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

JeongIk Cho

Konkuk University

Abstract

StarGAN, which has impressive performance in image-to-image translation, is based on the determination of three important hyperparameters: adversarial weight, classification weight, and reconstruction weight, which have a significant impact on the performance of the model. In this study, by proposing an attribute loss that can replace conditional GAN losses: adversarial loss and classification loss, the time required for the optimization of attribute weight replaced the time required for the optimization of adversarial weight and classification weight, which can drastically reduce the time required for hyperparameter optimization. Proposed attribute loss is the sum of the losses of each GAN when creating a GAN for each attribute, and since each GAN shares a hidden layer, it does not increase the amount of computation much. Also, propose simplified content loss, which reduces computation by simplifying reconstruction loss. Reconstruction loss of StarGAN goes through the generator twice, while simplified content loss goes through only once, reduce the amount of computation. Also, propose an architecture that prevents background distortion through image framing and improves training speed through a bidirectional progressive growing generator.

1. ACL-GAN

StarGAN [1] uses an adversarial loss of WGAN-GP [2], classification loss of conditional GAN [3], and reconstruct loss of CycleGAN [4] to train multi domain image-to-image translation GAN. Following formulas are losses of StarGAN.

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

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

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

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

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

$$L_{adv}^d = -L_{adv}$$

$$L_{adv}^g = L_{adv}$$

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 $x\sim P_r(x)$, x is real data.

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 the loss such as LSGAN [5] or WGAN-GP 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 because of cross-entropy measures only the KL-divergence.

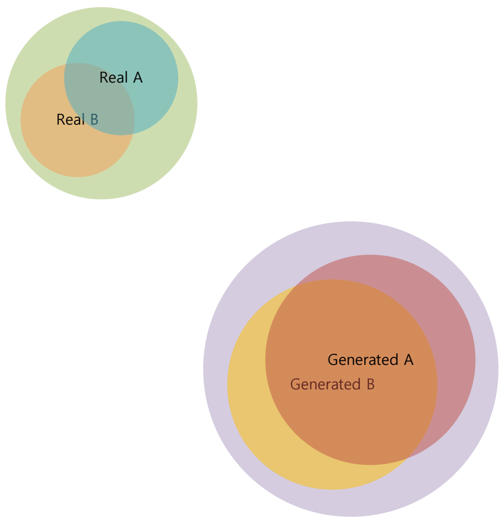


Fig1. Data distribution at the beginning of training

In the above figure, the big circle containing Real A and Real B is the distribution of the real data, and another big circle containing Generated A and Generated B is the distribution of the generated data. Real A is real data with attribute A and Generated A is data generated by the generator with condition A. In the early stage of learning, the classification loss does not produce meaningful gradients because the distance between Real A and Generated A, and Real B and Generated B. Only adversarial loss produces meaningful gradients.

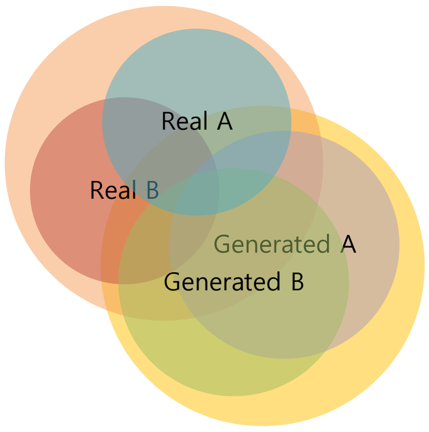


Fig2. After some training

As learning progresses to some degree with adversarial loss, when the real data distribution and the generated data distribution become somewhat similar, the classification loss 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 weight, classification weight. If adversarial weight is too bigger than the classification weight, the data would not have the target condition. If classification weight is too bigger than adversarial 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.

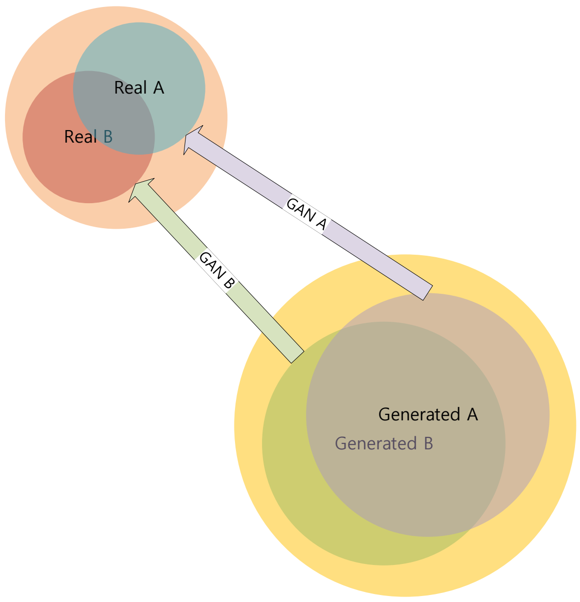


Fig3. Attribute Loss

Attribute loss is 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.

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

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

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'. '$\cdot$' is an inner product.

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

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

Also, in attribute GAN, when the output of discriminator A is 0, that does not mean input data does not have attribute A. It means input data tried to make data that have attribute A, but that does not look real. Therefore, to train attribute not-A, new GAN which trains attribute not A should be added.

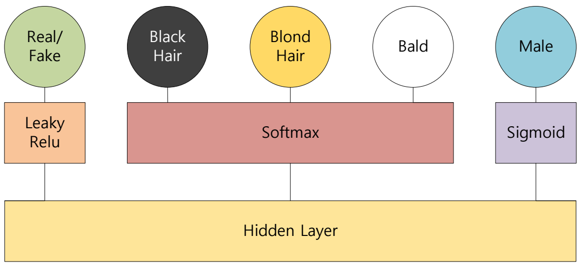


Fig4. conditional GAN discriminator output example

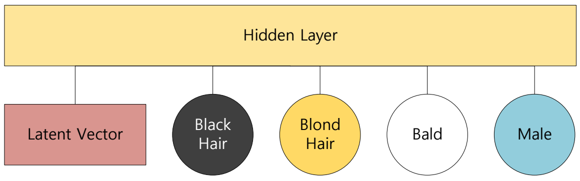


Fig5. conditional GAN generator input example

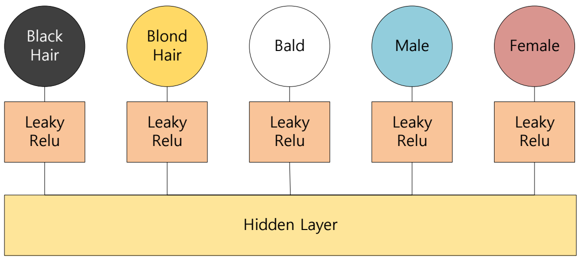


Fig6. attribute GAN discriminator output example

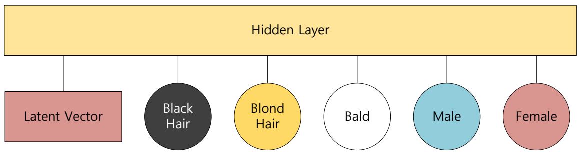


Fig7. attribute GAN generator input example

(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 weight, while conditional GAN loss has two hyperparameters: adversarial weight, classification weight.

Simplified Content Loss

The original StarGAN paper uses reconstruction loss from CycleGAN to ensure generated image not too different from the original image.

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

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. Generally use random binary vector for train.

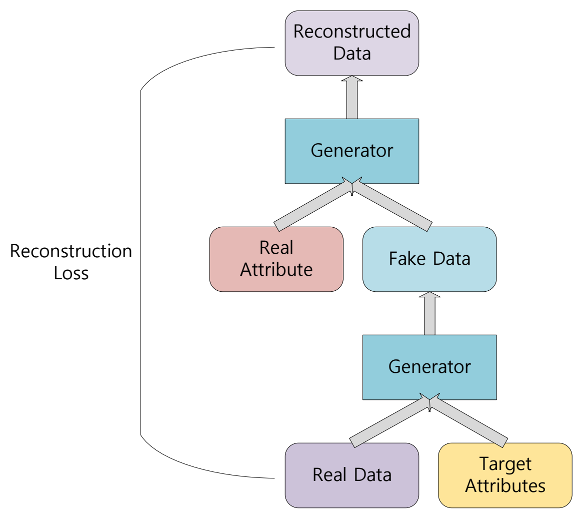


Fig8. Original reconstruction loss of StarGAN

However, since the original image and the generated image need only be somewhat similar, the real data does not have to go through the generator twice.

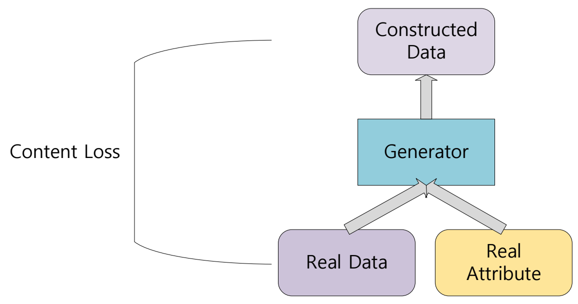


Fig9. Simplified content loss

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 can be reduced.

ACL-GAN

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

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

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

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

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

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

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

Image Framing

As image completion imply [10], while training, framing the generated image with the original image makes generated image match to the background of the original image.

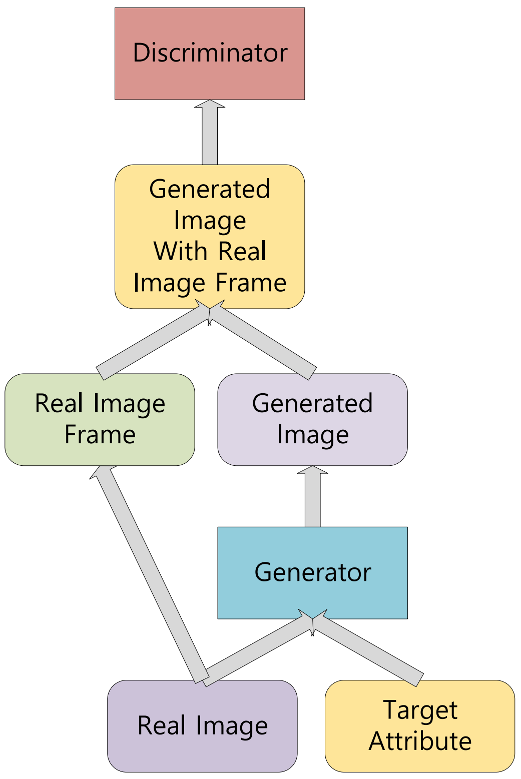


Fig10. Image Framing

Architecture

Generator

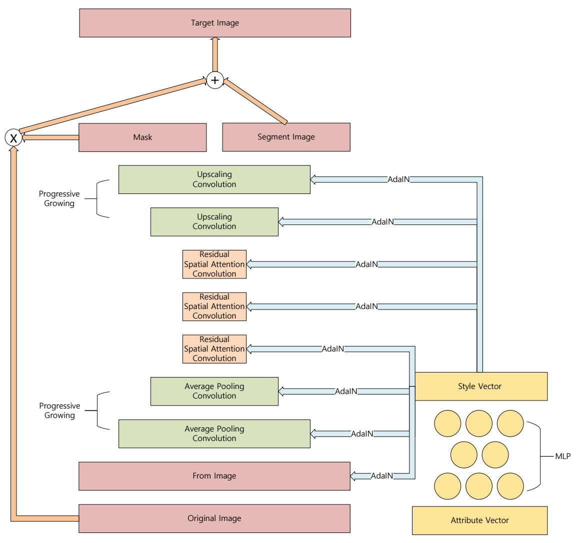

Target Image
+
X
Mask
Segment Image
Progressive Growing
Upscaling Convolution
Upscaling Convolution
Residual Spatial Attention Convolution
Residual Spatial Attention Convolution
Residual Spatial Attention Convolution
Average Pooling Convolution
Average Pooling Convolution
From Image
Original Image
AdaIN
Style Vector
MLP
Attribute Vector

Fig9. Generator Architecture

In generator architecture, the AdaIN module and embedder of Style-based generator [6], mask of CAGAN [7], and convolution block attention module of CBAM [8] 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.

Discriminator

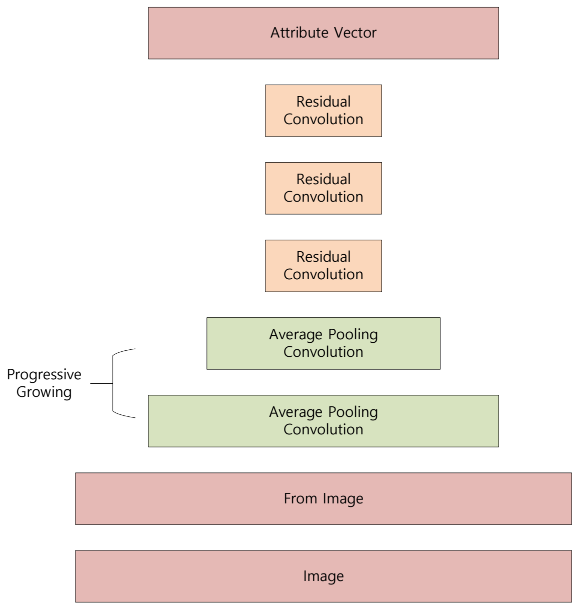


Fig10. Discriminator Architecture

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.

Mixed batch training

Recently, using batch normalization has

become orthodox. However, for the conditional GAN, using batch normalization in discriminator could be dangerous because the target attribute distribution and real data attribute distribution could be different. If there is batch normalization in discriminator, generated data batch will try to follow real data batch. This means condition distribution of generated data batch tries to follow condition distribution of real data batch, not target condition distribution. This means the generator is likely to ignore some target attributes.

Just adjusting target attribute distribution as real attribute distribution could cause imbalanced training. For example, if the ratio of attribute A: not-A of real data is 99:1, the target attribute ratio will become as same. This makes GAN difficult to learn attribute not-A.

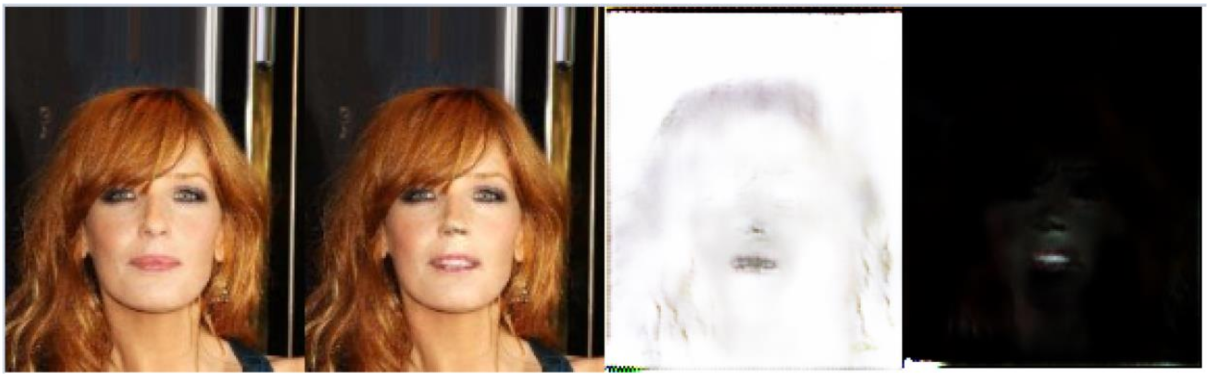
To avoid this problem, I suggest making a batch with a constant ratio (mostly 1:1) of real data and fake data to train the discriminator.

2. Experiments

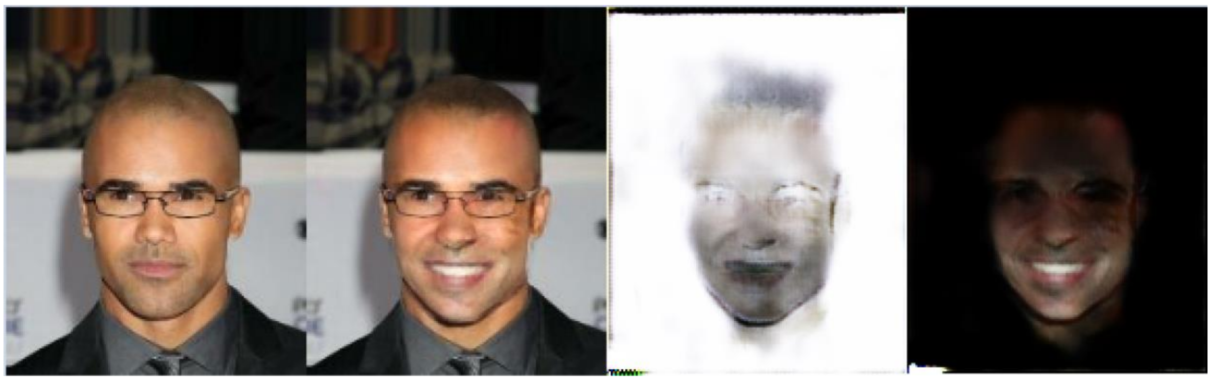
Used celeb_a [9] training dataset (162,770 pictures with attribute label) while training. Used celeb_a test dataset (19,962 pictures with attribute label) for test.

Model was trained almost one day on two rtx2080ti. Resolution of all image is 172 by 144. Trained six attributes(black hair, brown hair, blond hair, mouth slightly open, smiling, rosy cheeks).

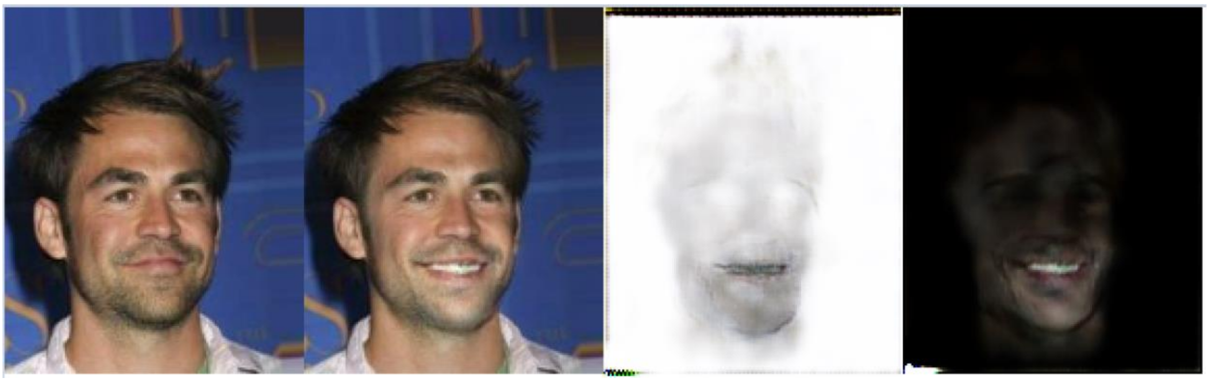
All first pictures are original pictures, second pictures are generated pictures, third pictures are mask images, and fourth pictures are generated segment images.



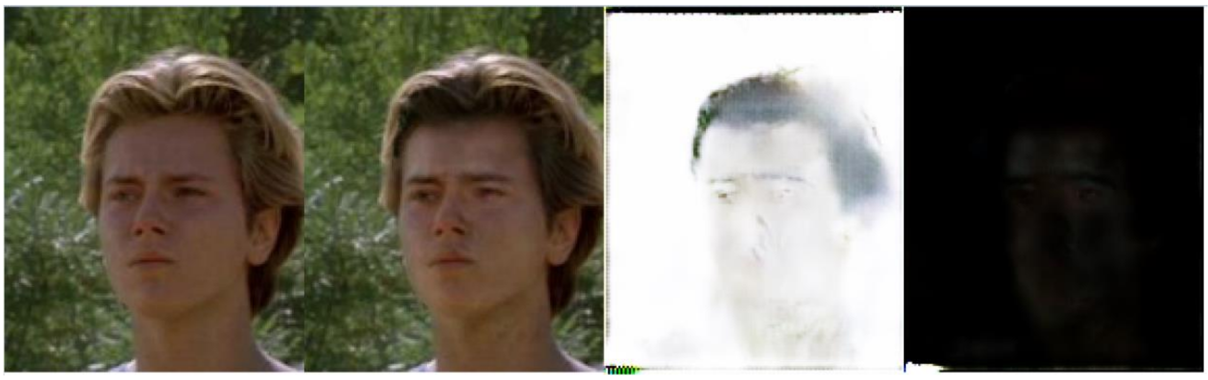
Target attribute: brown hair, mouth slightly open, not smiling, rosy cheeks



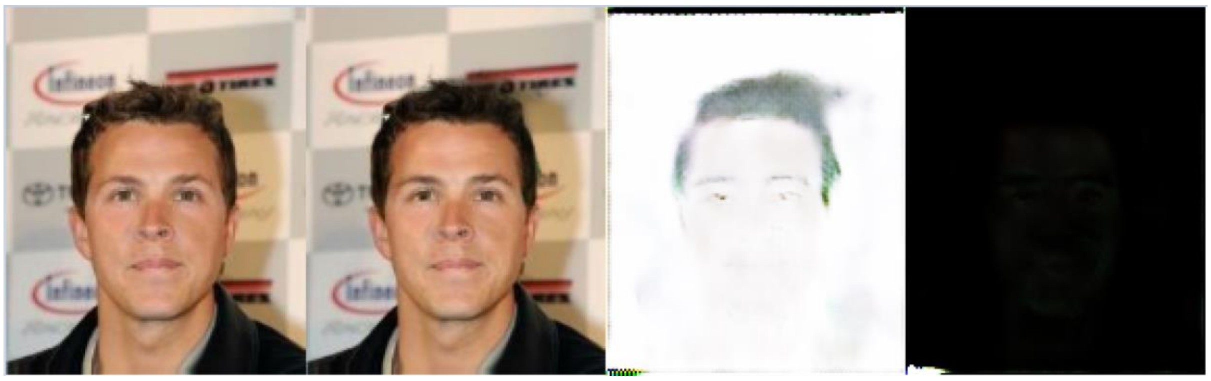
Target attribute: black hair, mouth slightly open, smiling, rosy cheeks



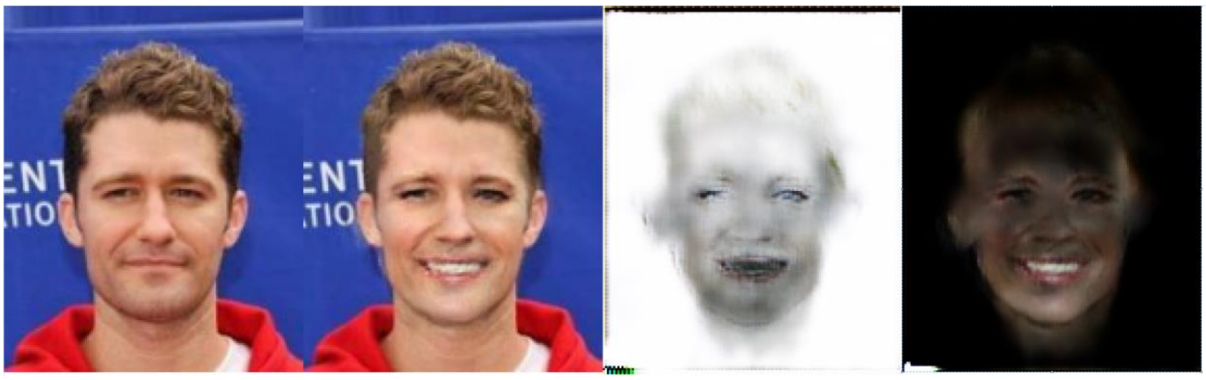
Target attribute: brown hair, mouth slightly open, smiling, not rosy cheeks



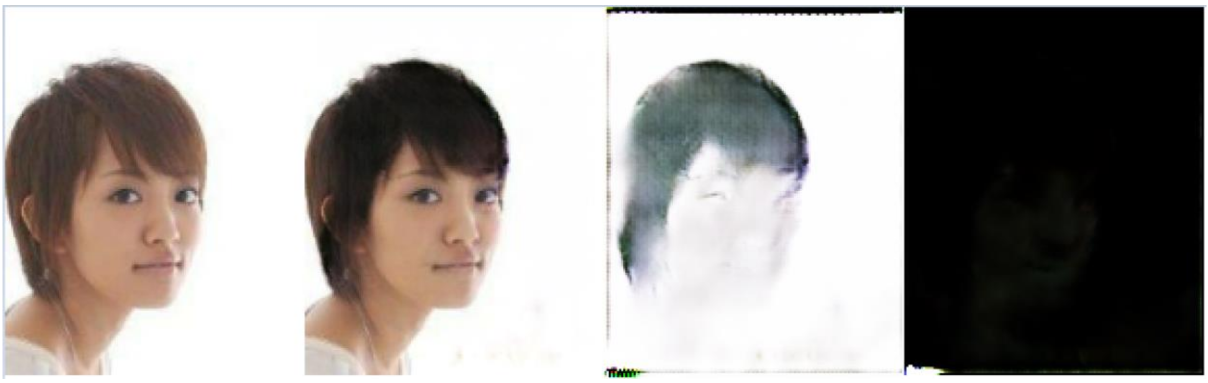
Target attribute: black hair, not mouth slightly open, not smiling, not rosy cheeks



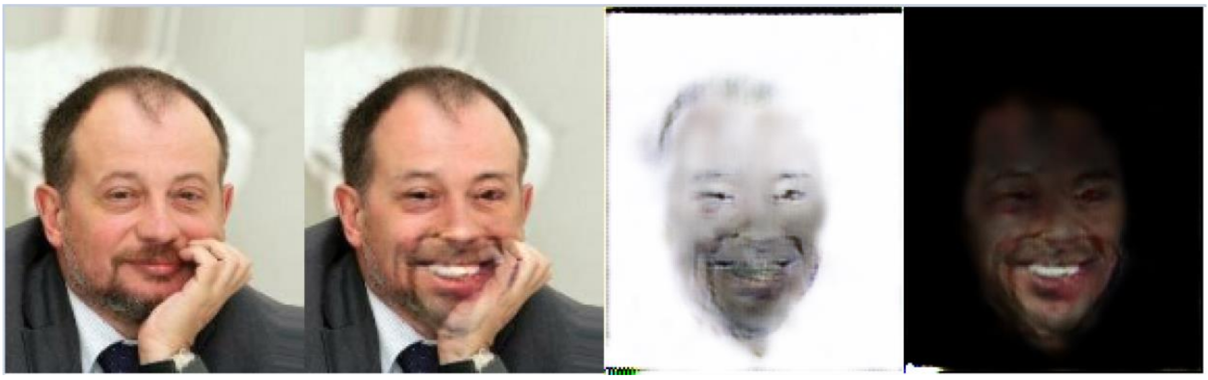
Target attribute: black hair, not mouth slightly open, smiling, not rosy cheeks



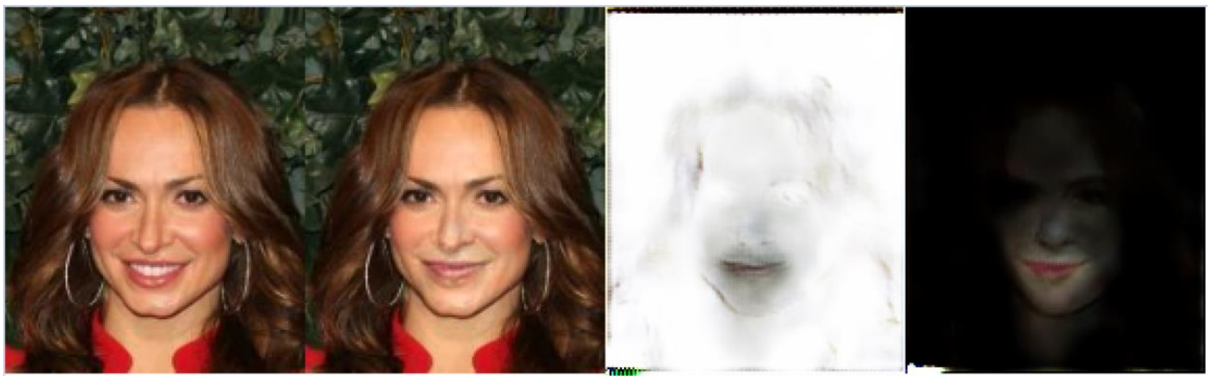
Target attribute: blond hair, mouth slightly open, smiling, rosy cheeks



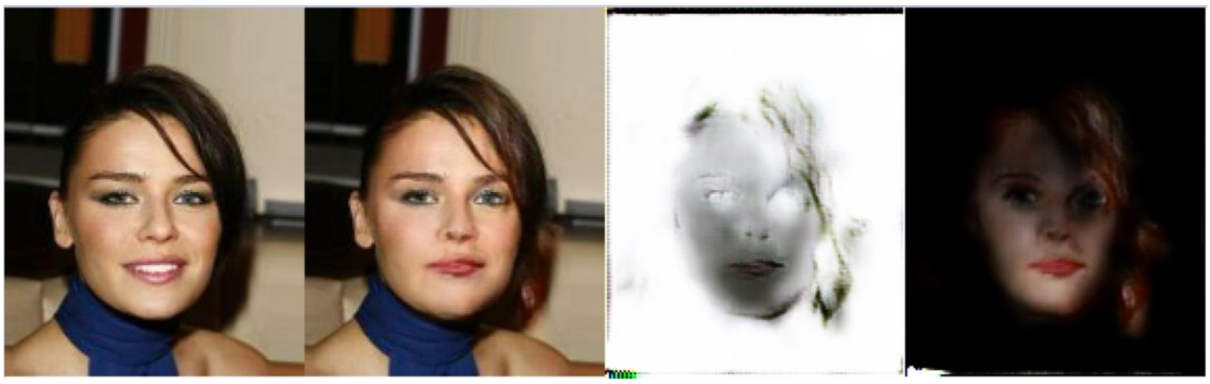
Target attribute: black hair, not mouth slightly open, smiling, not rosy cheeks



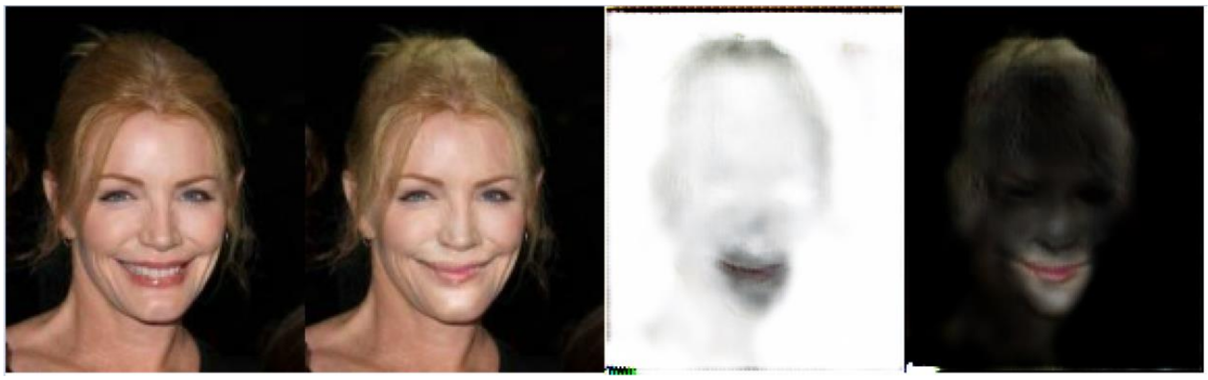
Target attribute: black hair, mouth slightly open, smiling, rosy cheeks



Target attribute: brown hair, not mouth slightly open, smiling, not rosy cheeks



Target attribute: brown hair, not mouth slightly open, not smiling, rosy cheeks



Target attribute: blond hair, not mouth slightly open, smiling, not rosy cheeks

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

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

Improved Training of Wasserstein GANs

https://arxiv.org/abs/1704.00028

[3] Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros

Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

https://arxiv.org/abs/1703.10593

[4] Mehdi Mirza, Simon Osindero

Conditional Generative Adversarial Nets

https://arxiv.org/abs/1411.1784

[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] Tero Karras, Samuli Laine, Timo Aila

A Style-Based Generator Architecture for Generative Adversarial Networks

https://arxiv.org/abs/1812.04948

[7] 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

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

CBAM: Convolutional Block Attention Module

https://arxiv.org/abs/1807.06521

[9] Ziwei Liu   Ping Luo   Xiaogang Wang   Xiaoou Tang

Large-scale CelebFaces Attributes (CelebA) Dataset

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

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

Globally and locally consistent image completion

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