Applications of Generative Adversarial Networks in Hairstyle Transfer

Capstone Project Report

Student
Pham Cao Bang
Ta Dang Khoa

Instructor
Dr.Phan Duy Hung

Bachelor of Computer Science
Hoa Lac Campus – FPT University
May 4, 2021

ACKNOWLEDGEMENT

Firstly, we would like to thank our instructor, Dr. Phan Duy Hung, for his patience and time, and for instructing and advising our enthusiastically. Secondly, we would also like to thank our University, FPT, for giving our a good environment to study and grow over the years. Finally, we always remember our family's encouragement and support. We want to give them a special thanks, because they are the motivation for us to improve ourself every day.

ABSTRACT

Recently, Generative Adversarial Networks (GANs) extensively studied, and a range of GANs architectures and related methods proposed. The evolution of GANs makes a significant impact in the areas of computer vision; thus, significant advances have been made, for instance, plausible image generation, image-to-image translation, facial attribute manipulation, and related domains. Most recent works have focused on many domains instead of focusing on a single domain of GANs. In this works, we want to focus on the applications in a single area named facial attribute manipulation, an exciting area of GANs. We apply the state-of-the-art architectures and methods of GANs in facial attribute manipulation for editing the hairstyle of a given facial image.

Keywords: Generative Adversarial Networks, GANs, Hairstyle Transfer, facial attribute manipulation.

TABLE OF CONTENTS

List of Figures

1 INTRODUCTION

1.1 Problem and motivation

In the past few years, Generative Adversarial Networks have significant success in the quality and resolution of the images synthesized, especially in style transfer. Numerous works try to improve and explore more different applications of GANs and specific results achieved. Motivated by the success of StyleGAN and InterFaceGan, where stochastic variation incorporated in the realistic-looking synthesized images in StyleGan paper, and the notion brought up about the existence of a hyperplane in the latent space serving as the separation boundary for any binary semantic in InterFaceGan paper, we propose to focus on one of the most practical variations - hairstyle.

1.2 Related works

Generative Adversarial Networks (GANs) [1] are attracting many researchers in the deep learning community because of the challenges and interesting problems, which GANs provide. GANs is a member of the generative models family in machine learning. GANs offer advantages compared to other generative models, e.g., variational autoencoders, such as handling sharp estimated density function, efficiently generating desired samples, eliminating deterministic bias, and with good compatibility with the internal neural architectures. These properties of GANs help this network's success, especially in the field of computer vision, such as plausible image generation, image-to-image translation, image super-resolution, facial attribute manipulation, etc.

StyleGAN [2] proposes an alternative generator architecture, which borrowed from style transfer literature, for generative adversarial networks. The new architecture can learn and unsupervised separation high-level attributes. In the new style-base generator architecture, the input vector is mapped to an intermediate latent space W. This new latent space allows the features vector to be disentangled with the probability density of the training data.

StyleGAN Encoder [4] is an encoder network that fed a series of style vectors directly generated by this network into a pre-trained StyleGAN generator, forming the extended W latent space. This encoder model uses perceptual loss, a special loss that can calculate how similar two images coincide with human judgment. By modifying the vectors, this research provides a way to change certain hair elements.

InterFaceGan is a framework that interprets the latent semantics learned by GANs for semantic face editing; this framework can manipulate gender, age, expression, etc. The authors found that disentangled representation after linear transformations learned

from the latent code of well-trained generative models. This property allows the disentanglement between various semantics explored and some entangled semantics decoupled with subspace projection, leading to more precise control of facial attributes. The framework provides single and multiple attributes manipulations (conditional manipulation), which allow us to manipulate single attribute or multiple attributes.

Affine Transformation is a geometric transformation. In this transformation, collinearity and ratios of distances are preserved, which means the correlation of the source object before and after transformation is the same. A range of perspective distortions can be corrected by applying an affine transformation, which transforms the measurements from the ideal coordinates to those used.

Face Landmark Detection [5] is a framework that could locate landmark points in images of faces, which is useful for aligning face images and detecting special points for transform tasks.

Recent works mention synthesis image generation, semantic editing image, geometric transformation, and critical points detection. However, the challenges are how we can use all this technique to build a system that can transfer the hairstyle of the given image. In this works, we proposed methods that take advantage of these recent works to develop systems to solve the hairstyle transfer problem.

1.3 Contribution

In this works, we apply some methods, which are based on the state-of-the-art techniques of GANs, to develop systems that can transfer hairstyle of given images. With the experimented results of our systems, we propose a comparison between those results to evaluate our systems efficiently.

In the first system, we develop an end-to-end process based on Stylegan and Interfacegan; we provide some hair attributes, for instance, wavy, straight, bald, hair color, etc. StyleGan acts as an encoder network, which takes images as input and output latent vectors corresponded to each image.

In the second system, we develop an end-to-end process based on segment, transform, Stylegan methods. Our system can completely overwrite a hairstyle photo onto the target face image without altering other facial properties.

2 BACKGROUND

2.1 Generative Adversarial Networks

Generative Adversarial Networks, which was proposed by Goodfellow et al. (2014), contain two main parts: Generator (G) and Discriminator (D), which shown in Fig.2.1. To learn the generator's distribution p_g over data x, the authors define a prior on input noise variables $p_z(z)$, then represent a mapping to data space as $G(z;\theta_g)$ where G is a differentiable function represented by a multilayer perceptron with parameters θ_g. The authors also define a second multilayer perceptron $D(x;\theta_d)$ that outputs a single scalar. D_x represents the probability that x came from the data rather than p_g. D was trained to maximize the probability of assigning the correct label to both training examples, and samples from G. Authors simultaneously minimize $log(1 - D(G(z)))$ by training G. In other words, D and G play the following two-player minimax game with value function V (G, D):

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

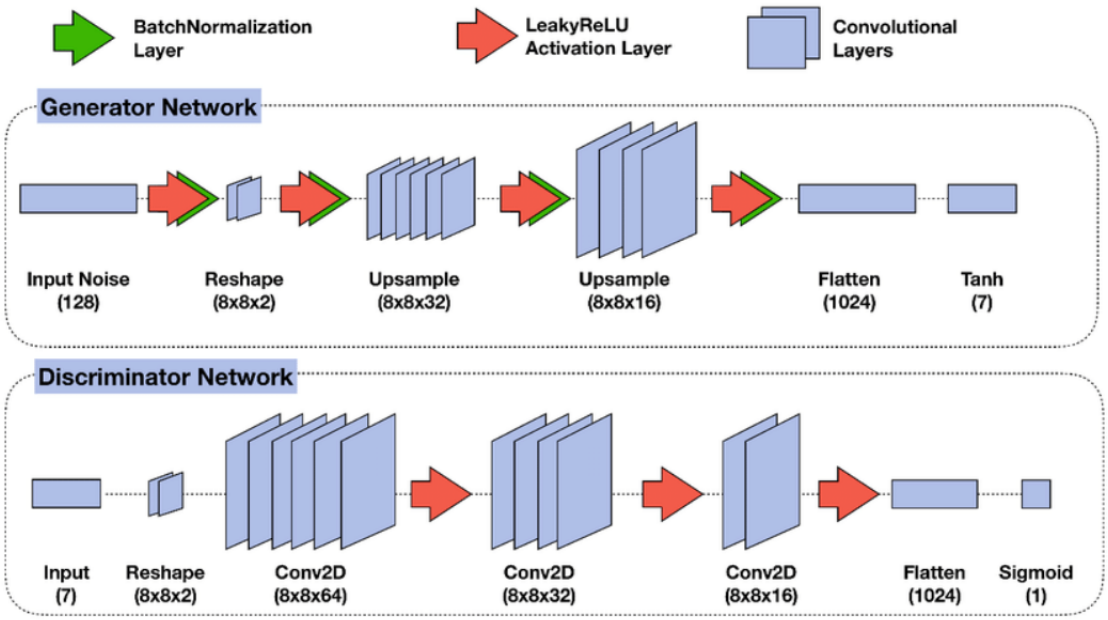

Figure 1: GANs architecture

2.2 Stylegan

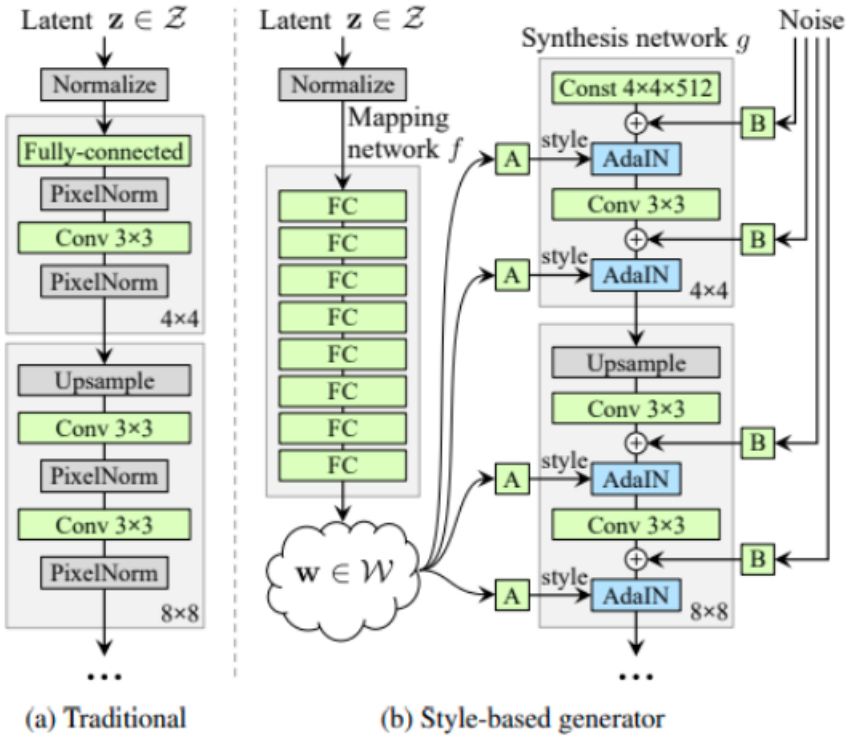

Figure 2: Stylegan architecture

The most significant difference between Stylegan and traditional GANs is that the traditional GANs generator uses the input latent code directly, and Stylegan's generator architecture maps the input to an intermediate latent space W. This overcomes a major weakness of traditional GANs, which is the input latent space must follow the probability density of the training data. Therefore the new style-based generator admits a more linear, less entangled representation of various factors of variation.

The new latent code W controls the generator through adaptive instance normalization (AdaIN) at each convolution layer. Gaussian noise is added after each convolution to get stochastic detail.

Linear separation of the new intermediate latent space makes it possible to change some of the image's properties without affecting the rest. Image quality tests, as well as linear separability, were tested on traditional GANs and Stylegan networks. Furthermore, the results show that traditional GAN generator architecture is inferior in every way compared to style-based design.

2.3 Perceptual loss

In many areas of computer science, the ability to compare data items is perhaps the most fundamental operation, for instance, compare binary patterns by Hamming distance, compare text files by edit distance, compare vectors by Euclidean distance, etc. However, assessing the perceptual similarity, which measures how similar two images are in a way that coincides with human judgment, between two images is quite complex.

The easiest way is using per-pixel measures, such as l_2 Euclidean distance. For example, one method for solving image transformation tasks is to use a per-pixel loss function to calculate the difference between output and ground-truth images when training a feedforward convolutional neural network in a supervised manner. This approach has been used for super-resolution by Dong et al. [6], for colorization by Cheng et al. [7], for segmentation by Long et al. [8], and for depth and surface normal prediction by Eigen et al. [9]. The approaches require only a forward pass through the trained network and provide efficiency at test-time. Nevertheless, pixel-wise is not independent, so these measures are insufficient for images, which are structured. For example, blurring images causes small perceptual but large l_2 change.

	Origin image	Blur 3x3	Blur 5x5	Blur 7x7
Image				
L2 loss	0	6000.650881362788	7041.8693540848935	7692.398455618378

Figure 3: Blurring a images causes small perceptual but large l_2 change

Additionally, there have been numerous perceptually motivated distance metrics proposed, such as SSIM [10], MSSIM [11], FSIM [12], and HDR-VDP [13]. These methods use mathematical formulas for the calculation. However, constructing a perceptual metric is challenging because human judgments of similarity depend on high-order image structure, are context-dependent, and may not constitute a distance metric. So these measures are inefficient.

In parallel, recent research has shown that high-quality images produced using perceptual loss functions base on differences between high-level image feature represen-

tations derived from pre-trained convolutional neural networks rather than differences between pixels. This strategy applied to feature inversion by Mahendran et al. [14], to feature visualization by Simonyan et al. [15] and Yosinski et al. [16], and to texture synthesis and style transfer by Gatys et al. [17]. These approaches are slow because it requires solving an optimization problem, but it makes the model more accurate.

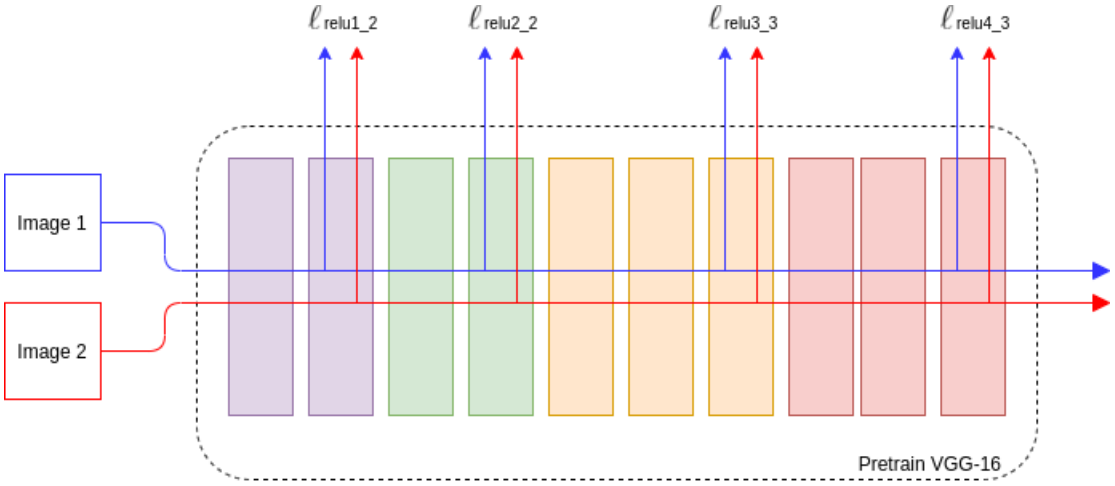

Figure 4: Perceptual loss model by minimizing a weighted combination

2.4 Resnet

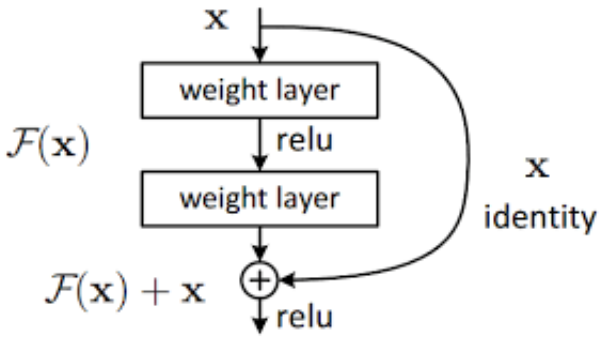

Figure 5: A Building Block of Residual Network

Conv layers then fully connected (FC) layers usually used; we call them plain networks. When the plain network is more extensive, the problem of vanishing and exploding gradients occurs. To solve this problem, He et al. propose skip connection technique in Resnet [18], which add input of current layer to the few layers after that layer. Detail of residual block shown in Fig.5

This skip connection allows information from the previous layer to transfer to the following few layers; thus, input information can be held when networks become deeper. Besides, even if gradient vanishing occurs, we still have the identity x to transfer back to earlier layers.

2.5 StyleGan Encoder

Recent GANs could generate very high-quality images, and researchers have created many good applications with them. This motivates research into embedding algorithms that embed a given photograph into a GAN latent space. And StyleGanEncoder is an encoder network that directly generates a series of style vectors. These vectors belong to the extended W latent space of Stylegan, which allows for local modifications such as missing regions and locally approximate embeddings.

This Stylegan-encoder algorithm is implemented in two steps: i) passing the input image through a pre-train neural network to get a latent estimation code; ii) optimizing the latent estimation code to match the input image.

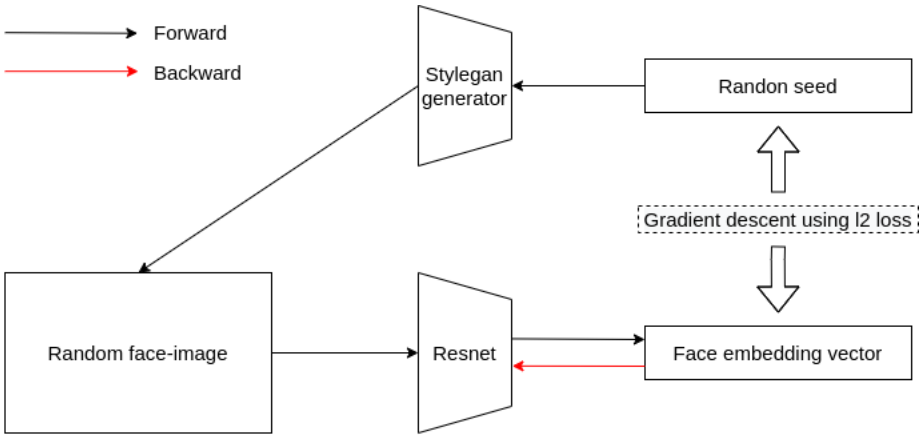

Figure 6: Resnet training process

In step i, we using a pre-train resnet to get a high semantic meaning latent space vector of an input image. This resnet model is trained by a large dataset, which is

generated from the Stylegan generator. The process of training this resnet is in Fig. 6. After this step, we could get a meaning latent space vector of input image, but the result image, which is created by this vector and Stylegan generator, has many many features that are not the same as the original image. So that, we need step ii, which optimizes this latent to match the input image.

In step ii, the input image will be provided to the pre-train resnet network in step i to estimate the latent code, and the Stylegan generator will use it to generate an initial prediction of the input image. Then, both the original input image and its initial conjecture are sent to a perceptual loss model, which is using a weighted combination technique with a pre-train VGG-16. We also use per-pixel loss and MS-SIM loss to calculate the similarity between the initial guess image and the input image. With this process, we can find latent representation in the Stylegan latent space of any real images. The process of encoding an image is in Fig. 7.

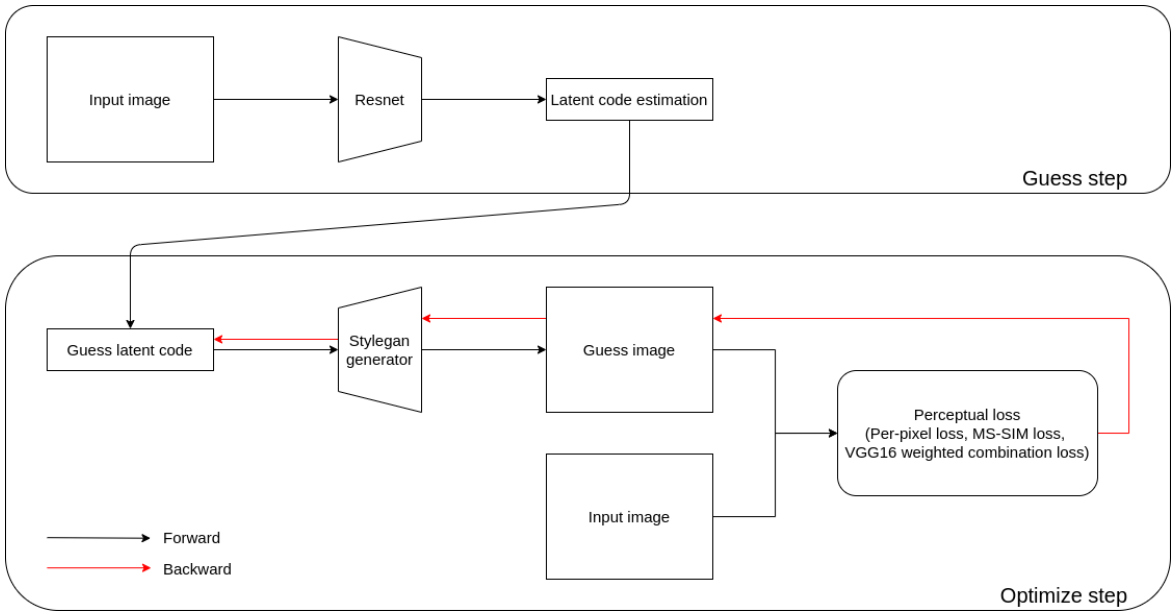

Figure 7: Stylegan encode process

2.6 InterFaceGan

Shen et al.(2020) propose a framework named InterFaceGan, which first provides a rigorous analysis of the semantic attributes emerging in the latent space of well-trained

GAN models and then constructs a manipulation pipeline of leveraging the semantics in the latent code for facial attribute editing.

2.6.1 Semantics in the Latent Space

Given a well-trained GAN model, the generator can be formulated as a deterministic function $g : \mathcal{Z} \longrightarrow \mathcal{X}$. Here, $\mathcal{Z} \subseteq \mathbb{R}_d$ denote the d-dimensional latent space, for which Gaussian distribution $\mathcal{N}(0, I_d)$ is commonly used. $\mathcal{X}$ stands for the image space, where each sample x possesses certain semantic information, like gender and age for face model. Suppose we have a semantic scoring function $f_S : \mathcal{X} \longrightarrow \mathcal{S}$, where $\mathcal{S} \subseteq \mathbb{R}^m$ represents the semantic space with m semantics. We can bridge the latent space $\mathcal{Z}$ and the semantic space $\mathcal{S}$ with $s = f_s(g(z))$, where s and z denote the semantic scores and the sampled latent code respectively.

Single Semantic. For any binary semantic such as male and female, a hyperplane exists in the latent space serving as the separation boundary. Semantic remains the same when the latent code walks within the same side of the hyperplane. This semantic will turn into the opposite when across the boundary.

Given a hyperplane with a unit normal vector $\mathrm{n} \in \mathcal{R}^d$,authors define the "distance" from a sample z to this hyperplane as

$$\mathrm{d(n,z)} = \mathrm{n}^T\mathrm{z} \tag{2}$$

Here, $\mathrm{d}(\cdot, \cdot)$ is not a strictly defined distance since it can be negative. When z lies near the boundary and is moved toward and across the hyperplane, both the "distance" and the semantic score vary accordingly. And it is just at the time when the "distance" changes its numerical sign that the semantic attribute reverses. We therefore expect these two to be linearly dependent with

$$f(\mathrm{g(z)}) = \lambda \mathrm{d(n,z)} \tag{3}$$

where f $(\cdot)$ is the scoring function for a particular semantic, and $\lambda > 0$ is a scalar to measure how fast the semantic varies along with the change of distance.

Multiple Semantics. When the case comes to m different semantics

$$s \equiv f_S(g(\mathrm{z})) = \Lambda N^T \mathrm{z} \tag{4}$$

where $s = [s_1, ..., s_m]^T$ denotes the semantic scores, $\Lambda = diag(\lambda_1, ..., \lambda_m)$ is a diagonal matrix containing the linear coefficients, and $N = [n_1, ..., n_m]$ indicates the separation boundaries. Aware of the distribution of random sample z, which is $\mathcal{N}(0, \mathrm{I}_d)$, the mean and covariance matrix of the semantic scores s can be computed by

$$\mu_s = \mathbb{E}(\Lambda N^T \mathrm{z}) = \Lambda N^T \mathbb{E}(\mathrm{z}) = 0 \tag{5}$$

$$\Sigma_s = \mathbb{E}(\Lambda N^T \mathrm{z}\mathrm{z}^T N \Lambda^T) = \Lambda N^T \mathbb{E}(\mathrm{z}\mathrm{z}^T) N \Lambda^T = \Lambda N^T N \Lambda \tag{6}$$

Where $\mathrm{s} \sim \mathcal{N}(0, \Sigma_\mathrm{s})$ is a multivariate normal distribution. Different entries of s are disentangled if and only if Σ_s is a diagonal matrix, which requires $\{\mathrm{n}_1, ..., \mathrm{n}_m\}$ to be orthogonal with each other. If this condition does not hold, some semantics will correlate with each other and $\mathrm{n}_i^T n_j$ can be used to measure the entanglement between the i-th and j-th semantics.

2.6.2 Manipulation in the Latent Space

Single Attribute Manipulation. According to Eq.(3), to manipulate the attribute of a synthesized image, we can easily edit the original latent code z with $\mathrm{z}_{\text{edit}} = \mathrm{z} + \alpha \mathrm{n}$. It will make the synthesis look more positive on such semantic with $\alpha > 0$, since the score becomes $f(g(\mathrm{z}_{\text{edit}})) = f(g(\mathrm{z})) + \lambda\alpha$ after editing. Similarly, $\alpha < 0$ will make the synthesis look more negative.

Conditional Manipulation. When there is more than one attribute, editing one may affect another since some semantics coupled with each other. To achieve more precise control, authors propose *conditional manipulation* by manually forcing $\mathrm{N}^T\mathrm{N}$ in Eq. (6) to be diagonal. In particular, we use projection to orthogonalize different vectors. As shown in Fig.8 given two hyperplanes with normal vectors n_1 and n_2, the projected direction $\mathrm{n}_1 - (\mathrm{n}_1^T \mathrm{n}_2)\mathrm{n}_2$, the movement in a new direction allow to change "attribute 1" without affecting "attribute 2". This operation is called conditional manipulation. If there is more than one attribute to be conditioned on, we subtract the projection from the primal direction onto the plane constructed by all conditioned directions.

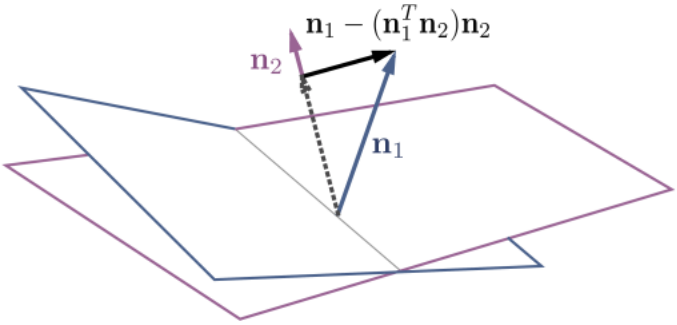

Figure 8: Illustration of the conditional manipulation in subspace.

Real Image Manipulation. Since the author's approach enables semantic editing from the latent space of a fixed GANs model, we need to first map a real image to a latent code before performing manipulation. For this purpose, existing methods have

proposed to directly optimize the latent code to minimize the reconstruction loss or learn an extra encoder to invert the target image back to latent space. Some models have already involved an encoder along with the training process of GANs.

2.7 Affine Transformation

A transformation contains two principal operations, which are matrix multiplication and vector addition. Matrix multiplication is a linear transformation, which could rotate an image or change the image scale. Another way, a vector addition could translate image.

An Affine Transformation could be represented by using a 2×3 matrix.

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2\times 2} \quad B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2\times 1} \tag{7}$$

$$M = \begin{bmatrix} A & B \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{10} \end{bmatrix}_{2\times 3} \tag{8}$$

For example, to transform a 2D vector $X = \begin{bmatrix} x \\ y \end{bmatrix}$, we can use the formula:

$$T = A \cdot \begin{bmatrix} x \\ y \end{bmatrix} + B = M \cdot [x, y, 1]^T = \begin{bmatrix} a_{00}x + a_{01}y + b_{00} \\ a_{10}x + a_{11}y + b_{10} \end{bmatrix} \tag{9}$$

By finding the Affine Transformation with three points in both images, then we can apply this found relation to all the pixels in an image.

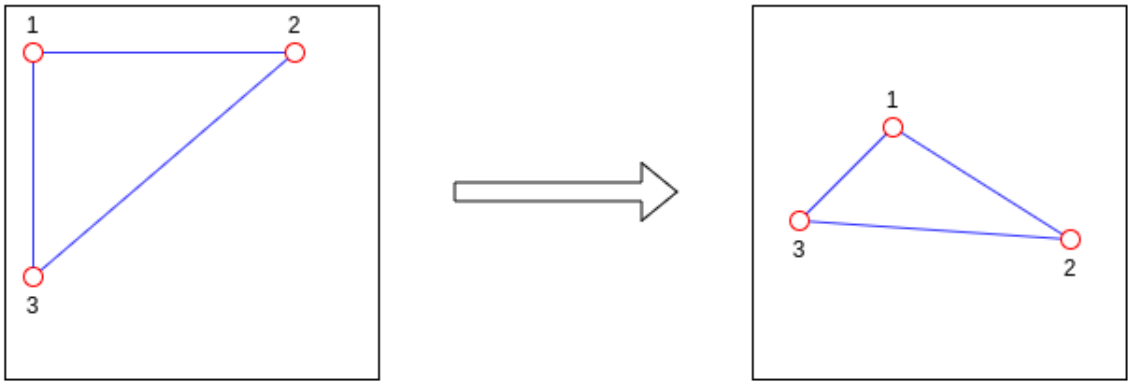

Figure 9: Affine Transformation with three points

2.8 Face Landmark Detection

Vahid et al. [5] present a new algorithm that detects specific points on the human face in milliseconds and achieves accuracy superior. Their model is trained on iBUG 300–W dataset which detects 68 points on the human face, which could be visualized in Fig. 10:

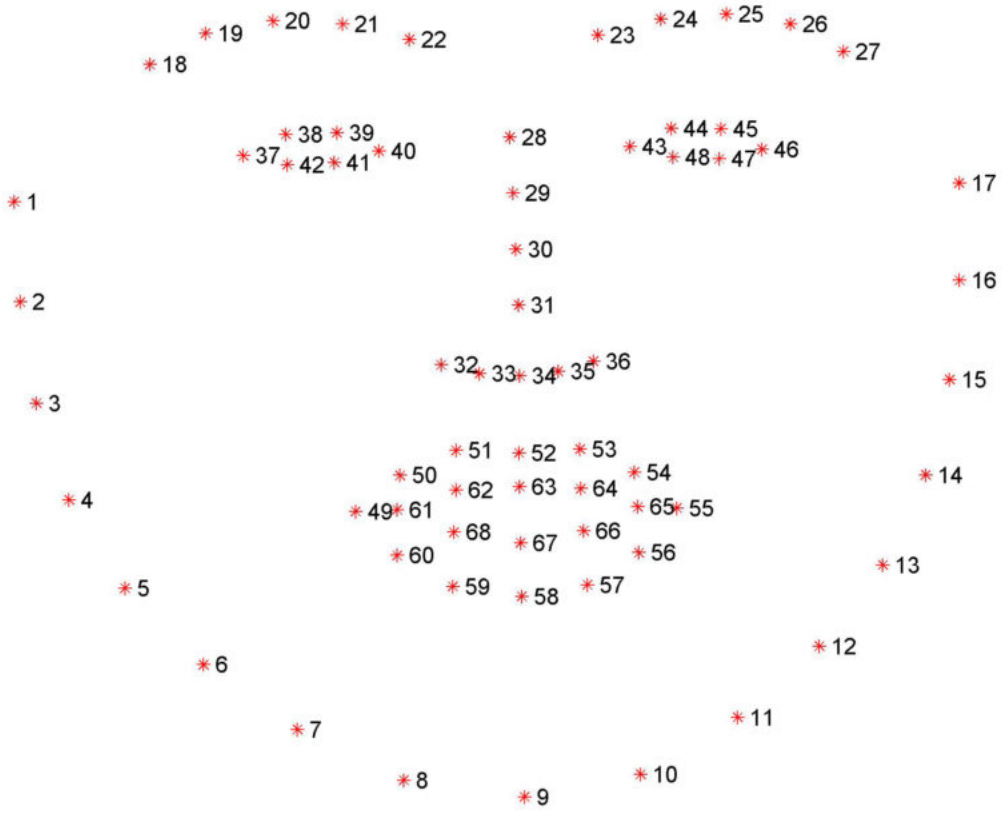

Figure 10: Visualizing the 68 facial landmark coordinates from the iBUG 300-W dataset

3 METHODOLOGY

3.1 Semantic editing latent code

As shown in section 2.6.2, InterFaceGan allows single and multiple attribute manipulation and multiple attribute manipulation done with conditional manipulation. In this work, we apply single attribute manipulation to manipulate hairstyle. The workflow of our first system is described in Fig.11. We take advantage of the single attribute manipulation technique proposed by Shen et al. to edit hair attributes.

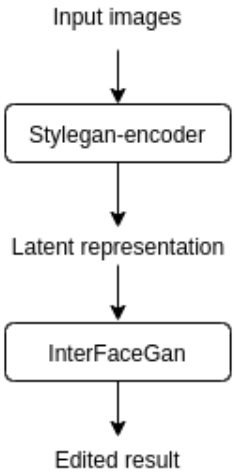

Figure 11: Work flow of first system

For semantic edit real images, we first need to map our images to latent space representation. One solution is feature vector optimization which produces a high-level semantic meaning of what is in the image. As we mention in 2.5, StyleGan Encoder allows mapping real images into the latent space of GAN. We first need to send the input image to pre-trained Resnet (Fig.12), fine-tuned, for initial latent code estimation. We choose Resnet because it is trained with an extensive image dataset that produces a high semantic meaning latent space vector of the input image. The detail about how Resnet fine-tuned shown in Fig.6. With the latent code estimation result from the Resnet encoder, we will optimize this latent code by the process in Fig.13. This is detail of VGG16 weighted combination loss block in Fig.7. Firstly, the latent code estimation is sent to the generator for an initial guess of the original input image. After that, we send both the original input image and it is an initial guess from the generator to the VGG16 weighted combination loss block. This block will send the image and initial

guess to a pre-trained image classifier(VGG) for feature extraction purposes. With two semantic vectors received from VGG, we then perform gradient descent to minimize the L2 loss of these feature vectors. We choose to use gradient descent in feature space instead of the original image because L2 optimization in pixel space can get stuck in bad local optima easier. With this process, we can find latent representation in the StyleGan latent space of any real images.

Figure 12: Resnet encoder

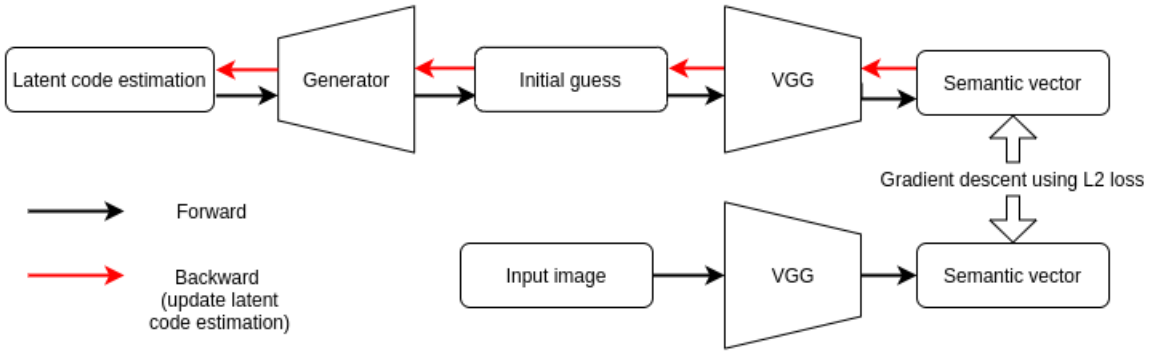

Figure 13: Latent code estimation process

In the next step, we use the single attribute manipulation technique proposed by InterFaceGan paper to edit hair attributes. According to this paper, for any binary attribute, there exists a hyperplane in latent space such that all samples from the same side are with the same attribute, so we first need to find this hyperplane. The process to find this hyperplane is shown in Fig.14. We will use a pre-trained classifier for hair attributes, which trained on a large dataset, to generate the input image's latent code and attribute score. The latent codes and attribute scores acquired from the classifier used to train a linear SVM to get the boundary (hyperplane) for the attribute. With the hyperplane for each attribute, we will use its normal vector as a manipulation direction. Along with this direction, the hairstyle will have continuous changes concerning the target attribute. For example, we found that our input images have latent vectors below the hyperplane in Fig.15, which means our face does not have bangs. We can add more bangs to our face image by moving our latent vector along the normal vector direction.

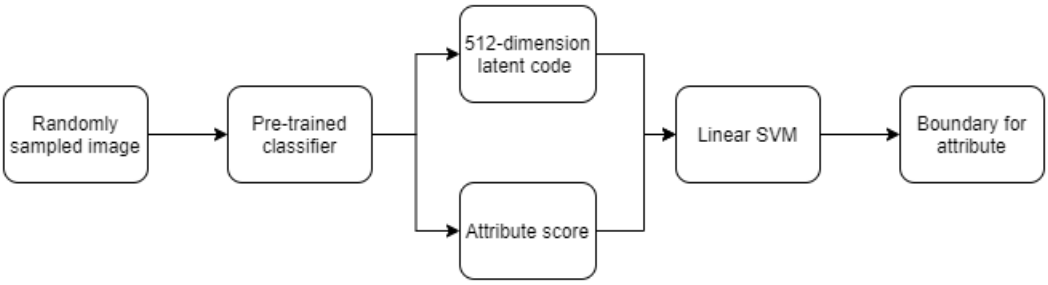

Figure 14: Training boundary process

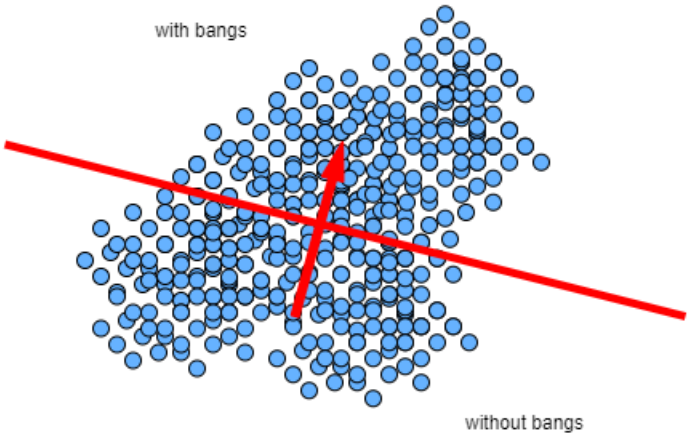

Figure 15: Hyperplane for bangs

3.2 Overwrite hair image over face image

Motivated by the StyleGan encoder and StyleGan generator, encoding and decoding can lose unnatural features, and retain good ones. This approach motivates various high-quality image editing applications, e.g. image reconstruction, image in-painting, image crossover, local style transfer, image editing using scribbles, and attribute level feature transfer. For example, Abdal et al. [19] a technique, which using StyleGan encoder and integrate various spatial masks into the optimization framework, so that this technique could change the properties of an area in the image. So we propose a method to overwrite the hairstyle photo on the target face, after that we use to encode and decode to make the picture look more realistic. Our model is described in general in Fig. 16.

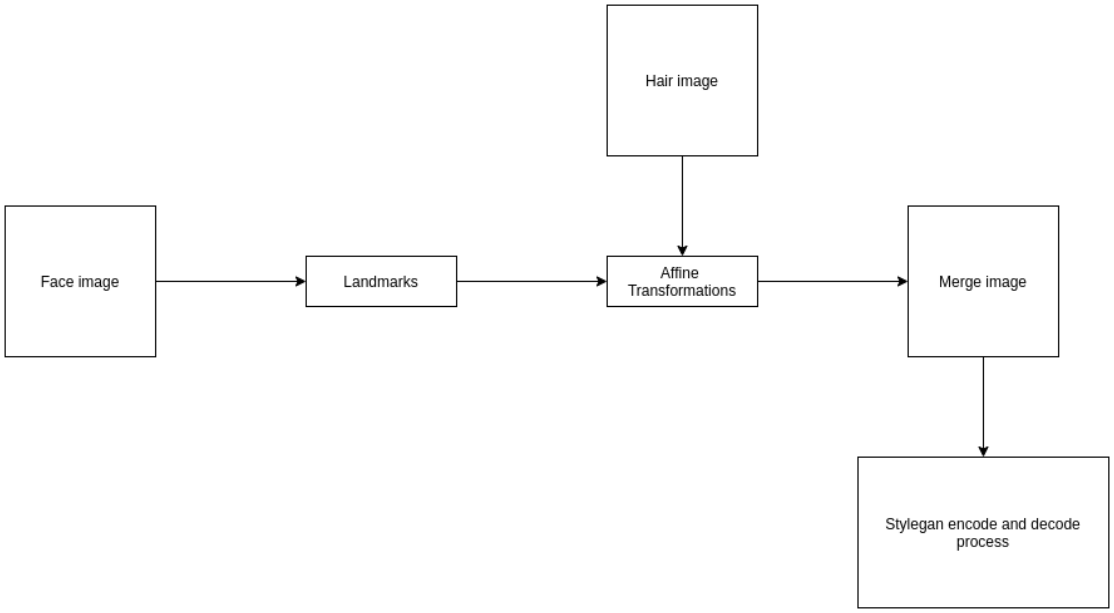

Figure 16: Merge hair process

For the first step, we use landmarks to mark 3 points on the face to mark the position of the hair on the target's face. We chose 2 position points above the ear, and a hairline point of the face. We also do the same with our pre-prepared hairstyle photos.

We use affine transformations to overwrite the hair image on the face image based on 3 pairs of highlighted points in the next step. We recommend hair segment and blurring methods for the target faces with complex hairstyles before overwriting the new hairstyle. The resulting photo has many unnatural features, hairline, skin color difference, inappropriate position, etc.

Finally, the photo, which has been overwritten with the new hairstyle, passed through the encode and decode StyleGan. At certain steps, the unnatural properties of the photo were blurred. Based on the results in Fig. 19, at the first stages, there is a lot of blurred properties, such as the skin of the target face. However, at high stages, undesirable traits such as hairline and skin discoloration appear. Through the tests, the stage from 15 to 20 gives the best results.

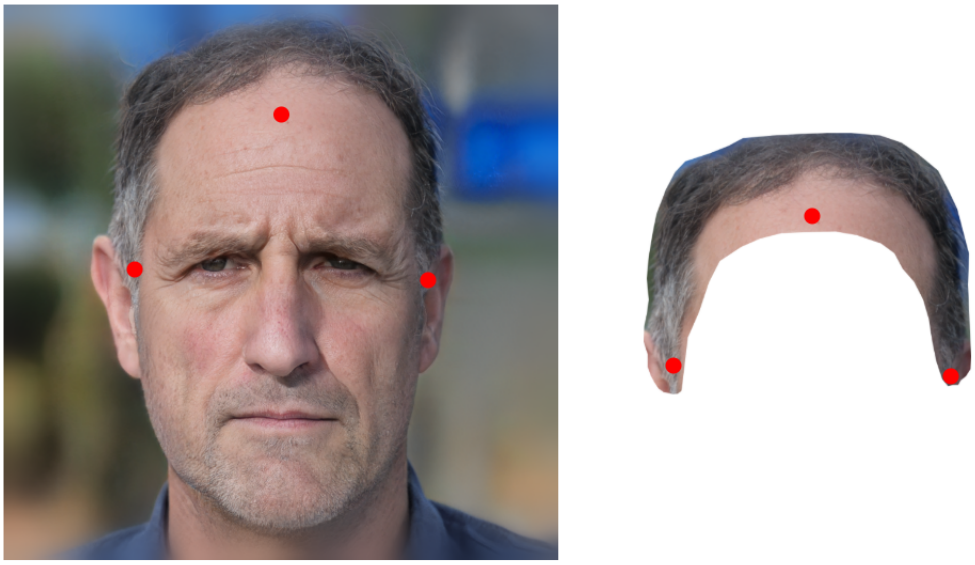

Figure 17: Face image and hair image with landmark points example

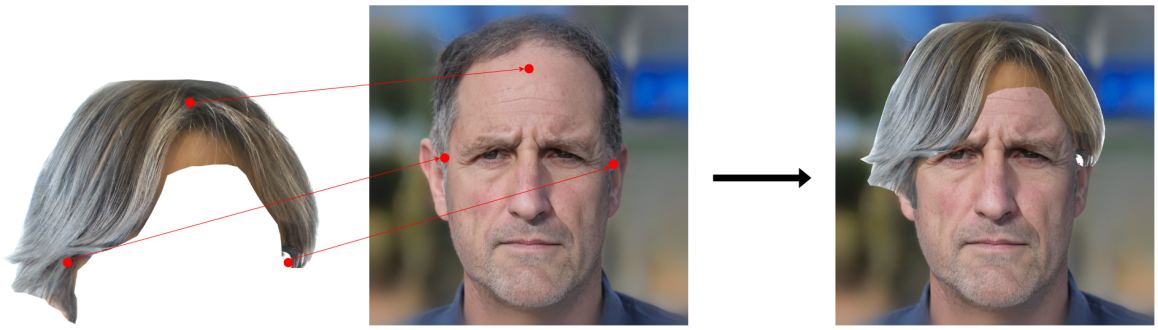

Figure 18: Use affine transformations to overwrite hairstyle

4 IMPLEMENTATION AND ANALYSYS

4.1 Dataset

4.1.1 Randomly sampled image dataset for boundary training

As shown in Fig.14, we use randomly sampled image as our input for boundary training process. This dataset contain 100000 images which are resized to 256x256 to match the input of pre-trained classifier.

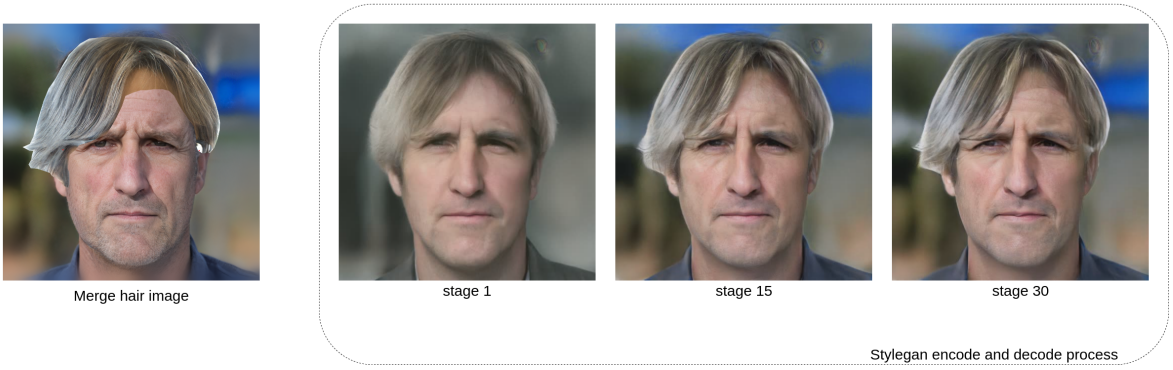

Figure 19: Use stylegan encode and decode process to improve quality of image

4.1.2 Generated dataset for overwrite hair image method

We use the Stylegan model to generate 100 images with the size 1024x1024. Dataset is selected with all gender and different ages. In the dataset, we highlighted and cut out some simple hairstyles for testing.

4.2 Experiment results

4.2.1 Semantic editing latent code

As we mention in 3.1, we can find the latent vector represent our input image in StyleGan latent space and use the normal vector of attribute's hyperplane as our di-

rection, along which we modify our latent vector representation. There is two-direction according to the normal vector, which is positive or negative.

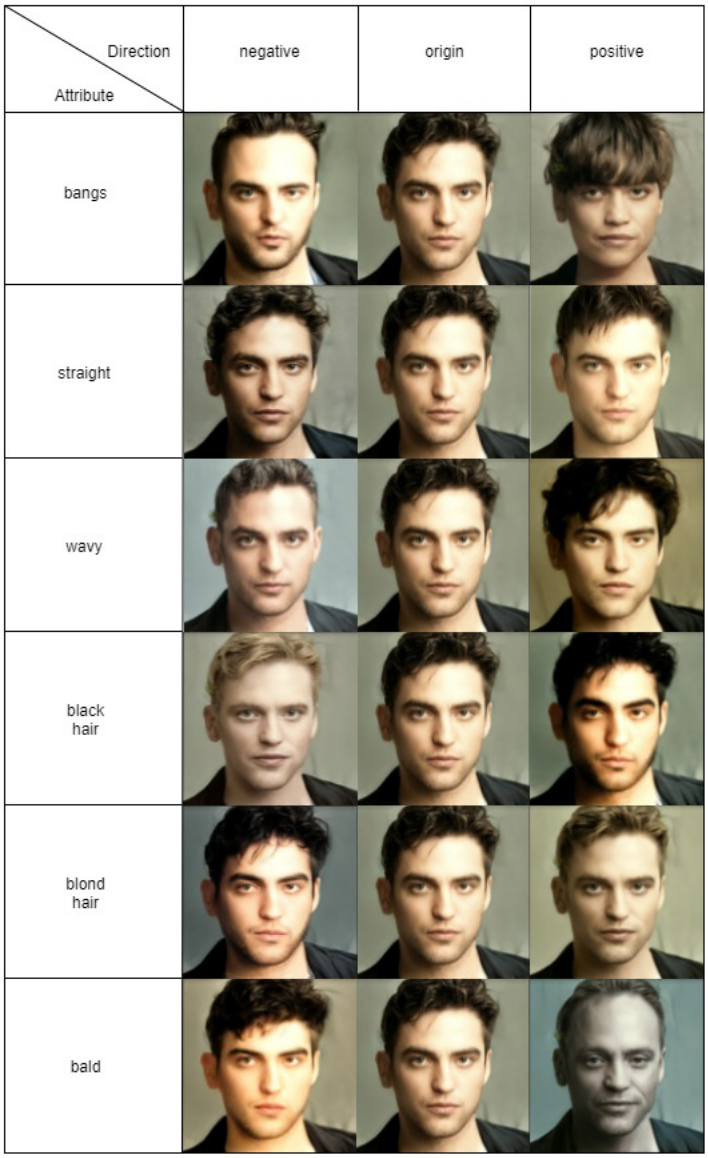

Figure 20: Experiment result

We can see that hairstyles changed concerning the attribute. However, some attribute correlated with other, for instance, Fig.21 show that the gender of people in the

image will be changed when we move the latent vector of that image too much regard to wavy hair attribute. This problem can be solved with conditional manipulation, which mentioned in A 2.6.2

Figure 21: Changing latent vector too much regard to the target attribute (wavy hair)

Another problem is InterFaceGan's failure when we use our face (Asian people). In Fig.4.2.1, we experiment with bangs target attribute. As we can see, InterFaceGan changes the face slightly like European people; that is the problem of the training dataset of InterFaceGan. The problem is that this dataset lacks the Asia people image for training. We can improve by adding more Asia people images to our training dataset.

4.2.2 Overwrite hair image over face image

We trained the resnet model, which is mentioned in 2.5, with 10000 images. After training, we got a good resnet, which could create a high semantic meaning latent space vector of an input image. In optimize step, we combine per-pixel loss and VGG16 weighted combination loss for loss function. About 100 face images and 20 hairstyle images are chosen for testing. Finally, we get a positive result, Fig. 23 presents examples of images synthesized by mixing face and hair image in the test dataset.

The results show that our model has a good score. The transformed hairstyle retains all of its unique properties, and the image is naturally born. In particular, when the affine transformations step is not good, and the image merge has many unnatural points. Still, the model optimization works very well, making the resulting image very natural and accurate.

In the actual data, some properties often go together, and it is difficult to separate. For example, babies often have short hair, bald people or gray hair are usually elderly, women often have long hair, etc. However, our method is unaffected by this issue

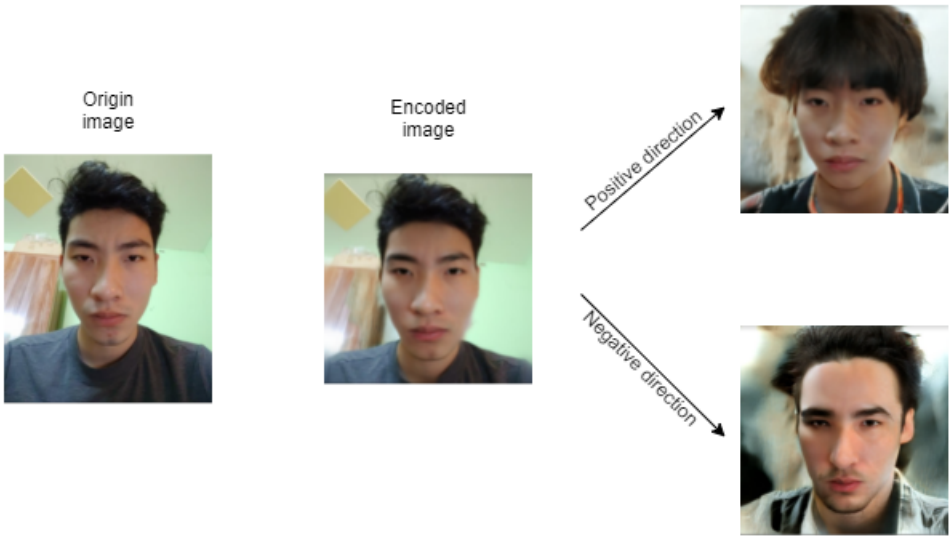

Figure 22: Problem of InterFaceGan

because we use the local editing technique to not affect properties in other areas. An example is shown in Fig. 25.

However, this method also has many limitations. Firstly, this method needs an optimization step, so it takes time to generate result images. We also need an accurate collage image, so that the resulting photo doesn't have too many unnatural elements. Because of this reason, The direction of the hair image should be the same as the direction of the face image. If the opposite is true, the position of the hair in the photo will be skewed, resulting in very bad results. An example is shown in Fig. 26.

The brightness and background of the two images are also a factor in the quality of the results. The difference in these features could change the hair color of the resulting photograph. Besides, it also creates a border in the hair area of some photos. But this problem can be overcome when increasing the quality of the hair photo.

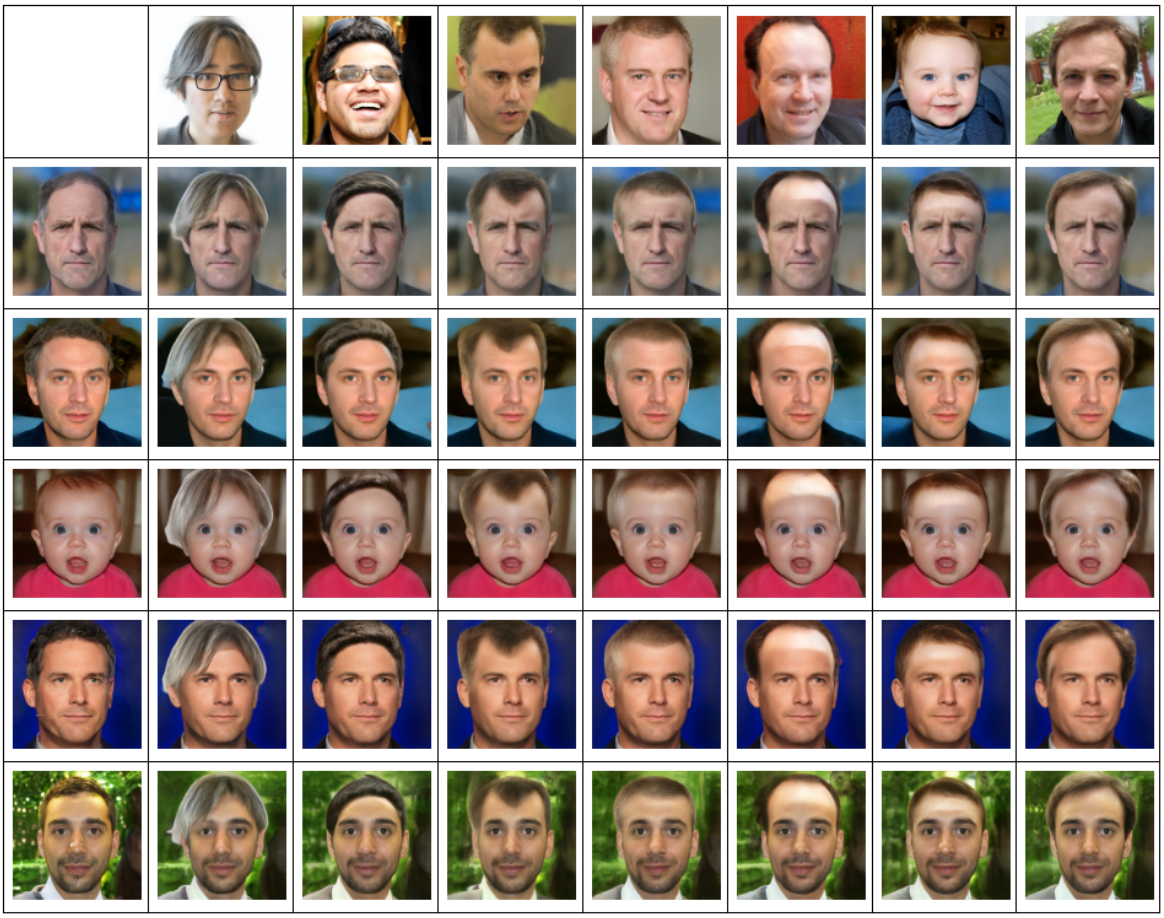

Figure 23: Some result images in Overwrite hairstyle method

5 CONCLUSION AND PERSPECTIVES

In this work, we propose two methods to change the hairstyle of a given image, which take advantage of two successful GAN architecture: StyleGan and InterFaceGan.

In the first method, we use an assumption in InterFaceGan about hyperplane separate any binary attribute in GAN latent space. In the second method, we use affine transformation to merge a hair image to a given image and then optimize the image using StyleGan. We also point out the problem of this method when applying real-world images and how we can improve the problem.

Because of problems with our method, we desire to improve our result in future works. We will apply conditional manipulation for dealing with an attribute that cor-

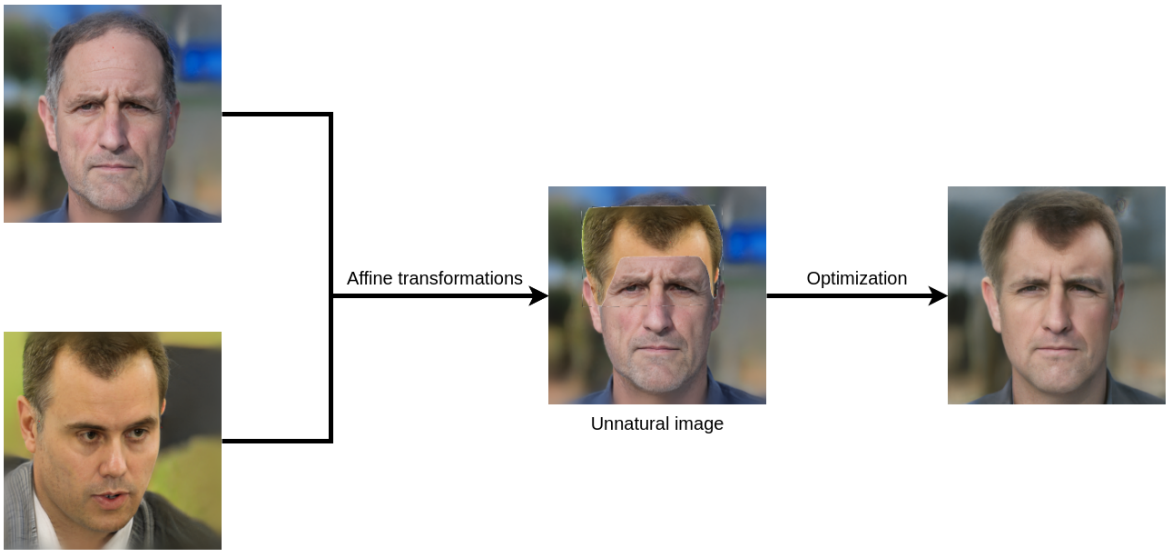

Figure 24: Optimize model could remove unnatural features

Figure 25: Model uses local editing so that properties in other areas were not affected (Baby age is not affected)

relates with others. InterFaceGan also can be improved by training with a training set, which has more Asia people images. In the second method, we will improve the resnet model and Stylegan generator with a dataset, which has more hairstyle images.

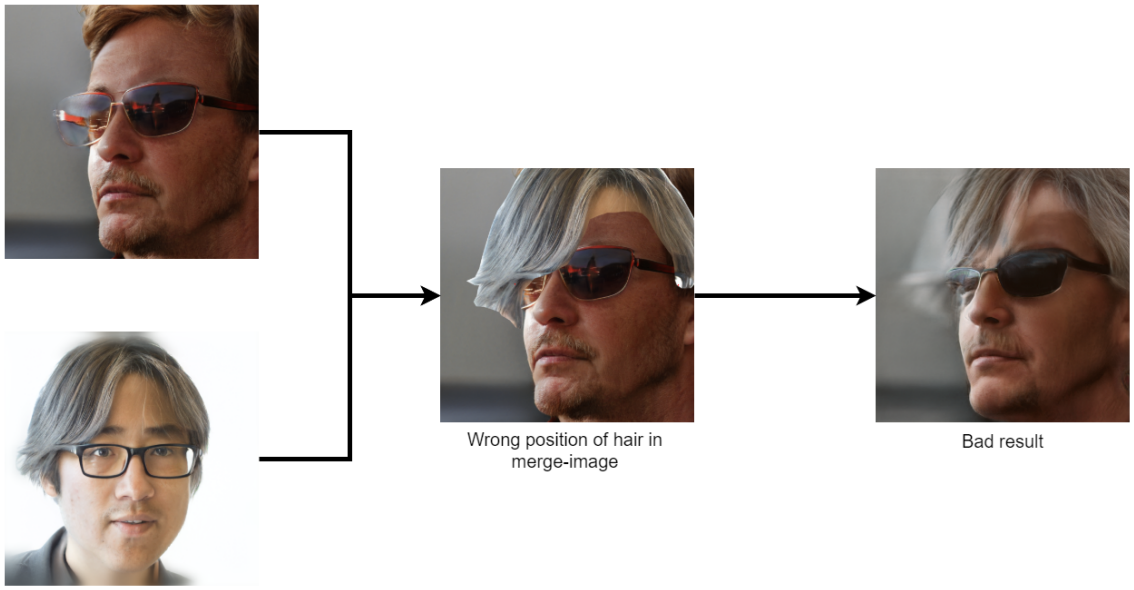

Figure 26: Different directions images could generate bad result

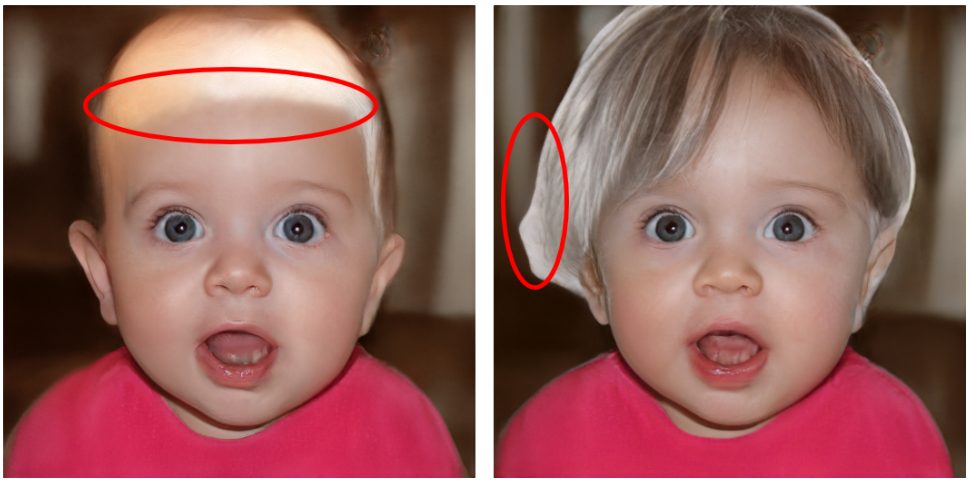

Figure 27: Effect of brightness to the result

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In

Advances in neural information processing systems, pages 2672–2680, 2014.

[2] Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In CVPR, 2019

[3] Shen Y, Gu J, Tang X, Zhou B (2019) Interpreting the latent space of gans for semantic face editing. arXiv preprint arXiv:190710786

[4] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. arXiv preprint arXiv:2008.00951, 2020.

[5] Kazemi, V., Sullivan, J.: One millisecond face alignment with an ensemble of regression trees. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014),

[6] C. Dong, C. C. Loy, K. He, and X. Tang, "Image Super-Resolution Using Deep Convolutional Networks", IEEE Transactions on Pattern Analysis and Machine Intelligence 38, 295 (2016).,

[7] Cheng, Z., Yang, Q., Sheng, B.: Deep colorization. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 415–423,

[8] Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Computer vision and pattern recognition. IEEE, pp 3431–3440,

[9] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Proceedings of the IEEE International Conference on Computer Vision, pages 2650–2658, 2015.,

[10] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. TIP, 2004.,

[11] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In Signals, Systems and Computers. IEEE, 2004.,

[12] L. Zhang, L. Zhang, X. Mou, and D. Zhang. Fsim: A feature similarity index for image quality assessment. TIP, 2011.,

[13] R. Mantiuk, K. J. Kim, A. G. Rempel, and W. Heidrich. Hdrvdp-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions. In ACM Transactions on Graphics (TOG), 2011.,

[14] Mahendran, A., Vedaldi, A.: Understanding deep image representations by inverting them. In: Proc. CVPR (2015).,

[15] Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034, 2013.,

[16] A. M. Nguyen, J. Yosinski, and J. Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. CoRR, abs/1602.03616, 2016.,

[17] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. CoRR, abs/1508.06576, 2015.,

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015.

[19] Abdal, R., Qin, Y., Wonka, P.: Image2stylegan++: How to edit the embedded images? arXiv preprint arXiv:1911.11544 (2019),