# ACL-GAN: multi-domain image-to-image translation GAN using new losses to reduce the time for hyperparameter optimization and training

Jeonglk Cho

Konkuk University

**Abstract**

*StarGAN, which has impressive performance in multi-domain image-to-image translation, used three losses to train GAN: adversarial loss, classification loss, reconstruction loss. In this study, by proposing an attribute GAN loss that can replace conditional GAN losses: adversarial loss and classification loss, reduce hyperparameter (classification loss weight) and fasten training speed. Proposed attribute loss is the sum of the losses of each GAN when creating a GAN for each attribute, and since each GAN shares hidden layers, it does not increase the amount of computation much. Also, propose simplified content loss, which can replace reconstruction loss of StarGAN. Reconstruction loss of StarGAN uses the generator twice, while simplified content loss uses only once, reduce the amount of computation. Also, propose an architecture that prevents background distortion through image framing, improves training speed through a bi-directional progressive growing generator, and mixed batch training to apply batch normalization in discriminator.*

## 1. Introduction

StarGAN [1] is a multi-domain image-to-image translation GAN that uses three losses: adversarial loss to the generated image looks real, classification loss to the generated image has target attribute, reconstruct loss to the generated image changes only target attribute, not other hidden attributes. Therefore, to train StarGAN, it is necessary to find out the proper ratio of these three losses through experiments.

In this study, by proposing attribute GAN, which replaces conditional GAN [2], reduce hyperparameter of conditional GAN, and improve training speed. Loss of attribute GAN is the sum of losses of each GANs that each GAN trains only one attribute. Because each GAN shares all hidden layers, it is possible to consider the GANs as one GAN. Unlike conditional GAN using two losses (adversarial loss, classification loss), attribute GAN uses only one loss (attribute loss), which means it does not need to find the ratio of adversarial loss and classification loss. Also, attribute loss always produces meaningful gradients, whereas generator classification loss using cross-entropy produces meaningless gradients at the beginning of training.

In conditional GAN (or attribute GAN), applying batch normalization to discriminator distorts condition distribution of input batch. If discriminator applied batch normalization, the distribution of generated data batch tries to follow the distribution of real data batch. Likewise, condition distribution of generated data batch tries to follow the condition distribution of real data batch. This means that some data in the batch may ignore the input condition to follow the condition distribution of the real data batch. I suggest mixed batch training, which is composing batch always with the same ratio of real data and generated data, to keep condition distribution of batch same to apply batch normalization in the discriminator.

The StarGAN uses reconstruction loss (cycle consistency loss for StarGAN) to ensure the generated image change only target attribute, not other hidden attributes. If generator changes hidden attribute of input data, for example, in face expression change GAN, generated face will be face of different person to input person. However, not want to change any attribute other than the target attribute, do not have to use cycle consistency loss. Proposing simplified content loss is a difference between real data and generated data that is generated by a generator with the real data and real attribute of the real data. Simplified content loss guarantees immutability of hidden attributes, uses the generator only once, while reconstruction loss uses the generator twice, thus reduce memory usage and computation amount.

In StarGAN, since adversarial loss and classification loss (or attribute loss) focus only on the face, not background, cause background distortion. Although high reconstruction loss weight (or simplified content loss weight) can prevent background distortion, there can be a problem that the entire image hardly changes. Instead of raising reconstruction loss weight (or simplified content loss weight), image framing can prevent background distortion. As image completion GAN [3] implies, pasting frame of the real image to the generated image while training makes generated image match frame of the real image. And the easiest way to generate a background that matches the frame of the real image for the generator is not distorting the background.

In architecture, by extending progressive-growing generator [4], used a bi-directional progressive-growing generator to improve training speed.

## 2. ACL-GAN

### 2.1 Attribute GAN

The loss of conditional GAN is as follows.

$$L_D = L_{adv}^d + \lambda_{cls} L_{cls}^r$$

$$L_G = L_{adv}^g + \lambda_{cls} L_{cls}^g$$

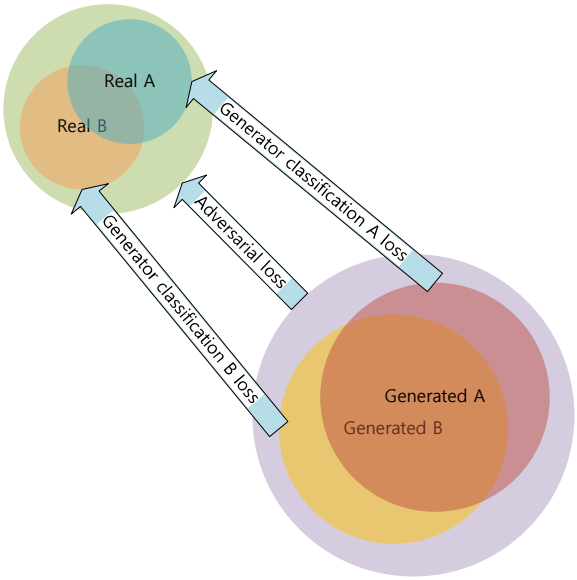
$$L_{cls}^r = E_{x,att\sim P_r(x,att)}\left[-\log\left(D_{cls}(att|x)\right)\right]$$

$$L_{cls}^g = E_{x',att'\sim P_g(x',att')}\left[-\log\left(D_{cls}(att'|x')\right)\right]$$

In $x, att \sim P_r(x, att)$, $x$ is real data, and $att$ is the binary vector that expresses the attributes of real data. In $x', att' \sim P_g(x', att')$, $x'$ means generated data, and $att'$ is the target binary

vector to make $x'$.

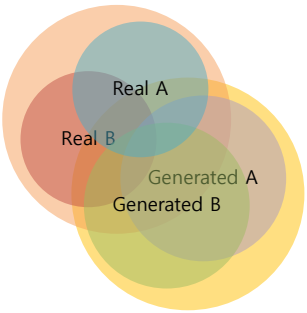
In the conditional GAN, adversarial loss trains model well because there are well known adversarial losses such as LSGAN [5] or WGAN-GP [6] that can produce meaningful gradients even if real data distribution and generated data distribution are far from each other. However, classification loss of conditional GAN, which is using cross-entropy, is hard to produce meaningful gradients if real data distribution and generated data distribution are far from each other because cross-entropy measures only KL-divergence.




Real X: Real data distribution with attribute X
Generated X: Generated data distribution to have attribute X

Fig1. Data distribution at the beginning of training using conditional GAN

In the early stage of learning, the generator classification loss $L_{cls}^{g}$ does not produce meaningful gradients because the distance between Real A and Generated A, Real B and Generated B are too far from each other. Only adversarial loss produces meaningful gradients.



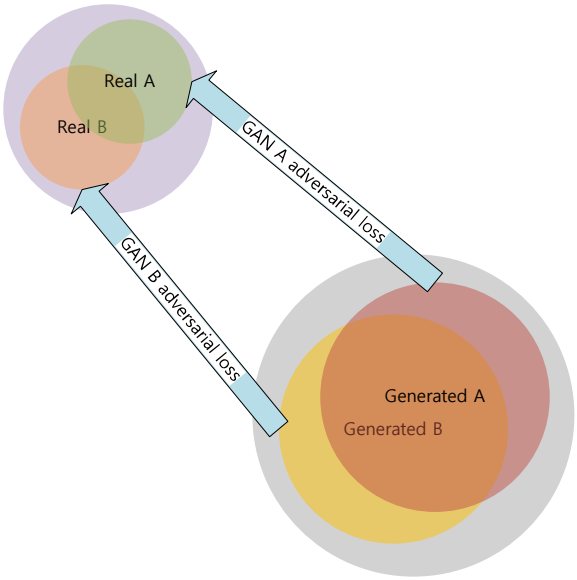

Real X: Real data distribution with attribute X
Generated X: Generated data distribution to have attribute X

Fig2. After some training using conditional GAN

As learning progresses to some degree with adversarial loss, when the real data distribution and the generated data distribution become somewhat similar, the generator classification loss $L_{cls}^{g}$ begins to produce a meaningful gradient because real A and generated A, real B and generated B become somewhat similar.

Also, conditional GAN has important hyperparameters: adversarial loss weight and classification loss weight. If adversarial loss weight is too bigger than the classification loss weight, the data would not have the target condition. If classification loss weight is too bigger than adversarial loss weight, the data does not look real.

To solve these problems of conditional GAN, I propose attribute loss, which is similar to having multiple GANs that each GAN trains each attribute.




Real X: Real data distribution with attribute X
Generated X: Generated data distribution to have attribute X
GAN X: GAN which trains about only attribute X

Fig3. Attribute loss

Attribute loss is the sum of each GAN's loss. Each GAN trains only one attribute.

$$L_{att}^{D} = \sum_{c}^{att} L_{D_c}$$

$$L_{att}^{G} = \sum_{c}^{att} L_{G_c}$$

$$L_{D_c} = E_{x,c\sim P_r(x,c)}[f_r^D(D_c,x)] + E_{x'\sim P_{G_c}(x',1)}[f_g^D(D_c,x')]$$

$$L_{G_c} = E_{x'\sim P_{G_c}(x',1)}[f^G(D_c,x')]$$

$c$ means one specific attribute among several attributes. GAN $c$ is the GAN that train about only attribute $c$.

$G_c$ and $D_c$ are generator and discriminator of GAN $c$. $G_c$ receives a binary activation value with a latent vector. If $G_c$ receives 1 as an activation value, $G_c$ tries to trick $D_c$, and $D_c$ tries to discriminate generated data as fake. If $G_c$ receives 0 as activation value, $G_c$ and $D_c$ don't care about it (do not train). $D_c$ only tires of discriminating real data, which has attribute $c$ as real, and don't care about other real data.

In $x,c\sim P_r(x,c)$, $x$ is real data which has attribute $c$. In $x'\sim P_{G_c}(x',1)$, $x'$ is generated data by $G_c$ when it receives latent vector and 1 as activation value.

$f_r^D$ is an adversarial loss of discriminator about real data. $f_g^D$ is an adversarial loss of discriminator about generated data. $f^G$ is an adversarial loss of generator.

The following formula is an example of LSGAN adversarial loss.

$$L_{D_c} = E_{x,c\sim P_r(x,c)}[(D_c(x)-1)^2] + E_{x'\sim P_{G_c}(x',1)}[D_c(x')^2]$$

$$L_{G_c} = E_{x'\sim P_{G_c}(x',1)}[(D_c(x')-1)^2]$$

Since each GAN shares all hidden layers, attribute loss can be changed as the following formula.

$$L_{att}^{D} = E_{x,att\sim P_r(x,att)}[f_r^D(D,x)\cdot att] + E_{x',att'\sim P_g(x',att')}[f_g^D(D,x')\cdot att']$$

$$L_{att}^{G} = E_{x',att'\sim P_g(x',att')}[f^G(D,x')\cdot att']$$

In $x,att\sim P_r(x,att)$, $x$ is real data, and $att$ is the binary vector that expresses the attributes of real data. In $x',att'\sim P_g(x',att')$, $x'$ means generated data, and $att'$ is the target binary vector to make $x'$. '$\cdot$' is an inner product.

The following formula is an example of attribute loss with LSGAN adversarial loss.

$$L_{att}^{D} = E_{x,att\sim P_r(x,att)}[(D(x)-1)^2\cdot att] + E_{x',att'\sim P_g(x',att')}\left[\left(D(x')\right)^2\cdot att'\right]$$

$$L_{att}^{G} = E_{x',att' \sim P_g(x',att')}\left[\left(D(x') - 1\right)^2 \cdot att'\right]$$

Also, in conditional GAN, when the output of classifier A is 0, that means input data does not have attribute A. However, in attribute GAN, GAN A does not care about attribute not-A. Therefore, to train attribute not-A, new GAN which trains attribute not-A should be added.

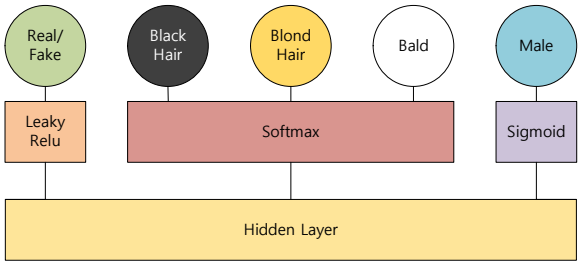



Fig4. conditional GAN discriminator output example

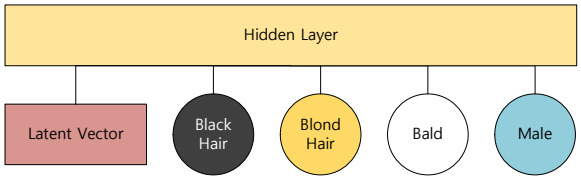



Fig5. conditional GAN generator input example

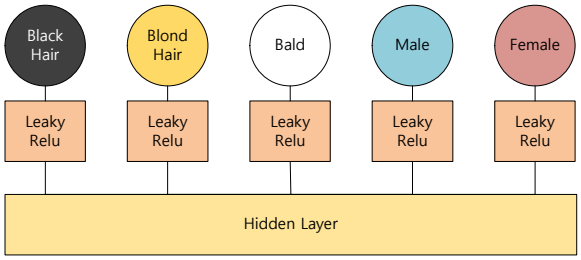



Fig6. attribute GAN discriminator output example

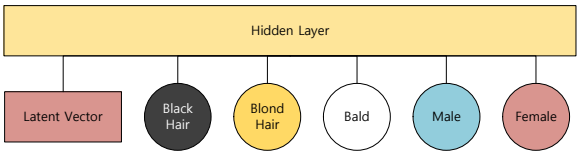



Fig7. attribute GAN generator input example

(Assume P(Black hair) + P(Blond hair) + P(Bald) = 1, P(Male) + P(Female) = 1)

Using attribute loss with adversarial loss of LSGAN or WGAN-GP or other GAN can generate meaningful gradients at the beginning of the training when real data distribution and generated data distribution are far from each other. Also, attribute loss can replace adversarial loss and classification loss, which can reduce one important hyperparameter of conditional GAN. Attribute GAN loss has only one hyperparameter: attribute loss weight, while conditional GAN loss has two hyperparameters: adversarial loss weight, classification loss weight.

## 2.2 Mixed batch training

In conditional GAN (or attribute GAN), applying batch normalization to discriminator distorts condition distribution of input batch. If discriminator applied batch normalization, the distribution of generated data batch tries to follow the distribution of real data batch. Likewise, condition distribution of generated data batch tries to follow the condition distribution of real data batch. This means that some data in the batch may ignore the input condition to follow the condition distribution of

the real data batch. I suggest mixed batch training, which is composing batch always with the same ratio of real data and generated data, to keep condition distribution of batch same to apply batch normalization to the discriminator.

### 2.3 Simplified Content Loss

The original StarGAN paper uses reconstruction loss (cycle consistency loss for StarGAN) to ensure generator change only target attributes, not hidden attributes.

$$L_{rec} = E_{x,att\sim P_r(x,att)}[||G(G(x,\ att'),att) - x||_1]$$

In $x, att\sim P_r(x, att)$, $att$ is the real attribute of real image $x$, and $att'$ is the target attribute to change image.

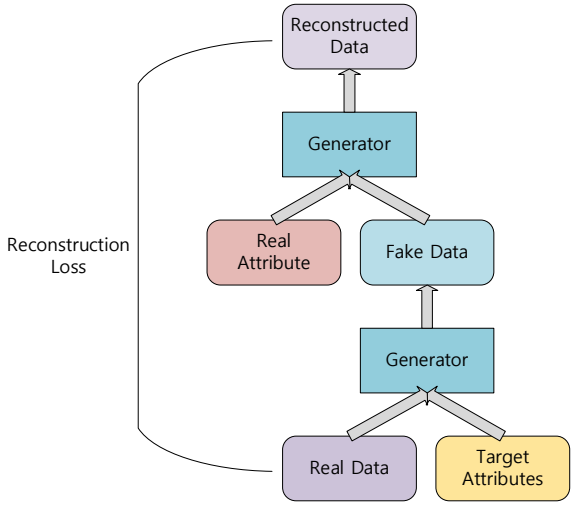



Fig8. Original reconstruction loss of StarGAN

However, not want to change any attribute other than the target attribute, do not have to use cycle consistency loss.

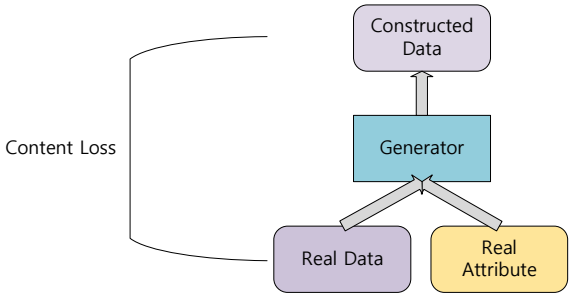



Fig9. Simplified content loss

I suggest simplified content loss that the original image goes through the generator only once.

$$L_{cnt} = E_{x,att\sim P_r(x,att)}[||G(x,\ att) - x||_1]$$

Since the original image goes through the generator only once, the calculation and memory usage can be reduced. Also, the immutability of hidden attributes is guaranteed.

### 2.4 ACL-GAN Loss

Used attribute loss with LSGAN and simplified content loss to train multi-domain image-to-image translation GAN.

$$L_D = L_{att}^D$$

$$L_G = L_{att}^G + \gamma_{cnt}L_{cnt}$$

$$L_{att}^D = E_{x,att\sim P_r(x,att)}\left[(D(x)-1)^2 \cdot att\right]$$

$$+E_{x',att'\sim P_g(x',att')}\left[\left(D(x')\right)^2 \cdot att'\right]$$

$$L_{att}^G = E_{x',att'\sim P_g\left(x',att'\right)}\left[\left(D\left(x'\right)-1\right)^2 \cdot att'\right]$$

$$L_{cnt} = E_{x,att\sim P_r(x,att)}[||G(x,\ att) - x||_1]$$

### 2.5 Image Framing

In StarGAN, since adversarial loss and classification loss (or attribute loss) focus only

on the face, not background, cause background distortion. For example, when the generator changes the hair color of an image from blond to black, it is easier for the generator to change the entire image, including the background, to black rather than just finding the hair and changing it black. Although high reconstruction loss weight (or simplified content loss weight) can prevent background distortion, there can be a problem that the entire image hardly changes. Instead of raising reconstruction loss weight (or simplified content loss weight), image framing can prevent background distortion. As image completion GAN implies, pasting frame of the real image to the generated image while training makes generated image match frame of the real image. And the easiest way to generate a background that matches the frame of the real image for the generator is not distorting the background.

Figure 10. Single attribute transfer on CelebA (Input, Black hair, Blond hair, Brown hair, Gender, Mouth, Pale skin, Rose cheek, Aged).


Figure 4. Facial attribute transfer results on the CelebA dataset. The first column shows the input image, next four columns show the single attribute transfer results, and rightmost columns show the multi-attribute transfer results. H: Hair color, G: Gender, A: Aged.

Fig10. Background distortion of StarGAN. Capture from the paper of StarGAN

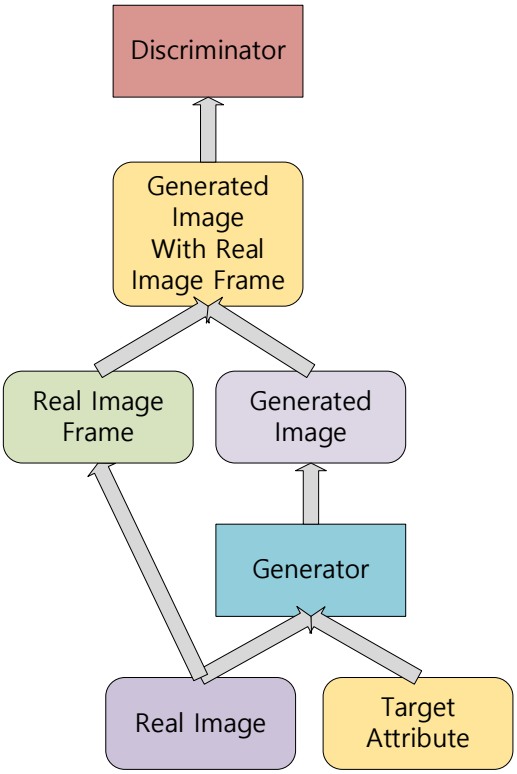



Fig11. Image Framing

## 2.6 Architecture

### 2.6.1 Generator

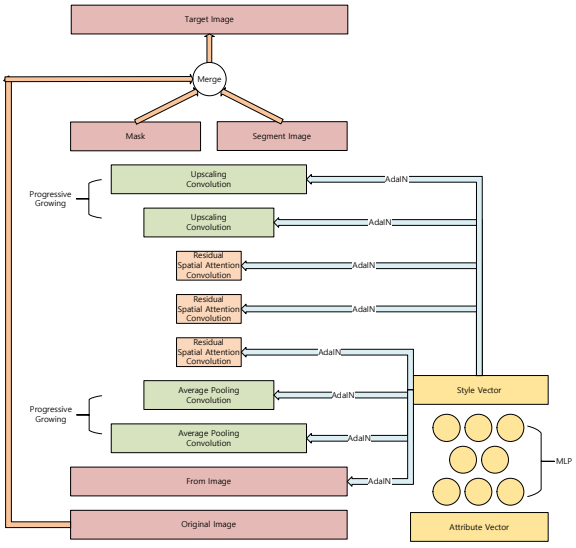



Fig12. Generator Architecture

In generator architecture, the AdaIN module and embedder of Style-based generator [7], mask of CAGAN [8], and convolution block attention module of CBAM [9] were used. There is no batch normalization in the generator.

To improve the training speed, I suggest a bi-directional progressive growing generator, which grows in both input and output directions, not just in one direction.

### 2.6.2 Discriminator

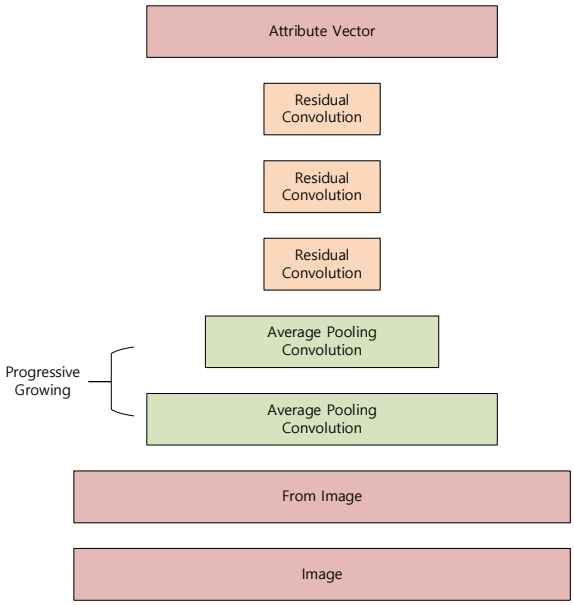



Fig13. Discriminator Architecture

Discriminator has attribute outputs that each output discriminates whether real image with each attribute or generated image with each attribute. Batch normalization was applied between each layer.

## 3. Material and methods

Used celeb_a [10] train dataset (162,770 pictures with attribute label) for the train. Used celeb_a test dataset (19,962 pictures with attribute label) for the test. Trained 5 attributes: black hair, blond hair, brown hair, smiling, mouth slightly open. Attribute weight, reconstruction weight, learning rate, and progressive-growing timing was manually tuned. Used two rtx2080ti with Tensorflow 2.0. Trained under 200 epochs.

## 4. Results and Conclusions

All first pictures are original pictures, second pictures are generated pictures, third pictures are mask images, and fourth pictures are generated segment images.

All generated image uses a four-pixel frame of the original image. Since the generator used image framing while training, the generator does not generate meaningful edges of generated images.

4.1 Good Cases

Target attributes: brown hair, not mouth slightly open, smiling

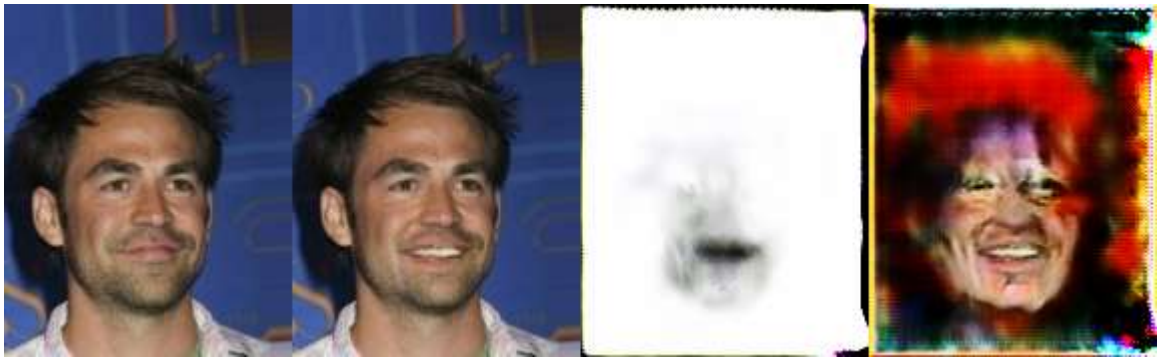
Target attributes: brown hair, not mouth slightly open, smiling

Target attributes: brown hair, not mouth slightly open, smiling

4.2 Bad Cases

Target attributes: blond hair, mouth slightly open, not smiling

Target attributes: black hair, not mouth slightly open, smiling



Target attributes: blond hair, mouth slightly open, smiling

In these cases, one or more target attributes were ignored

Target attributes: black hair, mouth slightly open, smiling

Target attributes: blond hair, mouth slightly open, smiling

Target attributes: black hair, not mouth slightly open, not smilling

In these cases, images do not look natural.

Target attributes: blond hair, mouth slightly open, not smiling

Target attributes: blond hair, not mouth slightly open, smiling

Target attributes: blond hair, mouth slightly open, not smiling

In these cases, because of image framing, the edge of images and surrounding pixels were not changed.

Target attributes: blond hair, not mouth slightly open, not smiling

Target attributes: brown hair, not mouth slightly open, not smiling

Target attributes: brown hair, mouth slightly open, smiling

Target attributes: blond hair, mouth slightly open, smiling

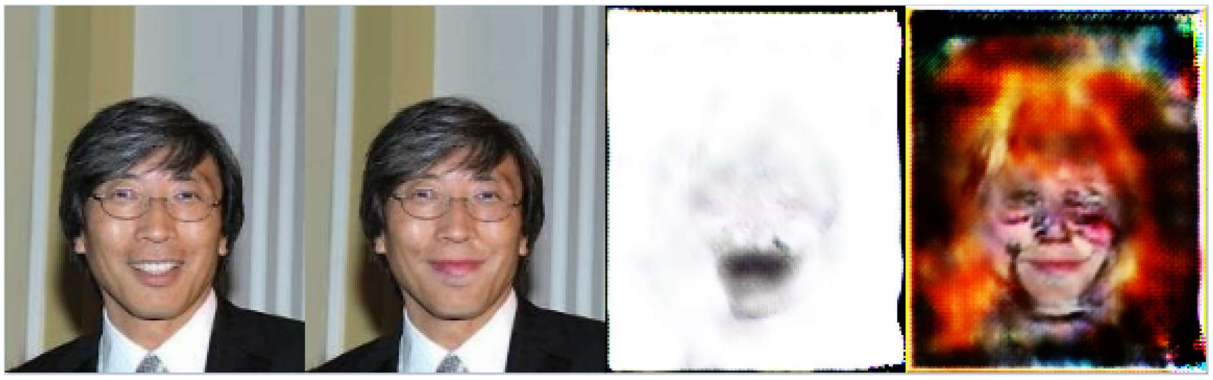
In these cases, the image is completely corrupted in the same way.

In conclusion, it was possible to change the attributes of some images to target attributes.

**5. appendices**

5.1 Good results from more training

Target attributes: brown hair, not mouth slightly open, smiling



Target attributes: brown hair, not mouth slightly open, smiling

Target attributes: brown hair, mouth slightly open, not smiling

Target attributes: brown hair, mouth slightly open, smiling

Target attributes: blond hair, mouth slightly open, not smiling

Target attributes: black hair, mouth slightly open, smiling

5.2 Funding

This work was supported by "University Innovation Grant" from the Ministry of Education and National Research Foundation of Korea

**6. References**

[1] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, Jaegul Choo

StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation

https://arxiv.org/abs/1711.09020

[2] Mehdi Mirza, Simon Osindero

Conditional Generative Adversarial Nets

https://arxiv.org/abs/1411.1784

[3] Satoshi Iizuka, Edgar Simo-Serra, Hiroshi Ishikawa

Globally and locally consistent image completion

https://dl.acm.org/citation.cfm?id=3073659

[4] Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen

Progressive Growing of GANs for Improved Quality, Stability, and Variation

https://arxiv.org/abs/1710.10196

[5] Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, Stephen Paul Smolley

Least Squares Generative Adversarial Networks

https://arxiv.org/abs/1611.04076

[6] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville

Improved Training of Wasserstein GANs

https://arxiv.org/abs/1704.00028

[7] Tero Karras, Samuli Laine, Timo Aila

A Style-Based Generator Architecture for Generative Adversarial Networks

https://arxiv.org/abs/1812.04948

[8] Nikolay Jetchev, Urs Bergmann

The Conditional Analogy GAN: Swapping Fashion Articles on People Images

http://openaccess.thecvf.com/content_ICCV_2017_workshops/papers/w32/Jetchev_The_Conditional_Analogy_ICCV_2017_paper.pdf

[9] Sanghyun Woo, Jongchan Park, Joon-Young Lee, In So Kweon

CBAM: Convolutional Block Attention Module

https://arxiv.org/abs/1807.06521

[10] Ziwei Liu Ping Luo Xiaogang Wang Xiaoou Tang

Large-scale CelebFaces Attributes (CelebA) Dataset

http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html