technology (FERET) Retrieved from <https://www.nist.gov/programs-projects/face-recognition-technology-feret>
6. Malay K. Kundu; Sushmita Mitra; Debasis Mazumdar; Sankar K. Pal, eds. (2012). *Perception and Machine Intelligence: First Indo-Japan Conference, PerMIn 2012, Kolkata, India, January 12-13, 2011, Proceedings*. Springer Science & Business Media. p. 29. ISBN 9783642273865.
7. Imperial College London, Faculty of Engineering, Department of Computing. PCA Principal Component Analysis. Retrieved from <https://www.doc.ic.ac.uk/~hh4017/PCA>
8. Jun Wang; Laiwan Chan; DeLiang Wang, eds. (2012). *Neural Information Processing: 13th International Conference, ICONIP 2006, Hong Kong, China, October 3-6, 2006, Proceedings, Part II*. Springer Science & Business Media. p. 198. ISBN 9783540464822.
9. F. Z. Chelali, A. Djeradi and R. Djeradi, "Linear discriminant analysis for face recognition," 2009 International Conference on Multimedia Computing and Systems, Ouarzazate, Morocco, 2009, pp. 1-10, doi: 10.1109/MMCS.2009.5256630.
10. Okada, K., Steffens, J., Maurer, T., Hong, H., Elagin, E., Neven, H., & von der Malsburg, C. (1998). The Bochum/USC face recognition system and how it fared in the FERET phase III test. In *Face Recognition* (pp. 186-205). Springer, Berlin, Heidelberg.
11. Wechsler, Harry (2009). Malay K. Kundu; Sushmita Mitra (eds.). *Reliable Face Recognition Methods: System Design, Implementation and Evaluation*. Springer Science & Business Media. p. 12. ISBN 9780387384641.
12. Malay K. Kundu; Sushmita Mitra; Debasis Mazumdar; Sankar K. Pal, eds. (2012). *Perception and Machine Intelligence: First Indo-Japan Conference, PerMIn 2012, Kolkata, India, January 12-13, 2011, Proceedings*. Springer Science & Business Media. p. 29. ISBN 9783642273865.
13. Li, Stan Z.; Jain, Anil K. (2005). *Handbook of Face Recognition*. Springer Science & Business Media. pp. 14–15. ISBN 9780387405957.
14. Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.
15. Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.
16. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8110-8119).

17. Gauthier, J. (2014). Conditional generative adversarial nets for convolutional face generation. Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester, 2014(5), 2.
18. Lu, Y., Tai, Y. W., & Tang, C. K. (2018). Attribute-guided face generation using conditional cyclegan. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 282-297).
19. Di, X., Sindagi, V. A., & Patel, V. M. (2018, August). Gp-gan: Gender preserving gan for synthesizing faces from landmarks. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 1079-1084). IEEE.
20. Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.
21. Zhang, S., He, R., Sun, Z., & Tan, T. (2017). Demeshnet: Blind face inpainting for deep meshface verification. *IEEE Transactions on Information Forensics and Security*, 13(3), 637-647.
22. Chen, X., Qing, L., He, X., Su, J., & Peng, Y. (2018). From eyes to face synthesis: a new approach for human centered smart surveillance. *IEEE access*, 6, 14567-14575.
23. Tuan Tran, A., Hassner, T., Masi, I., & Medioni, G. (2017). Regressing robust and discriminative 3D morphable models with a very deep neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5163-5172).
24. Blanz, V., & Vetter, T. (2003). Face recognition based on fitting a 3d morphable model. *IEEE Transactions on pattern analysis and machine intelligence*, 25(9), 1063-1074.
25. Ichim, A. E., Bouaziz, S., & Pauly, M. (2015). Dynamic 3D avatar creation from hand-held video input. *ACM Transactions on Graphics (ToG)*, 34(4), 1-14.
26. Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., & Nießner, M. (2016). Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2387-2395).
27. Zhao, H., Gallo, O., Frosio, I., & Kautz, J. (2016). Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1), 47-57.
28. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2536-2544).
29. Che, T., Li, Y., Jacob, A. P., Bengio, Y., & Li, W. (2016). Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*.
30. Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.
31. Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.
32. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
33. Chen, X., Qing, L., He, X., Su, J., & Peng, Y. (2018). From eyes to face synthesis: a new approach for human-centered smart surveillance. *IEEE access*, 6, 14567-14575.
34. Che, T., Li, Y., Jacob, A. P., Bengio, Y., & Li, W. (2016). Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*.
35. Johnson, J., Alahi, A., & Fei-Fei, L. (2016, October). Perceptual losses for real-time style transfer and superresolution. In *European conference on computer vision* (pp. 694-711). Springer, Cham.
36. Liu, Z., Luo, P., Wang, X., & Tang, X. (2018). Largescale celebfaces attributes (celeba) dataset.
37. Viola, P., & Jones, M. (2001, December). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition*.

CVPR 2001 (Vol. 1, pp. I-I). IEEE.

38. Zhao, H., Gallo, O., Frosio, I., & Kautz, J. (2016). Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1), 47-57.
39. Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.
40. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
41. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03*
42. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*.