Improved Multi-Domain Image-to-Image Translation GAN

Jeongik Cho

Abstract

StarGAN has shown excellent performance in image-to-image translation using adversarial, reconstruction, and classification losses in multi-domain image-to-image translation. The Style-Based Generator Architecture boosts generator performance through the Embedder and AdaIN modules. I propose an attribute loss, improved version of conditional GAN loss, which is like having multiple GANs, and improved reconstruction loss to speed up learning speed and improve image quality. And I also suggest bi-directional progressive growing generator architecture to improve learning speed.

1. Introduction

StarGAN [1] uses an adversarial loss of WGAN-GP [2], reconstruct on the loss of CycleGAN [3] and classification loss of conditional GAN [4] to transfer the image to target domain without much distortion.

$$L_D = -L_{adv} + \lambda_{cls} L^r_{cls}$$

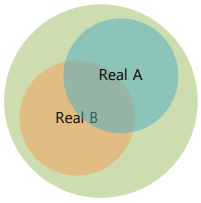
$$L^r_{cls} = E_{x,att\sim P_r(x,att)}[-\log\left(D_{cls}(\text{att}|\text{x})\right)]$$

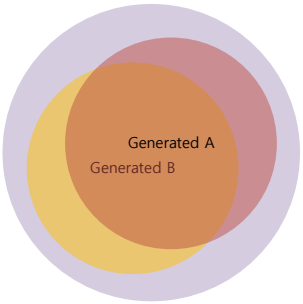
$$L_G = L_{adv} + \lambda_{cls} L^g_{cls}$$

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

In $x, att\sim P_r(x, att)$, x means real data, and att is the binary vector that expresses the attribute 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' .

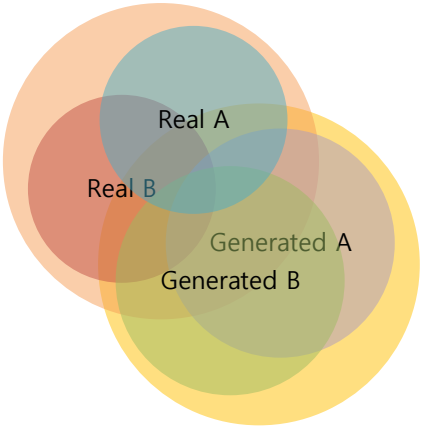
In the conditional GAN, adversarial loss trains model well because there are well known the loss such as LSGAN [5] or WGAN-GP that can produce meaningful gradients even if real data distribution and generated data distribution are far from each other. However, classification loss of conditional GAN which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





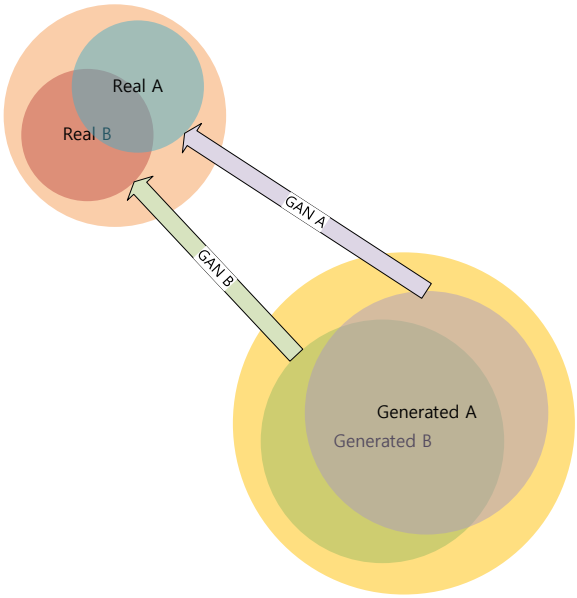
In the above figure, the circle containing Real A and Real B is the distribution of the real data, and the circle containing Generated A and Generated B is the distribution of the generated data. Real A is real data with attribute A and Generated A is data generated by the generator with condition A. In the early stage of learning,

the classification loss does not produce meaningful gradients because the distance between the real data distribution and the generated data distribution is far. Only adversarial loss produces meaningful gradients.



As the learning progresses to some extent, the actual data distribution and the generated data distribution are somewhat similar, and classification loss starts to produce meaningful gradients when each conditional data distribution overlap (Real A and Generated A, Real B and Generated B).

To solve the problem that classification loss does not have meaning at the beginning of learning, I propose attribute loss which is similar to having many GANs that each GAN learn only one attribute. Each Generator only generates data with each attribute. Each Discriminator determines that it is true only for the real data with each attribute and that it is a fake for the data that the generator generates for each attribute. Attribute loss can replace adversarial loss and classification loss of StarGAN.

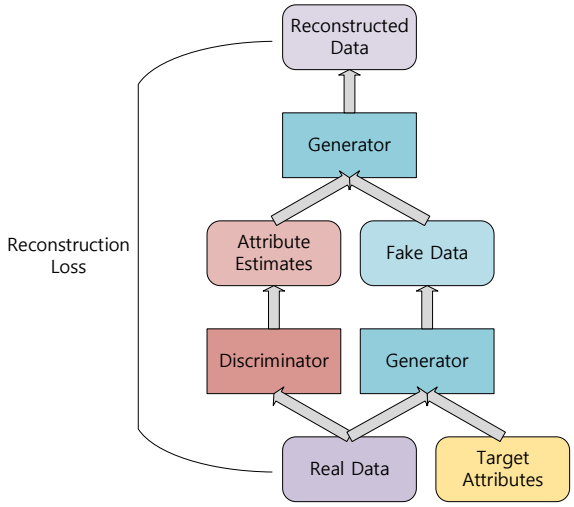


Attribute loss is the sum of each GAN loss. Each GAN has its adversarial loss. Therefore, using LSGAN loss or WGAN-GP loss for each GAN can generate meaningful gradients at the beginning of learning. Since each discriminator shares all layers except the output layer, and each generator shares all layers except the input layer, the learning time does not increase significantly. Also, attribute loss can replace adversarial loss and classification loss, which means one hyperparameter can be removed.

In CycleGAN, the reconstructed image is generated with the original attribute vector.

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

In $x, att\sim P_r(x, att)$, att is the original attribute of real image x, and att' is the target attribute. In most data, the original attribute is a discrete value of 0 or 1. In reality, however, like there are a wide smile and a light smile, the ground truth attribute is a continuous value, not a discrete value. Therefore, restoring the original image with the binary attribute vector can be unreasonable to the generator.



To make reconstruction loss more reasonable, I suggest improved reconstruction loss which is using attribute estimates by discriminator, instead of the original binary attribute vector. When generating reconstructed data, using attribute estimates by the discriminator instead of original binary attribute vector can produce more reasonable reconstruction loss.

In generator, I used embedder and generator architecture that is a simplified version of Few-Shot Adversarial Learning of Realistic Neural Talking Head Models [6], U-Net architecture of Pix2Pix [7], AdaIN module and embedder of Style-based generator [8], activation functions of DCGAN[9], and convolution block attention module of CBAM [10].

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

2. Improved Star GAN

First, it is assumed that attribute information is matched with real data.

2.1 Loss

Overall Loss is as follows.

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

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

Attribute Loss

Attribute loss is as follows.

$$L_{att}^D = \sum_c^{att} L_c^D$$

$$L_{att}^G = \sum_c^{att} L_c^G$$

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

$$L_c^G = E_{x\sim P_r(x)}[(D_c(G_c(x,1))-1)^2]$$

c means one specific attribute among several attributes. L_c^D and L_c^G are the losses of one discriminator and one generator that discriminate against a particular attribute c. L_{att}^D is the sum of the attribute losses of all discriminators and L_{att}^G is the sum of the attribute losses of all generators.

G_c is a generator that converts an image x to have an attribute c when the image x and 1 are received as inputs. G_c tries to trick D_c only if 1 is entered with x, and does not care if 0 is

entered (not learn).

D_c determines only about attribute c. D_c discriminates real only for real data with attribute c and doesn't care about real data without attribute c and determines fake when received the fake image from G_c that receives real image x and 1.

This is an example of using the least square loss as an adversarial loss. Wasserstein-GP or other adversarial loss can be used for attribute loss, however, to make attribute estimates for reconstruction loss, I used least square loss.

L_{att}^{D} is the sum of each discriminator. Each discriminator shares all layers with other discriminators except the output layer. Considering a set of discriminators as one discriminator, the loss can be changed like below.

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

In $x, att \sim P_r(x, att)$, x is the real image, and att is attribute binary vector. '$\cdot$' means inner product.

In $x', att' \sim P_g(x', att')$, x' is generated image and att' is a binary attribute vector input generator to make x'.

Since each generator also shares all layers except the input layer, L_{att}^{G} can be written as the following by considering a set of generators as one.

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

att' is a binary vector representing the attribute you want to change in the real image x. Use random binary vectors for training.

Incidentally, $G_c(x, 0)$ does not convert x to x' that doesn't have attribute c but simply disables G_c. Therefore, if you want to remove attribute c from image x, you need to add the attribute 'not c' while training.

Reconstruction Loss

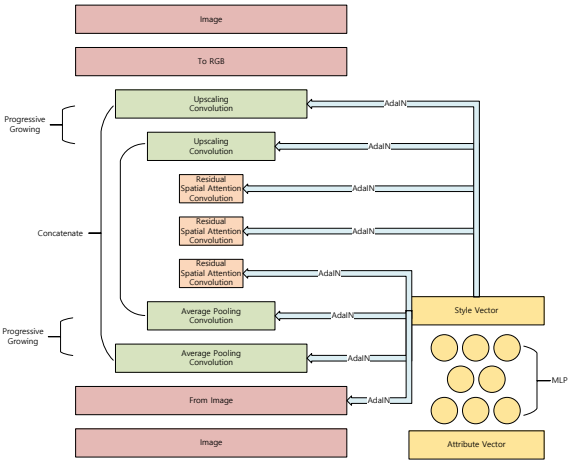
I used improved reconstruction loss that is the improved version of CycleGAN's reconstruction loss.

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

In $x\sim P_r(x)$, x is real image. c' is random binary attribute vector. Original reconstruction loss of CycleGAN uses real attribute binary vector of real image x instead of D(x).

2.2 Architecture

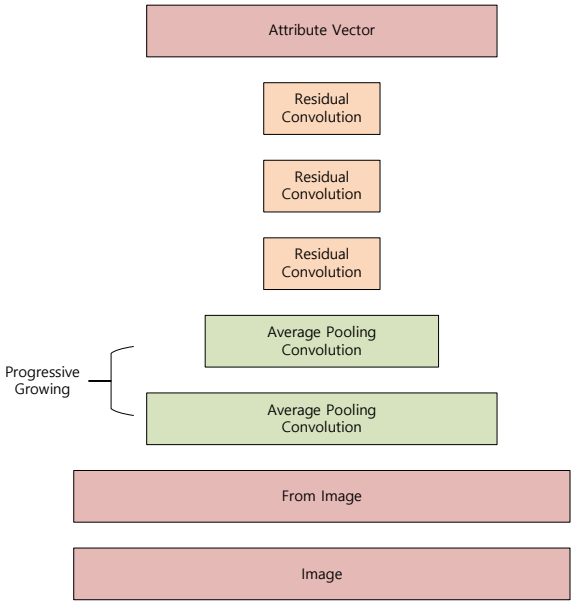
Generator



I used embedder and generator that is a simplified architecture of Few-Shot Adversarial Learning of Realistic Neural Talking Head

Models, U-Net architecture of Pix2Pix, and AdaIN module and embedder of Style-based generator, activation functions of DCGAN, and convolution block attention module of CBAM. AdaIN module is already playing the role of channel attention module of CBAM, so I used only spatial attention module of CBAM. To improve the learning speed, the generator grows in both input and output directions, not just in one direction.

Discriminator



Discriminator has attribute outputs that each output discriminates whether real image with each attribute or generated image with each attribute.

3. Experiments

3.1 Results

I used celeb_a[11] dataset.

Model was trained almost 50hours on rtx2080ti.

All right pictures are original pictures and left pictures are generated pictures.

Results

Generated with black hair, not blond hair, mouth slightly open, smiling, not male.

Generated with black hair, blond hair, not mouth slightly open, smiling, male.

Generated with not black hair, not blond hair, mouth slightly open, smlilng, not male

Generated with black hair, blond hair, not mouth slightly open, smiling, male.

Generated with not black hair, not blond hair, not mouth slightly open, not smiling, male

Generated with black hair, blond hair, mouth slightly open, not smiling, male

Generated with black hair, not blond hair, mouth slightly open, smiling, not male.

3.2 Progressive growing compare

I compared the bi-directional progressive growing model and non-progressive growing model. Both models trained approximately 1900sec on rtx2080ti (1901sec for the bi-directional progressive growing model, 1942sec for non-progressive growing model). The bi-directional progressive growing model learned 2% of celeb A dataset in resolution 18 by 22 with 150 sec, 4% in resolution 36 by 44 with 462sec, 8% in resolution 72 by 88 with 1289sec. The non-progressive model learned 12% of celeb A dataset in resolution 72 by 88 with 1942sec.

Results

Left pictures are results of the non-progressive growing model, middle pictures are a bi-directional progressive growing model, and the right pictures are original pictures.

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] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, Victor Lempitsky

Few-Shot Adversarial Learning of Realistic Neural Talking Head Models

https://arxiv.org/abs/1905.08233

[7] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros

Image-to-Image Translation with Conditional Adversarial Networks

https://arxiv.org/abs/1611.07004

[8] Tero Karras, Samuli Laine, Timo Aila

A Style-Based Generator Architecture for Generative Adversarial Networks

https://arxiv.org/abs/1812.04948

[9] Alec Radford, Luke Metz, Soumith Chintala

Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks

https://arxiv.org/abs/1511.06434

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

CBAM: Convolutional Block Attention Module

https://arxiv.org/abs/1807.06521

[11] Ziwei Liu Ping Luo Xiaogang Wang Xiaoou Tang

Large-scale CelebFaces Attributes (CelebA) Dataset

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