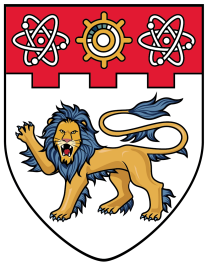
Generating human face by generative adversarial networks

Soh, Cecelia Yan Pei

2023

Soh, C. Y. P. (2023). Generating human face by generative adversarial networks. Final Year Project (FYP), Nanyang Technological University, Singapore.
https://hdl.handle.net/10356/165834



NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

SCSE22-0307

Generating Human Faces by Generative Adversarial Networks

Submitted by: Cecelia Soh Yan Pei (U1920800C)

Supervisor: Assoc Prof Chen Change Loy

Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Computer Engineering of the Nanyang Technological University Singapore

School of Computer Science and Engineering

2023

Abstract

The Generative Adversarial Network (GAN) method is widely used for image generation, especially human face generation and modification. The recent GANs model can generate diverse photorealistic images and this relies on the powerful capabilities of semantic latent representation and feature disentanglement. Those capabilities are also fully demonstrated in this project. This project focuses on utilising GAN for generating human face images to train a visual alignment model, GANgealing, and utilising GAN for modifying human facial attributes on images that have been aligned by GANgealing. To achieve this, I first study the latest facial image generation methods, the StyleGAN series, which possess the capabilities of semantic latent representation and feature disentanglement. After that, I examine a GAN-supervised visual alignment method called GANgealing, which relies on StyleGAN2's feature disentanglement ability to generate the training data and the corresponding labels, in which human facial features are the same but the poses are diverse. The GANgealing aligns facial features to a unique central mode. With this alignment, a simple GAN approach, AttGAN, can semantically modify the facial attributes of face images. Undoubtedly, the editing must be correctly transferred to the original images. After all the models are trained, I integrated AttGAN into the GANgealing algorithm, allowing for alignment, editing, and editing propagation to be performed in a single application. Lastly, I investigated the performance of this facial attribute editing pipeline.

Acknowledgement

Throughout this project, I have received a great amount of support. Firstly, I would like to express my gratitude and appreciation to my supervisor, Assoc Prof. Chen Change Loy, for his supervision, professional guidance and patience throughout the duration of the Final Year Project. Next, I would also like to thank my mentor, PhD. Wang Yuhan, for his regular support and insightful help and directions, offered for this project. His experience and knowledge were really beneficial in getting the project started.

Table of Contents

Lists of Figures

List of Tables

Introduction

1.1 Background

Generative Adversarial Network (GAN) is a popular Machine Learning framework used to generate data samples from random noise. A GAN model is composited by two sub-models, i.e., a generator and a discriminator. The GAN model was first designed by Ian GoodFellow *et al.* [1] in June 2014. However, the traditional GAN can only produce low-resolution images. Over the past few years, some advanced GAN models were proposed, such as StyleGAN series models [2]–[5], and they are able to generate high-quality and photo-realistic images. Simultaneously, those StyleGAN models also possess the ability to feature disentanglement in a semantic-aware manner which is very useful for developing more diverse applications, especially human face modification tasks.

In the year 2021, Peebles *et al.* [6] leveraged the semantic understanding of StyleGAN in innovative ways. They approach the task of dense visual alignment by using GAN in a supervised manner. The proposed algorithm, GANgealing, utilises a pre-trained StyleGAN2 to generate the data and the corresponding label, which are used to train a Spatial Transformer Network (STN) [7] for geometric transformation and joint alignment. The label is the aligned version of the data and its latent code is learned simultaneously with STN training. Eventually, the well-trained STN enables the input images to be manipulated to a unique single view, as shown in the second row of Figure 1-1. The authors have also demonstrated that this technique can be used for image editing. As shown in the third and last row, the algorithm can visualise the correspondence and propagate any objects, such as moustaches, to human faces.

1.2 Objective and Scope

GANgealing's significant contribution, apart from visual alignment, is its capability to apply edits made on aligned images back to their original counterparts. The authors [6] demonstrated this ability by simply adding objects onto faces, as shown in Figure 1-1. This project aims to implement an alternative

type of editing, specifically semantic facial attribute editing. To achieve this, I first study and reimplement the GANgealing algorithm. After that, I looked for a suitable facial attribute manipulation technique. From my study [8], the generative models are the popular way for image editing. In my works, the facial attribute can be aligned to a single view, so I try to apply a relatively small and simple generative model, AttGAN [9], to modify the aligned images and avoid the training overhead. All in all, I investigate the effectiveness of the visual alignment executed by GANgealing for manipulating specific attributes of a person's face in an image or video.

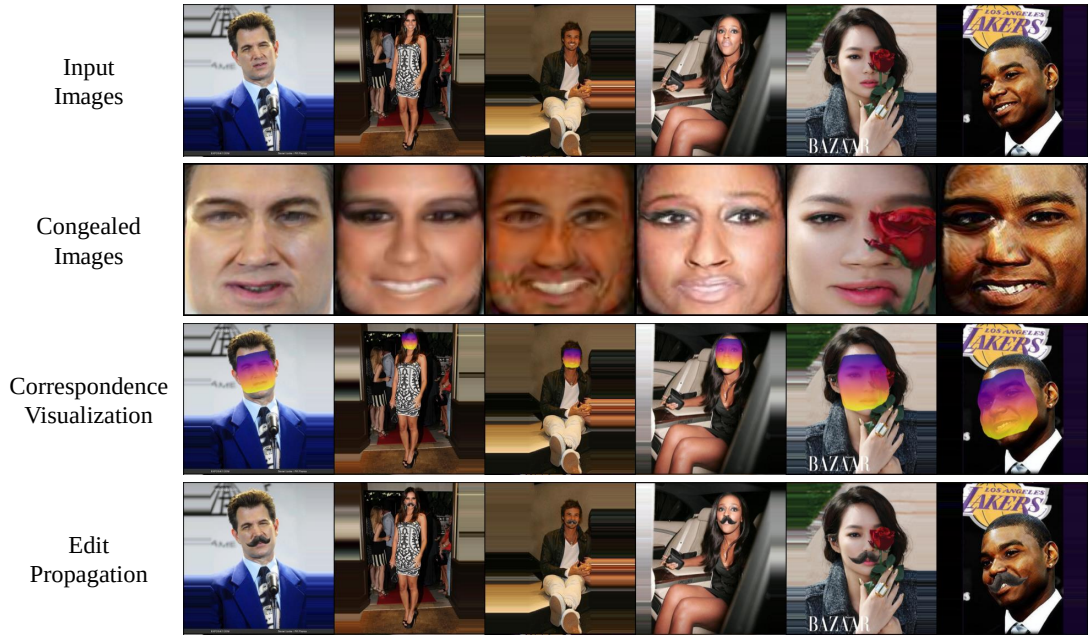


Figure 1-1: Given an input dataset of unaligned images (Top row), the GANgealing algorithm can align all images to a unique central mode (Second row). After adding some object, such as a mask or moustache, to the aligned images, the algorithm can propagate the objects back to the initial image (Third & Bottom row).

Related Work

2.1 Generative Adversarial Network

In computer vision, Generative Adversarial Network (GAN) is a powerful machine learning technique to learn the generation of realistic images. Typically, a GAN model is composited by two deep neural networks, i.e., a generator and a discriminator. The generator is trained to map the noise sample to synthetic images that can 'fool' the discriminator. On the other hand, the discriminator is trained to classify real images and images synthesised by the generator. In so doing, the discriminator can guide the generator to synthesise realistic images.

The first GAN model is proposed by Ian Goodfellow *et al.* [1], and its objective function is formulated as a min-max game as follows:

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}[logD(x)] + \mathbb{E}_{z\sim p_z}[log(1 - D(G(z)))]. \tag{2.1}$$

However, the training process of a GAN model is unstable. This instability can be caused by various factors, such as the vanishing gradient problem, mode collapse, and difficulty finding an equilibrium between the generator and discriminator. To mitigate those problems, various techniques have been proposed, such as DCGAN [10] utilising the CNN and batch normalisation to stabilise the training, CGAN [11] and InfoGAN [12] providing attribute or class labels to help the generator generate corresponding images and the discriminator to distinguish real images better. In addition, some techniques improve the cost function to avoid gradient vanishing and mode collapse. For example, the cost function proposed by WGAN [13] computes the Wasserstein distance between the distributions of real samples and generated samples:

$$\min_G \max_{||D||_L \leq 1} \mathbb{E}_{x\sim p_{data}}[D(x)] - \mathbb{E}_{z\sim p_z}[D(G(z))]. \tag{2.2}$$

where D must be the 1-Lipschitz function with weight clipping. Using weight clipping for the Lipschitz constraint can lead to a loss of capacity and complexity of the model. WGAN-GP [14] instead applies a gradient penalty on the

discriminator to enforce the Lipschitz continuity constraint.

As the development of GAN goes on, the usage of GAN has become variant. The typical applications of GAN include image synthesis, style transfer, image super-resolution, etc. For instance, CycleGAN [15] is able to train the model to switch the style between two domain images, while StarGAN [16] only trains a single generator to learn mappings among multiple domains. Super Resolution GAN [17] allows the model training to produce higher-resolution images. Progressive Growing GAN (PGGAN) [18] can generate high-quality 1024×1024 images by starting the model training with 4×4 resolution images and gradually increasing the resolution to 1024×1024. In recent years, StyleGAN series models [2]–[5] have been the most popular GAN models. StyleGAN models retain the advantages of PGGAN, i.e., the ability to generate high-quality images. They also perform well in style mixing, which is achieved by inputting another random latent code into upper or lower layers.

In this project, GANgealing utilises StyleGAN2 to generate the training data from latent vector $\mathbf{w}$. Then, it leverages the style mixing property to synthesise the correspondence labels by inputting a different latent vector $\mathbf{c}$ in the first entry. Besides, WGAN-GP is adopted for the adversarial learning of AttGAN, the model used for facial attribute editing in this project.

2.2 Facial Attribute Editing

Human facial attribute editing is a subfield of image editing. Semantic facial attribute editing focuses on manipulating certain attributes of a human face image while other irrelevant attributes remain untouched. AI-driven facial attribute editing has the potential to develop or improve AI-based facial photo editing software, such as digital make-up and facial feature adjustment. The editable facial attributes can be categorised into global and local attributes. Global attributes encompass the entire area of the input image, while local attributes only consider specific parts of the face. For instance, age, gender and skin colour belong to global attributes, while the shape and size of the eyes, nose and mouth are examples of local attributes.

According to [8], the existing semantic facial attribute models based on their architecture are reviewed in three main categories: photo-guide models, image-

to-image translation methods and encoder-decoder models. Photo-guide models typically have an additional generator to synthesise mask or landmark images, which indicate the facial editing region. The edited images are regenerated based on the changing of those auxiliary images. Image-to-image translation methods can be further categorised into bi-modal models and multi-modal models. The bi-modal models can only support the editing of a single attribute, while the multi-modal models can edit multiple attributes simultaneously. The representative model of this method is StarGAN [16], which is used for multi-domain image-to-image translation. Lastly, the encoder-decoder model is the common architecture for facial attribute editing. The encoder maps the input images into a latent space vector, whereas the decoder uses each point in the latent space to create a reconstructed face image. To modify specific attributes, some vector manipulation operators are used or the decoder is conditioned on a target attribute vector. AttGAN [9], the facial attribute editing model used in this project, is an encoder-decoder model with a conditional decoder.

Methodology

3.1 Overview

In this project, I trained a facial attribute editing model, AttGAN [9], which can modify the aligned images formed by GANgealing [6]. After that, I built an application by incorporating the pre-trained AttGAN in the GANgealing model. The overview of this process is depicted in Figure 3-1. Given an input image x, with an attribute a, a Spatial Transformer Network T, which has been trained through GANgealing, synthesises the corresponding aligned image $T(x)$. An attribute classifier C, trained through the AttGAN model, can then be used to predict the binary attribute label of $T(x)$. This eliminates the need for manual attribute labelling. After that, the predicted label a' is modified to the desired label b, which is used to condition the decoder of the AttGAN generator. Lastly, the image $T(x)^b$, which has been edited to attribute b, is wrapped back to the original image.

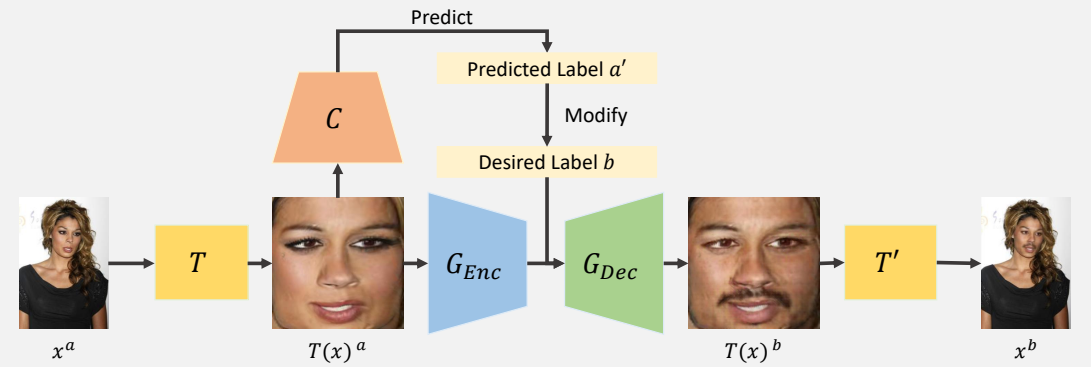


Figure 3-1: The pipeline of facial attribute editing via visual alignment.

3.2 GANgealing

Motivated by the classic Congealing method [19], Peebles *et al.* [6] proposed a GAN-Supervised Learning algorithm, GANgealing, for the dense visual alignment problem. The GANgealing algorithm is designed to train a Spatial Transformer Network (STN) utilising the GAN-generated data. STN aims to execute the geometric transformation of joint alignment, which enables the alignment of all specific objects presented in the image to a unique central mode across the data.

I used the GANgealing algorithm for human facial attribute alignment in this project. However, the algorithm is also trainable for other object datasets, such as LSUN Cars, Cats, Horses, etc. Some objects' poses might vary across the whole dataset; hence, those objects' features cannot simply align to a single central mode. GANgealing is adapted to a clustering algorithm to address the limitation, which can learn more than one target latent $\mathbf{c}$.

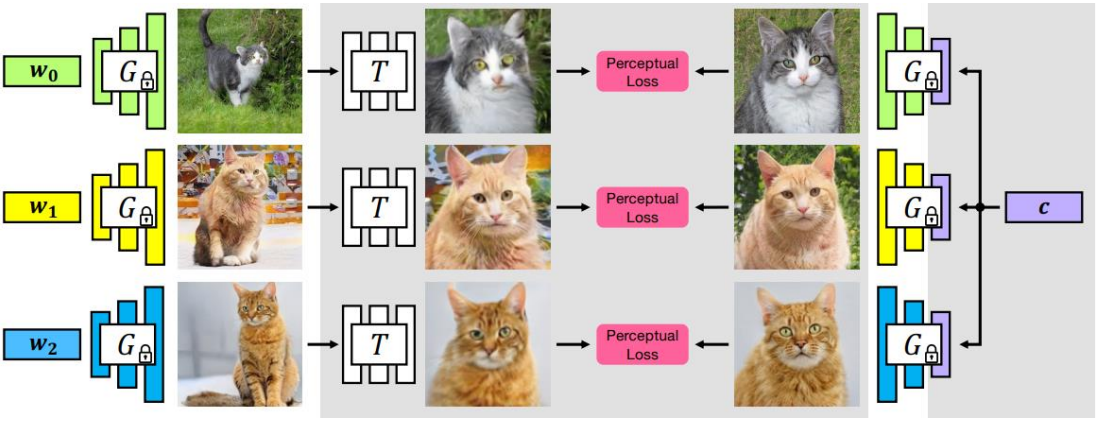


Figure 3-2: GANgealing algorithm's pipeline. Adopted from [6].

The pipeline of the algorithm is shown in Figure 3-2. The input of the STN (T) is a (x, y) pair, where x and y, respectively, represent the unaligned and aligned data generated by a pre-trained generator (G). The unaligned synthetic image $x = G(\mathbf{w})$ is mapped from a randomly sampled latent vector $\mathbf{w} \sim \mathcal{W}$, where $\mathcal{W}$ denotes the distribution over latent and $\mathbf{w} \in \mathbb{R}^{512}$. The aligned synthetic image $y = G(mix(\mathbf{c}, \mathbf{w}))$ is the label for the STN training, where $\mathbf{c}, \mathbf{w} \in \mathbb{R}^{512}$ and $mix(\mathbf{c}, \mathbf{w})$ stands for style mixing of two latent features $\mathbf{c}$ and $\mathbf{w}$. The details of pre-trained GAN and style mixing are introduced in Section 3.2.1. Simultaneously, the latent vector $\mathbf{c}$ is also optimized by learning scalar coefficients

$\{\alpha_i\}_{i=1}^{N}$. The latent vector $\mathbf{c}$ is parameterized as a linear combination of the top-N principal directions of $\mathcal{W}$ space:

$$\mathbf{c} = \bar{\mathbf{w}} + \sum_{i=1}^{N} \alpha_i d_i, \tag{3.1}$$

where $\bar{\mathbf{w}}$ is the empirical mean $\mathbf{w}$ vector, d_i is the i-th principal direction and α_i is the learned scalar coefficient of the direction. In this project, I follow the suggestion of the authors to keep N as 512 for training on the CelebA dataset [20]. After that, the objective function used to optimize the training is represented as follows:

$$\mathcal{L}_{align}(T, \mathbf{c}) = \ell(T(G(\mathbf{w})), G(mix(\mathbf{c}, \mathbf{w}))), \tag{3.2}$$

$$\mathcal{L}(T, \mathbf{c}) = \mathbb{E}_{\mathbf{w}\sim\mathcal{W}}[\mathcal{L}_{align}(T, \mathbf{c}) + \lambda_{TV}\mathcal{L}_{TV}(T) + \lambda_I \mathcal{L}_I(T)] \tag{3.3}$$

where ℓ is a perceptual loss function, $\mathcal{L}_{TV}$ and $\mathcal{L}_I$ is the regularization terms, which are introduced with Spatial Transformer in Section 3.2.2, and the λ_{TV} and λ_L are the loss weighting.

3.2.1 Pre-trained GAN Model

Theoretically, any GAN architecture could be used in the GANgealing algorithm. Peebles *et al.* [6] opts to utilise StyleGAN2 [3] because it possesses an innate ability of style mixing. As shown in Figure 3-3, the latent code extracted from image B is fed into the generator's lower layers, while the latent code extracted from image A is fed into the generator's middle and upper layers. On account of this, the generated image retains the orientation and pose of image B, while the finer facial attributes are preserved from image A.

The GANgealing algorithm applies style mixing in label generation on the grounds that the target image y needs to share the appearance with $x = G(\mathbf{w})$. During generating target images, the target vector $\mathbf{c}$, which represents the pose and orientation, is fed into the first few layers, while the remaining layers are taken from $\mathbf{w}$ in order to ensure the facial attribute are identical to the training data. Hence, the target image y is expressed as the equation $y = G(mix(\mathbf{c}, \mathbf{w}))$.

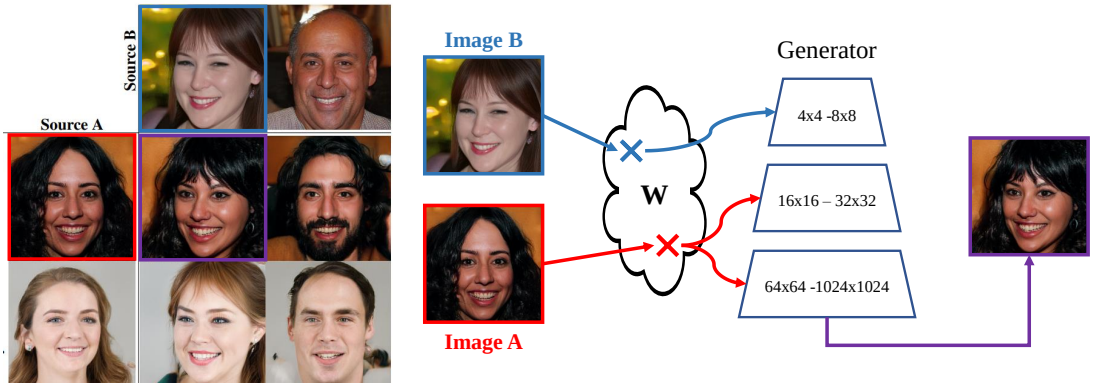


Figure 3-3: Examples and the mechanism of style mixing.

3.2.2 Spatial Transformer Networks

A Spatial Transformer module [7] is to actively spatially transform an image or feature map by providing a suitable transformation for each input sample. The structure of a Spatial Transformer is shown in Figure 3-4. The Spatial Transformer Network defined in GANgealing algorithm is actually a composition of 2 Spatial Transformers, i.e., a similarity Spatial Transformer T_{sim} and an unconstrained Spatial Transformer T_{flow}. T_{sim} performs the similarity transformation, such as rotation, translation and scaling, whereas T_{flow} performs the transformation based on an unconstrained dense flow field. The sampling grid w.r.t. the unconstrained dense flow field is defined as $g \in \mathbb{R}^{128\times128\times2}$. The output of T_{sim} is directly fed into T_{flow}, so the expression of the composed Spatial Transformer is $T = T_{flow} \circ T_{sim}$, and T is trained end-to-end. The authors also suggest two regularizers to encourage the flow to be smooth and not to be deviated:

$$\mathcal{L}_{TV}(T) = \mathcal{L}_{Huber}(\triangle_x g) + \mathcal{L}_{Huber}(\triangle_y g) \tag{3.4}$$

$$\mathcal{L}_I(T) = \|g\|_2^2, \tag{3.5}$$

where $\mathcal{L}_{Huber}$ denotes the Huber loss and $\triangle_x$ and $\triangle_y$ denote the partial derivative w.r.t. x and y coordinates under finite differences.

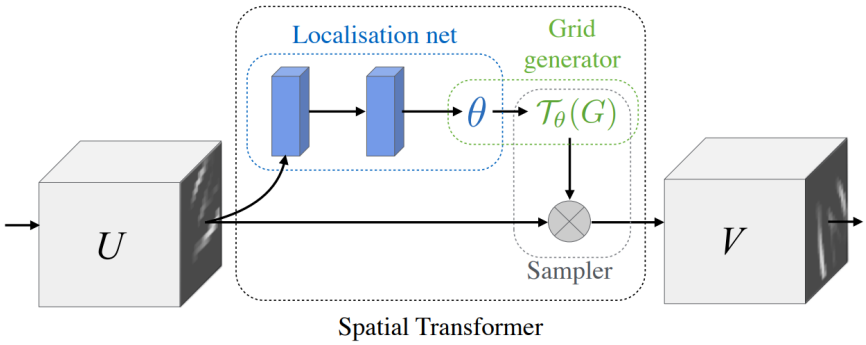


Figure 3-4: Overview of Spatial Transformer Network. Adopted from [7].

3.3 AttGAN

He *et al.* [9] proposed AttGAN as a framework for facial attribute editing, which can generate images with desired attribute modifications while preserving the identity and other image details. Compared to some of the more recent models in the GAN framework, AttGAN can be considered a relatively small and simple model. The overall pipeline for training and testing is shown in Figure 3-5. In this project, the testing framework of AttGAN is embedded in the GANgealing algorithm for performing facial attribute editing.

3.3.1 Testing Pipeline

In AttGAN, the input image is denoted as x^a where a is the corresponding binary attribute label with size n. In the testing framework, an encoder G_{enc} encodes the input into latent representation z. After modifying a to the desired attribute label b, the regenerated image $x^{\hat{b}}$ with the desired editing is formed by the decoder G_{dec} conditioned on b. The overall testing framework can be formulated as:

$$x^{\hat{b}} = G_{dec}(G_{enc}(x^a), b), \tag{3.6}$$

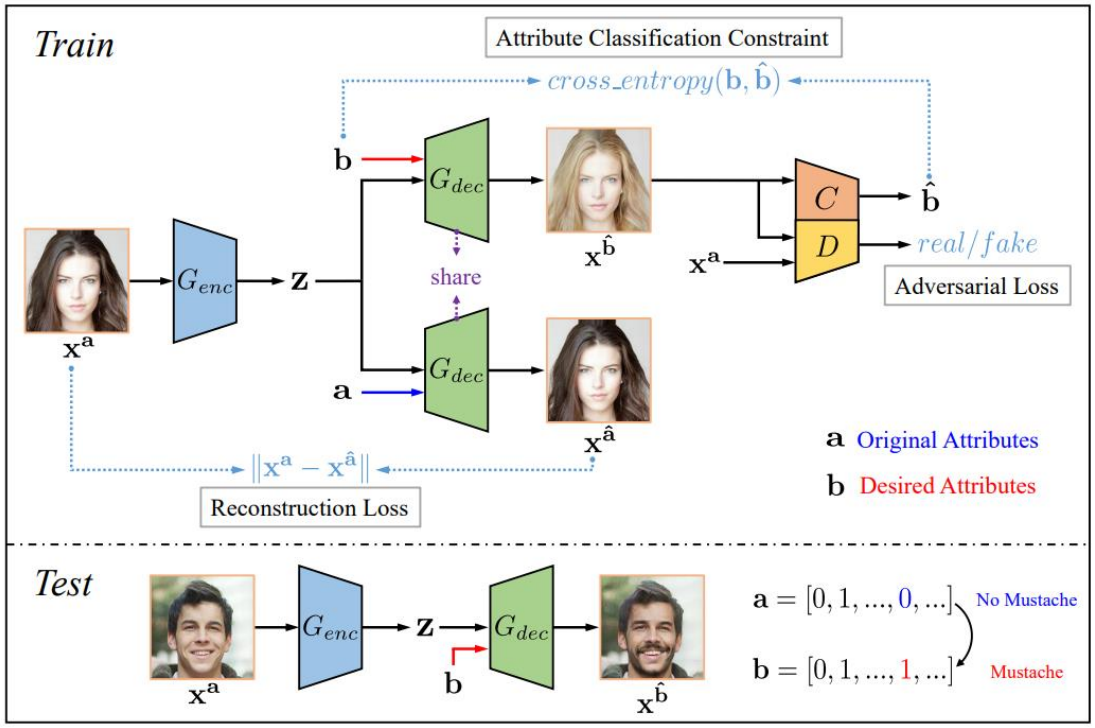


Figure 3-5: AttGAN model's pipeline. Adopted from [9].

3.3.2 Training Pipeline

The AttGAN training framework consists of three components: adversarial learning, attribute classifier constraint and reconstruction learning.

Attribute Classifier Constraint. The attribute classifier constraint can ensure the generator forms the edited image $x^{\hat{b}}$ with the desired attributes. It is formulated as:

$$\min_{G_{enc},Gdec} \mathcal{L}_{cls_g} = \mathbb{E}_{x^a \sim p_{data}, b \sim p_{attr}}[\ell_{bce}(x^{\hat{b}}, b)], \tag{3.7}$$

where p_{data} and p_{attr} indicate the distribution of real images and the distribution of attributes, and ℓ_{bce} indicates the binary cross entropy loss. However, this attribute classifier is learned on the input image x^a with the original attribute label a:

$$\min_{C} \mathcal{L}_{cls_c} = \mathbb{E}_{x^a \sim p_{data}}[\ell_{bce}(x^a, a)]. \tag{3.8}$$

Reconstruction Learning. The reconstruction learning guarantees the input image x^a is encoded with enough information of attribute-excluding details to the latent representation z. On the other hand, it enables the decoder to learn

the reconstruction of x^a by decoding z conditioned on the original attribute a. The reconstruction loss is formulated as follows:

$$\min_{G_{enc},Gdec} \mathcal{L}_{rec} = \mathbb{E}_{x^a \sim p_{data}}[||x^a - x^{\hat{a}}||_1]. \tag{3.9}$$

Adversarial Learning. The adversarial learning between the generator (G_{enc} and G_{dec}) and the discriminator (D) is used to make the edited image visually realistic. The adversarial loss is optimised via WGAN-GP [14]:

$$\min_{G_{enc},Gdec} \mathcal{L}_{adv_g} = -\mathbb{E}_{x^a \sim p_{data}, b \sim p_{attr}}[D(x^{\hat{b}})], \tag{3.10}$$

$$\min_{||D||_L \leq 1} \mathcal{L}_{adv_d} = -\mathbb{E}_{x^a \sim p_{data}}[D(x^a)] + \mathbb{E}_{x^a \sim p_{data}, b \sim p_{attr}}[D(x^{\hat{b}})] + \lambda_{gp}\mathbb{E}_{\hat{x} \sim p_{\hat{x}}}[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2], \tag{3.11}$$

where D is constrained to be the 1-Lipschitz function, λ_{gp} is set to 10, and $\hat{x}$ sampled from x^a and $x^{\hat{b}}$ with α uniformly sampled between 0 and 1:

$$\hat{x} = \alpha x^{\hat{b}} + (1 - \alpha)x^a. \tag{3.12}$$

Overall Objective Function. The objective function of AttGAN for the encoder and decoder is defined as:

$$\min_{G_{enc},Gdec} \mathcal{L}_{enc,dec} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{cls_g} + \mathcal{L}_{adv_g}, \tag{3.13}$$

and the objective function for the discriminator and the attribute classifier is defined as:

$$\min_{D,C} \mathcal{L}_{dis,cls} = \lambda_3 \mathcal{L}_{cls_c} + \mathcal{L}_{adv_d}, \tag{3.14}$$

where the λ_1, λ_2 and λ_3 are the hyper-parameters for balancing the losses, and most layers of the discriminator and the attribute classifier are shared. The network architectures of AttGAN are shown in Table 3-1.

Table 3-1: Network Architectures of AttGAN for 128+[2] Images. Adopted from [9]. Conv(d,k,s) and DeConv(d,k,s) denote the convolutional layer and transposed convolutional layer with d as dimension, k as kernel size and s as stride. BN is batch normalization, LN is layer normalization and IN is instance normalization.

Encoder (G_{enc})	Decoder (G_{dec})
Conv(64,4,2), BN, Leaky ReLU	DeConv(1024,4,2), BN, ReLU
Conv(128,4,2), BN, Leaky ReLU	DeConv(512,4,2), BN, ReLU
Conv(256,4,2), BN, Leaky ReLU	DeConv(256,4,2), BN, ReLU
Conv(512,4,2), BN, Leaky ReLU	DeConv(128,4,2), BN, ReLU
Conv(1024,4,2), BN, Leaky ReLU	DeConv(3,4,2), Tanh

Discriminator (D)	Classifier (C)
Conv(64,4,2), LN/IN, Leaky ReLU	
Conv(128,4,2), LN/IN, Leaky ReLU	
Conv(256,4,2), LN/IN, Leaky ReLU	
Conv(512,4,2), LN/IN, Leaky ReLU	
Conv(1024,4,2), LN/IN, Leaky ReLU	
FC(1024), LN/IN, Leaky ReLU	FC(1024), LN/IN, Leaky ReLU
FC(1)	FC(13), Sigmoid

Results and Discussion

4.1 Setting up environment

The experiment is conducted under the following environment:

- NVIDIA GPUs and NVIDIA drivers
- Python 3.8
- Pytorch 1.10.1
- CUDA 10.3 toolkit

4.2 Dataset

The datasets used in the experiment include CelebA [20] and CelebA-HQ [18]. CelebFaces Attributes Dataset (CelebA) [20] is a large-scale dataset which consists of 202,599 face images from 10,177 identities. It has provided 5 landmark locations and 40 binary attribute annotations for each image. The attributes provided include gender, eyeglasses, mouth open/close, moustache, smiling, etc. CelebA-HQ, which is a subset of CelebA, encompasses 30,000 high-resolution face images chosen and post-processed by [18]. Both datasets are widely used to train and evaluate the algorithm of face attribute recognition, face recognition, face detection, landmark (or facial part) localization, and face editing and synthesis.

4.3 Implementation and Analysis of GANgealing

In this section, I tried to implement and train the GANgealing algorithm [6]. As mentioned before, GANgealing can perform the global alignment for any object, but I only used the model for the alignment of human faces. Although the authors have provided a pre-trained model for human face alignment, I also trained the model again in order to fully understand how the algorithm works. After that, I used the model to test some external images and videos and analysed the strengths and weaknesses of this method.

4.3.1 Training

According to [6], I used the StyleGAN2 pre-trained on the CelebA dataset to generate training data pairs. The size of generated images is 128×128. The loss weighting λ_{TV} and λ_L in Eqn. 3.3 are set as 2500 and 1, respectively, and the perceptual loss function used is LPIPS [21]. The training is undergone with 1,500,000 iterations, which is updated by an Adam optimizer [22] with a learning rate of 0.001, β_1 0.9 and β_2 0.999. Besides, the target latent $\mathbf{c}$ is also optimized simultaneously during the training via learning scalar coefficients $\{\alpha_i\}_{i=1}^{512}$ in Eqn. 3.1. It is also updated by an Adam optimizer with a learning rate of 0.01, β_1 0.9 and β_2 0.999.

Figure 4-1 shows the examples of the synthetic training data pairs and the output images transformed by a well-trained STN. The last column of the figure shows the average images, each of which is an average image across all the instances. Figure 4-2 shows the average target images changed along with the learning on pose-determined latent code $\mathbf{c}$. After an iteration of around 150,000, the average target image basically remains unchanged.

Figure 4-1: Training Result Examples. The first and second rows show the example of the training data pairs, whereas the last row shows the output images transformed by a well-trained STN. The average images are shown in the last column.

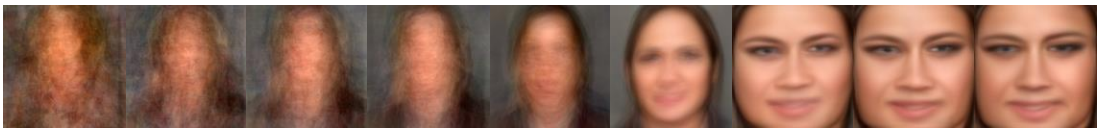


Figure 4-2: Average target images changed along the iterations. The corresponding iteration index is shown at the bottom of each image.

4.3.2 Testing

As mentioned in previous sections, a pre-trained StyleGAN2, which was trained on the CelebA dataset, is used to generate human face images for training the STN. Hence, the images of the CelebA dataset are *seen* by the model. The *unseen* images are any images excluded from the dataset, and I collect *unseen* images from the internet. Figure 4-3 shows the examples of *seen* (first row) and *unseen* (third row) images, as well as the corresponding results aligned by the well-trained STN (second and last row). The alignment effect of both group images is acceptable. However, with careful observation, we can find that some of the aligned images are flipped horizontally. This is because, during testing, I used the default setting, which let the model compare the transformation of flipped and unflipped input images. Then, it chooses whichever provides the smoother flow by comparing their total variation loss L_{TV}, as represented in Eqn. 3.4.

Figure 4-3: Testing images from and outside of the CelebA dataset. The top-row images are from the CelebA dataset, while the third-row images are collected from the internet. The second and last rows display the corresponding aligned results.

The total variation loss L_{TV}, indicated in Eqn. 3.4, play an essential role in the GANgealing algorithm. It represents the alignment ability of the human face in an image. The higher the loss, the lower performance of the alignment. Figure 4-4 represents the input images and the corresponding aligned images with different L_{TV}. I noticed from the figure that as the face pose in the yaw increases, the L_{TV} also increases, and the aligned images become more and more misaligned.

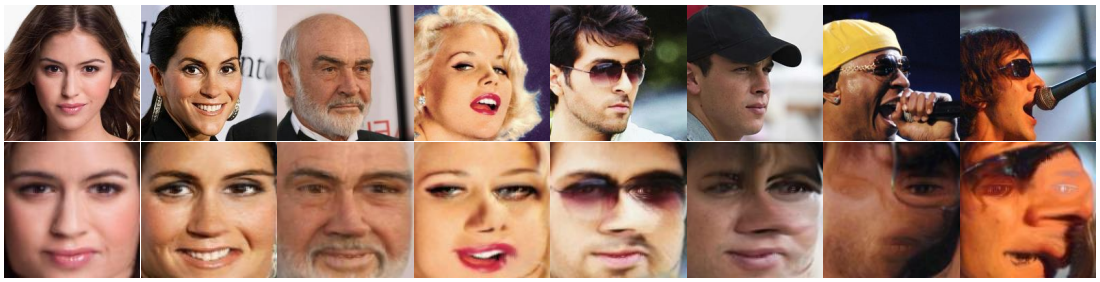


Figure 4-4: The input images and the corresponding images with different total variation loss. The values of the total variation loss is shown below the images.

4.3.3 Application and Analysis

The STN of GANgealing can not only perform the forward transformation but also in a reverse direction. In such a way, we can perform some image editing on the aligned image. Then, the transformer is able to propagate the editing back to the initial image. This application of the transformer also works well for augmented reality, i.e., performing the alignment and editing on realistic videos. Peebles *et al.* [6] perform the editing by simply adding some objects, such as a dense label, a moustache and a pair of pokemon stickers, to human face images or videos. In addition to the objects provided in [6], I also prepared two more objects, an eye mask and an anger symbol. As shown in Figure 4-5, the position of those objects must be based on an average-aligned face to ensure precise propagation. Figure 4-6 depicts those objects that are wrapped back to the face images via the reverse transformation.

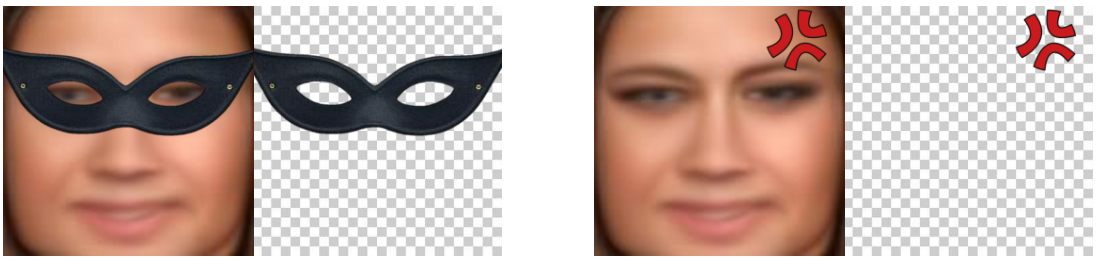


Figure 4-5: Preparation of propagating objects. The position of the object must be based on the average-aligned image.

Figure 4-6: Examples of objects propagating to images.

The Spatial Transformer has a little constraint which can only handle square images. All inputs of previous examples are undergone a centre crop before passing to the transformers. Therefore, all faces are basically in the centre of the image. If the face is not in the middle of the image, we can pad the image with zero. For example, the input image in Figure 4-7 and Figure 4-8, in which the human face is at the right and the upper part of the images, are also alignable. The dense label can also be propagated properly to the images.

Input Image Aligned Image Edit Propagation

Figure 4-7: Human face is at the right part of the image.

Figure 4-8: Human face is at the upper part of the image.

The GANgealing algorithm is also robust to the input image with low resolution. As shown in Figure 4-9, I down-sample an image to different resolutions, for which the initial resolution is 1080×720. Before passing to the transformer, the images are padded with zero and up-sampled again to 1024×1024, as shown in first-row images of the figure. From the second and third rows of the figure, we can see that the facial alignment is not much affected by the resolution, but the outline of the propagated mask is gradually inaccurate.

Figure 4-9: Input with different resolution images.

Besides, the facial alignment may be faulty when the human face is smaller than 10% of the input image's size. As shown in Figure 4-10, the human face is smaller than the red square, and the size of the red square is exactly 10% of the input image's size. In the aligned image shown in Figure 4-10, part of the hair at the bottom left is transformed into a part of the face. When the background of the input image is simultaneously complicated, as shown in Figure 4-11, the transformer might form a face from the background. When the face size in the same input image is increased by simply applying a centre crop instead of padding the with zero, the face can be aligned precisely, as represented in Figure 4-12.

Figure 4-10: Human face is small than 10% of the input image's size.

Figure 4-11: Human face is small than 10% of the input image's size and the background is complicated.

Figure 4-12: Human face is greater than 10% of the input image's size, but the background is complicated.

The algorithm can also perform the alignment well on each individual video frame, as shown in Figure 4-13. Even the human in the video is running, as shown in Figure 4-14, the alignment and edit propagation are not affected by the violent motion.

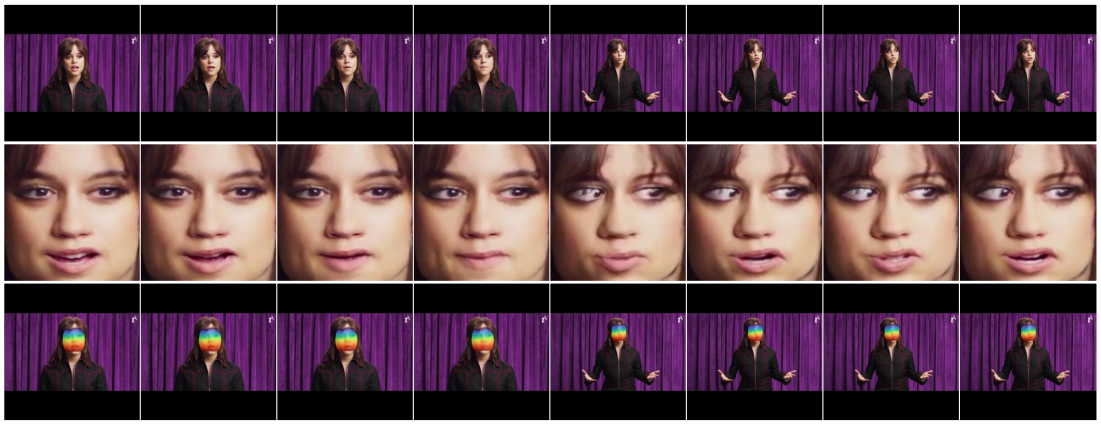


Figure 4-13: Frames from a human face video with a static motion.

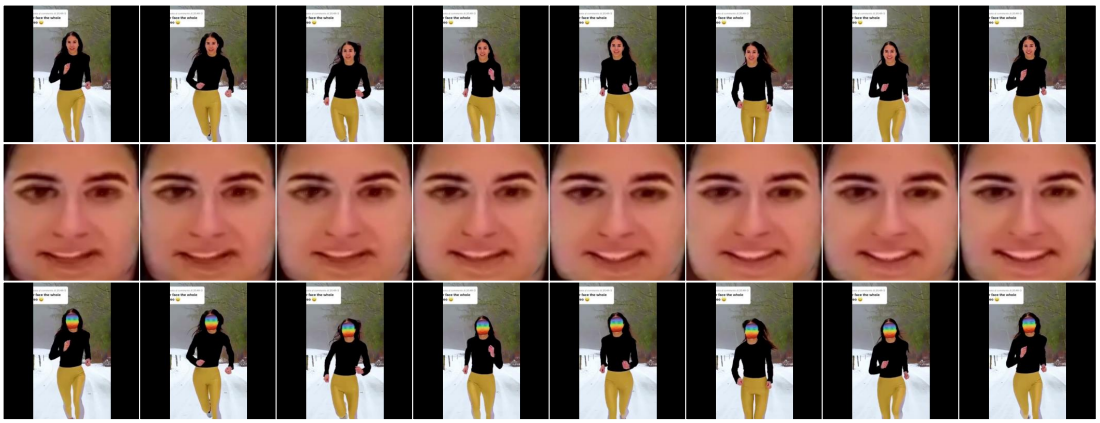


Figure 4-14: Frames from a human face video with a violent motion.

4.4 Implementation of AttGAN

In this section, I trained the AttGAN [9] on the images that had been pre-processed by GANgeaing. After that, I evaluated the performance of the attribute editing done by AttGAN from the perspective of visuality and quantitative.

4.4.1 Training Details

Data Preparation. The training data of AttGAN is the aligned images that have been processed by GANgealing. Since GANgealing can process high-quality images, I tried to retain this capability in AttGAN and I used a resolution of 384×384 for the input of AttGAN. The training dataset of AttGAN I used in the following experiments is CelebA-HQ instead of CelebA. Figure 4-15 illustrates that the pale skin face image reconstructed by an AttGAN model, which is trained on GANgealing-processed CelebA images, contains some undesired artifacts. Such a problem was solved when utilising a high-resolution dataset, CelebA-HQ, to train the AttGAN model. In addition, I also filtered the dataset by dropping the images in which the flow's L_{TV} is greater than 3.0×10^{-5} during the GANgealing alignment because those images cannot be aligned properly, as shown in Figure 4-4. After splitting CelebA-HQ into three sets (training, validation, and testing) at ratios of 0.9, 0.02, and 0.08 respectively, I utilised the training and validation set to train our model and then evaluated its performance using the testing set.

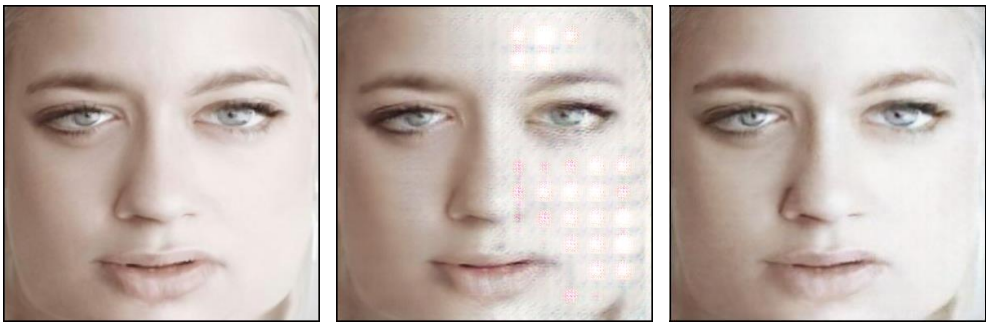


Figure 4-15: Some unwanted artifacts appear in the pale skin image reconstructed by the CelebA-trained AttGAN model.

Training Parameter. The training parameters are basically the same as [9]. To ensure the loss values are in the same order of magnitude, the loss weighting λ_1, λ_2 and λ_3 in Eqn. 3.13 and Eqn. 3.14 are set to 100, 10 and 1 respectively. The training is undergone with 200 epochs and batch size of 32, which is updated by an Adam optimizer [22] with a learning rate of 0.0002, β_1 0.5 and β_2 0.999.

Attribute Labels. The attribute annotations provided by CelebA or CelebA-HQ are up to 40 and the authors of AttGAN [9] used 13 of them in their research. On the ground that the GANgealing-aligned images crop out hairs, ears, part of the forehead and part of the chin, which limited the number of attributes that can be edited by the AttGAN, I can only choose up to 5 attributes from the original paper, which includes 'Bushy Eyebrows', 'Eyeglasses', 'Mouth Slightly Open', 'Mustache' and 'No Beard'. To fully demonstrate the capability to modify multiple attributes by the same model, I add 3 more attributes for this project, including 'Smiling', 'Wearing Lipstick', and 'Rosy Cheeks'.

4.4.2 Visual Analysis

The model trained on GANgealing-aligned images can perform well on single-attribute editing and also multiple-attribute combination editing, which benefited from the accurate modelling of the connection between the attributes and the latent representation, as shown in Figure 4-16 and Figure 4-17. Figure 4-16 depicts some results of single faces' attributes edited by AttGAN. The first row is the input images, the second row is the images reconstructed by AttGAN with their original attribute labels, and the other rows show the images where

a specified attribute was edited. Besides, Figure 4-17 depicts some results of multiple attributes edited simultaneously. However, the drawback of editing multiple attributes at the same time is the intensity of each attribute editing is affected.

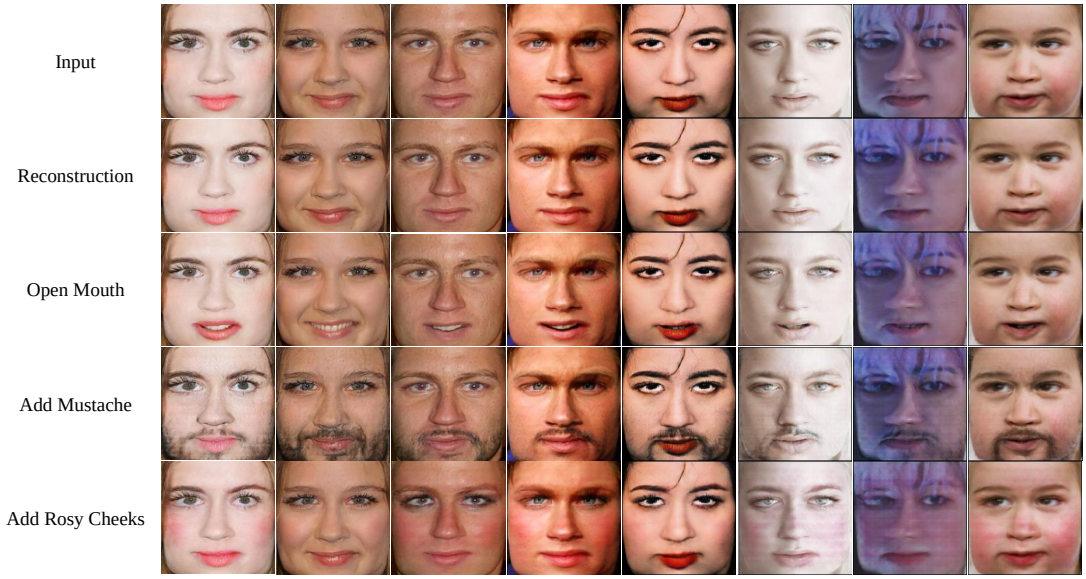


Figure 4-16: Editing specified attributes of aligned images.

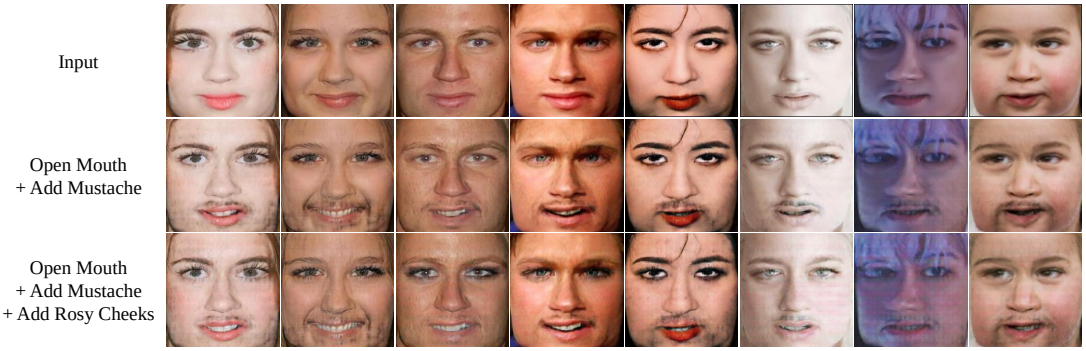


Figure 4-17: Editing multiple attributes of aligned images simultaneously.

4.4.3 Quantitative Analysis

In this section, I evaluated the facial attribute editing accuracy of the AttGAN. Following [9], I trained an attribute classifier independently on the GANgealing-aligned CelebA-HQ. However, the network architecture and loss function is similar to the attribute classifier of AttGAN, which can refer to Table 3-1 and Eqn. 3.8 respectively. I used the training set same as the training of AttGAN to train this classifier and it achieves an average accuracy of 94.07% per attribute on the testing set.

Table 4-1: Evaluation metrics for attribute editing. ↑ denotes higher the better, and ↓ denotes lower the better.

Attribute	Attribute Editing Accuracy ↑	Attribute Removing Error ↓	Attribute Adding Error ↓	Attribute Preservation Error ↓
Bushy Eyebrows	95.56%	6.37%	3.93%	3.90%
Eyeglasses	86.15%	4.39%	14.27%	0.04%
Mouth Slightly Open	97.74%	1.02%	3.38%	1.64%
Mustache	71.48%	11.54%	29.38%	2.14%
No Beard	92.70%	8.40%	2.93%	1.00%
Smiling	88.19%	18.13%	5.89%	1.76%
Wearing Lipstick	51.15%	48.19%	49.64%	14.85%
Rosy Cheeks	43.59%	14.18%	61.20%	5.51%
Average	78.32%	14.03%	21.33%	3.86%

Table 4-1 illustrates the attribute accuracy of AttGAN which is evaluated by the independent attribute classifier. If the classifier predicts that the attribute of an edited image is the same as the desired attribute, then it is considered a correct editing, otherwise an incorrect one. In addition to attribute editing accuracy, I further evaluated the error rate of AttGAN removing or adding a specified attribute, so we can know which attribute is relatively hard to be added or be removed. Besides, I also evaluated the error in preserving the other attribute while modifying a single attribute.

From the table, we can find that the accuracy of changing the attributes of 'Wearing Lipstick' and 'Rosy Cheeks' dropped significantly, and we could further find out the reason from the attribute removing error and attribute adding error. Since all the accuracy, the attribute removing error and the attribute adding error are around 50%, we know that the AttGAN might not

be suitable for modifying the attribute of 'Wearing Lipstick'. Different to 'Wearing Lipstick', the AttGAN is still able to modify the attributes of 'Rosy Cheeks' but it is in the wrong direction, especially when adding the rosy cheek to faces. We can compare a real rosy cheek image with a generated image, which is edited by AttGAN but it is predicted as false in holding a rosy cheek. As shown in Figure 4-18, the entire facial area of the generated image is red, not just the cheek. Furthermore, this erroneous editing of 'Rosy Cheeks' would also impact the preservation of the attribute 'Wearing Lipstick'. As shown in Figure 4-19, the lip of the generated image also becomes red. The error of the 'Rosy Cheeks' preservation is the only one greater than 10% as shown in Table 4-1.

Figure 4-18: Comparing the rosy cheek of the real and generated images.

Figure 4-19: Colour of the lip is changed when editing of the rosy cheek.

4.5 Incorporate AttGAN in GANgealing

After an AttGAN is well trained for facial attribute editing, I tried to wrap back those editing to the original images, using the pipeline shown in Figure 3-1. Figure 4-20 and Figure 4-21 depict the results of propagating back the editing in Figure 4-16 and Figure 4-17, respectively. More editing examples can be found in Appendix A.1. The results of editing propagating are acceptable, but the drawback is the chin cannot be fully edited, because the chin is not fully aligned by GANgealing. This drawback is obvious when adding mustache or bread.

Figure 4-20: Propagation of a specified attributes editing.

Figure 4-21: Propagation of multiple attributes editing.

In addition, this facial attribute editing method works well in modifying a video with a human face. Figure 4-22 depicts video frames with a human face which had been added on a mustache and rosy cheeks respectively using this editing pipeline.

Figure 4-22: Editing in a video with a human face.

4.5.1 Importance of Global Alignment.

As mentioned in Section 3.2.2, The Spatial Transformer Network T of GANgealing consists of two Spatial Transformers, i.e., a similarity Spatial Transformer T_{sim} and an unconstrained Spatial Transformer T_{flow}. The output of T_{sim} are faces that have been aligned and cropped but the pose and orientation are preserved. After that, the output is directly fed into T_{flow}, where T_{flow} is responsible to unify all the faces to a single view.

In this section, I want to evaluate the importance of the training model on a dataset which has been aligned with the same view, by comparing the two AttGAN models trained on the dataset which was pre-processed without T_{flow} and the dataset which was pre-processed by with T_{flow}. As shown in Figure 4-23, the AttGAN model that trained on the dataset without preprocessed T_{flow} cannot generate complicated accessories like eyeglasses clearer. In the attribute editing task, adding or removing eyeglasses is relatively harder than modifying other attributes. When all facial features are unified to the same view, this could be easier for the model to add on or remove such complicated accessories.

Figure 4-23: Comparing the result with and without the global alignment. T denotes the operation which performs global alignment, whereas T_{sim} denotes the operation which performs simple cropping and alignment.

Conclusion and Future Work

This report describes the application of GAN models in the task of editing human facial attributes using visual alignment. The GANgealing algorithm is used to align images in a dataset globally. This algorithm relies on a pre-trained StyleGAN2 to generate training data and labels for training a Spatial Transformer Network (STN) that learn geometric transformations for joint alignment. The GANgealing algorithm was used to align human faces, and the aligned images were then used to train AttGAN, a simple model that utilises the WGAN. To streamline the process, AttGAN was integrated into the GANgealing algorithm, allowing for alignment, editing, and editing propagation to be performed in a single application.

This project, to be precise, studies the application of GANgealing only in facial feature editing. Due to the cropping done by GANgealing, the number of features that can be edited is limited. On the other hand, facial expression modification is also a sub-task of facial attribute editing. In the future, we can try this editing method for facial expression modification. We can find a suitable model for changing the facial expression and train it on a facial expression dataset to replace the AttGAN.

In addition, AttGAN is also a relatively simple model for the attribute editing task. There are some improved versions of AttGAN or other advanced editing models published. Therefore, we can also try using other facial feature modification models to improve the performance of this method.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[2] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4401–4410.

[3] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8110–8119.

[4] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 104–12 114, 2020.

[5] T. Karras, M. Aittala, S. Laine, *et al.*, "Alias-free generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 852–863, 2021.

[6] W. Peebles, J.-Y. Zhu, R. Zhang, A. Torralba, A. A. Efros, and E. Shechtman, "Gan-supervised dense visual alignment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 470–13 481.

[7] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," *Advances in neural information processing systems*, vol. 28, 2015.

[8] A. Nickabadi, M. S. Fard, N. M. Farid, and N. Mohammadbagheri, "A comprehensive survey on semantic facial attribute editing using generative adversarial networks," *arXiv preprint arXiv:2205.10587*, 2022.

[9] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "Attgan: Facial attribute editing by only changing what you want," *IEEE transactions on image processing*, vol. 28, no. 11, pp. 5464–5478, 2019.

[10] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[11] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[12] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," *Advances in neural information processing systems*, vol. 29, 2016.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, PMLR, 2017, pp. 214–223.

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[15] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[16] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8789–8797.

[17] C. Ledig, L. Theis, F. Huszár, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[18] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[19] E. G. Learned-Miller, "Data driven image models through continuous joint alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 2, pp. 236–250, 2005.

[20] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, Dec. 2015.

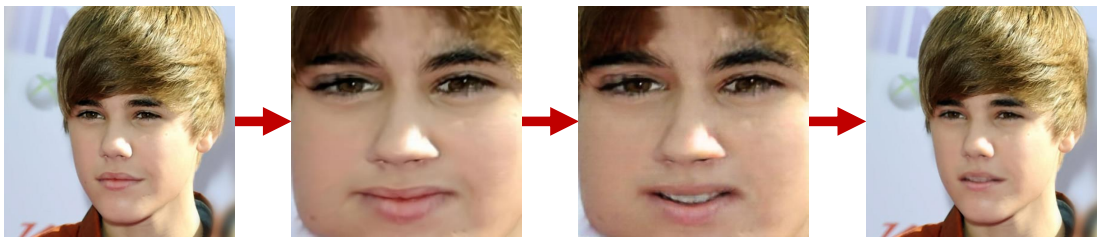
[21] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Appendix

A.1 Examples of All Attribute Editing

This section shows all the attribute editing excluding 'Wearing Lipstick', because it has been shown that the attribute is unable to be changed by AttGAN in Section 4.4.3. In the following figures, some editing cannot be performed obviously, including 'Remove Rosy Cheeks', 'Remove Eyeglasses' and 'Remove Smiling'.



(a) Open Mouth

(b) Close Mouth

Figure A-1: Mouth Slightly Open

(a) Add Mustache

(b) Remove Mustache

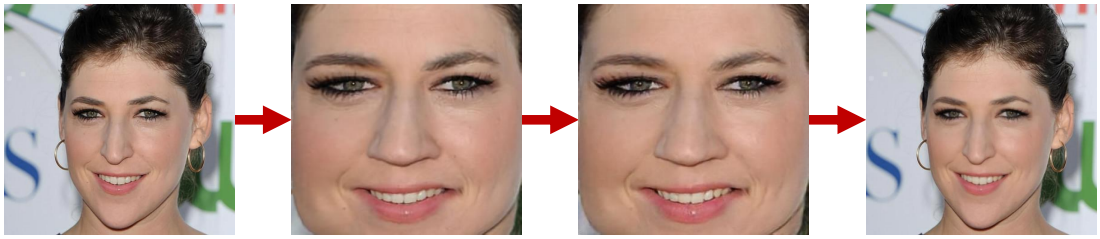
Figure A-2: Mustache

(a) Add Beard

(b) Remove Beard

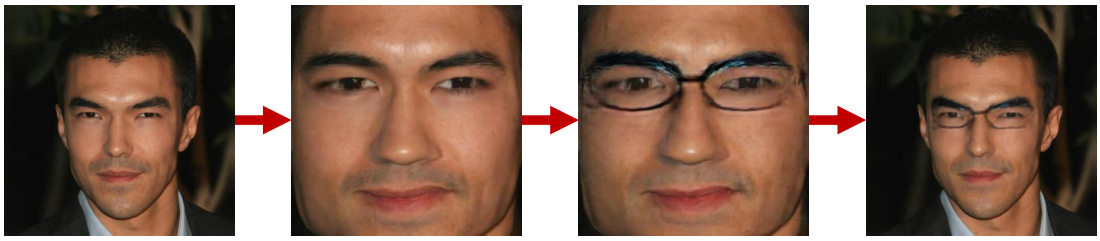
Figure A-3: No Beard

(a) Add Rosy Cheeks

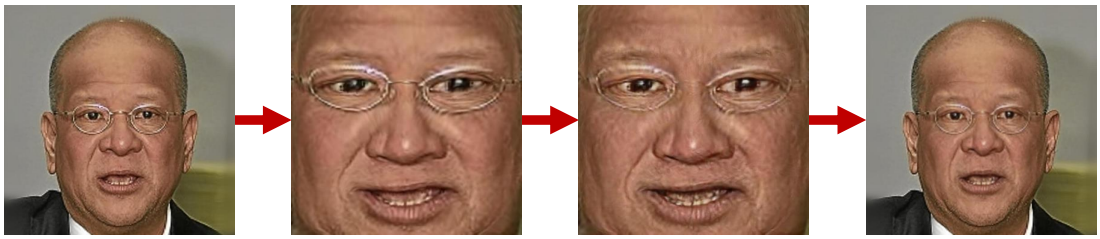


(b) Remove Rosy Cheeks

Figure A-4: Rosy Cheeks

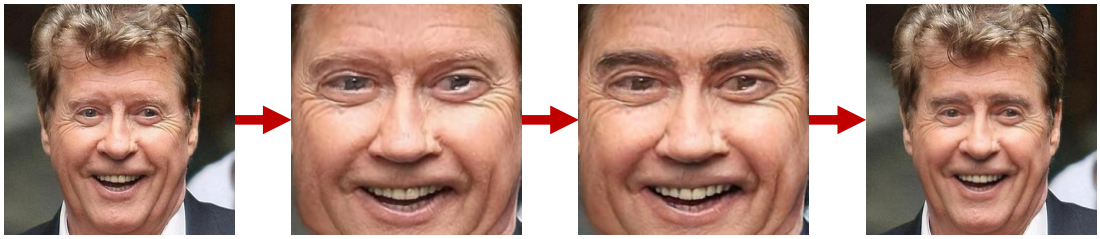


(a) Add Eyeglasses



(b) Remove Eyeglasses

Figure A-5: Eyeglasses



(a) Add Bushy Eyebrows

(b) Remove Bushy Eyebrows

Figure A-6: Bushy Eyebrows

(a) Add Smiling

(b) Remove Smiling

Figure A-7: Smiling