CASL-GAN: multi-domain image-to-image translation GAN

Jeonglk Cho

Konkuk University

Note: Conditional activation GAN was separated from this paper and written as a separate paper (http://vixra.org/abs/1912.0204).

Abstract

StarGAN, which has impressive performance in multi-domain image-to-image translation. Reconstruction loss of StarGAN requires reconstructed data from generated data, which means to get reconstruction loss, need to use the generator once more. Simplified content loss uses already generated data, reduces the amount of computation and memory usage. Also, propose image framing to prevent background distortion.

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 attributes, reconstruct loss to the generated image changes only target attributes, not other hidden attributes.

In this study, used conditional activation GAN [2] loss instead of auxiliary classifier GAN [3] loss of starGAN to reduce hyperparameter and improve training speed.

An image consists of attributes. The StarGAN uses reconstruction loss (cycle consistency loss for StarGAN) to ensure the generated image change only target attributes, not other hidden attributes. If the generator changes the hidden attributes of input data, for example, in face expression change GAN, generated face becomes the face of a different person to input person. However, not want to change any attributes other than the target attributes, 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 target attributes. Simplified content loss can use already generated data for other losses: conditional GAN losses or conditional activation GAN losses, while reconstruction loss requires a reconstructed data from generated data, thus reduce memory usage and computation amount.

In StarGAN, since adversarial loss and classification loss (or conditional activation 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 generated image hardly changes. Instead of raising reconstruction loss weight (or simplified content loss weight), image framing can prevent background distortion. As image completion GAN [4] 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.

2. CASL-GAN

2.1 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.

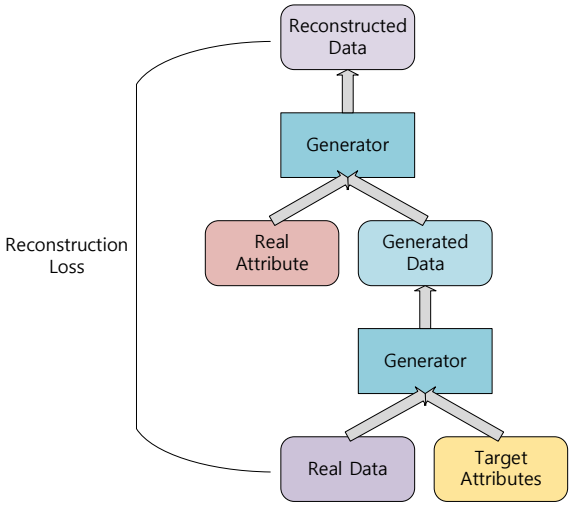

Fig1. Reconstruction loss of StarGAN

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

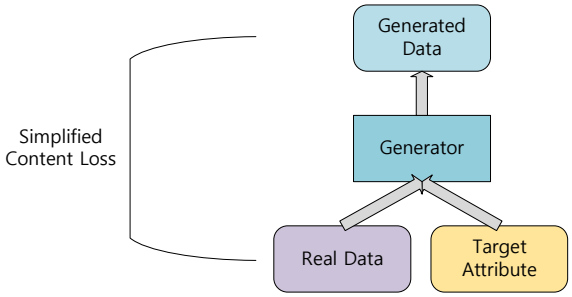

Fig2. Simplified content loss

I suggest simplified content loss that is difference between generated data and real data.

$$L_{sc} = E_{x',att'\sim P_g(x',att')}[\|x' - x\|_1]$$

Since simplified content loss uses already generated data for other losses: conditional GAN losses or conditional activation GAN loss, the calculation amount and memory usage can be reduced. For example, if simplified content loss uses generated data for conditional activation GAN, the generator loss formula is as follows.

$$L_{ca}^{G} + \gamma_{sc} L_{sc} = E_{x', att' \sim P_g(x', att')} \left[\left(D(x') - 1 \right)^2 \cdot att' + \gamma_{sc} \| x' - x \|_1 \right]$$

L_{ca}^{G} is conditional activation loss for generator and L_{sc} is simplified content loss. γ_{sc} is simplified content loss weight.

Simplified content loss prevents both target attributes and hidden attributes from changing. Thus, high simplified content loss weight can cause the image not to change. However, using appropriate simplified content loss weight can guarantee the immutability of hidden attributes.

2.2 CASL-GAN Loss

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

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

$$L_G = L_{ca}^{G} + \gamma_{sc} L_{sc}$$

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

$$L_{ca}^{G} + \gamma_{sc} L_{sc} = E_{x', att' \sim P_g(x', att')} \left[\left(D(x') - 1 \right)^2 \cdot att' + \gamma_{sc} \| x' - x \|_1 \right]$$

2.3 Image Framing

In StarGAN, since adversarial loss and classification loss (or conditional activation 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.

Fig3. Background distortion of StarGAN. Captured from the paper of StarGAN

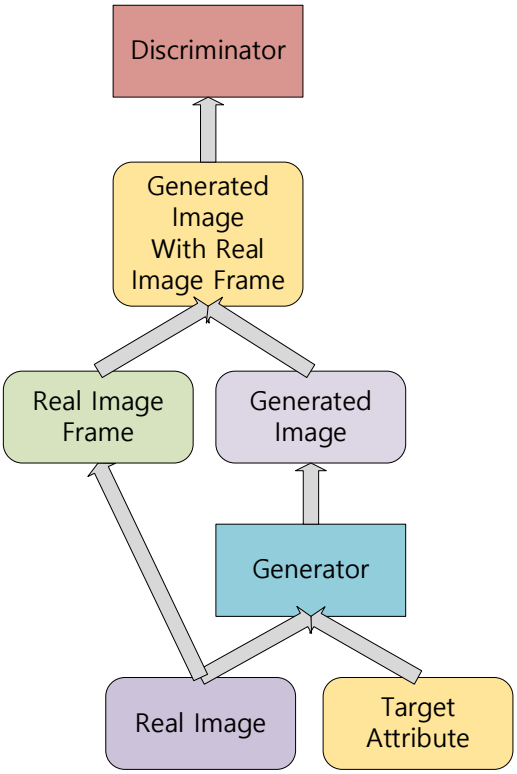

Fig4. Image Framing

2.4 Architecture

2.4.1 Generator

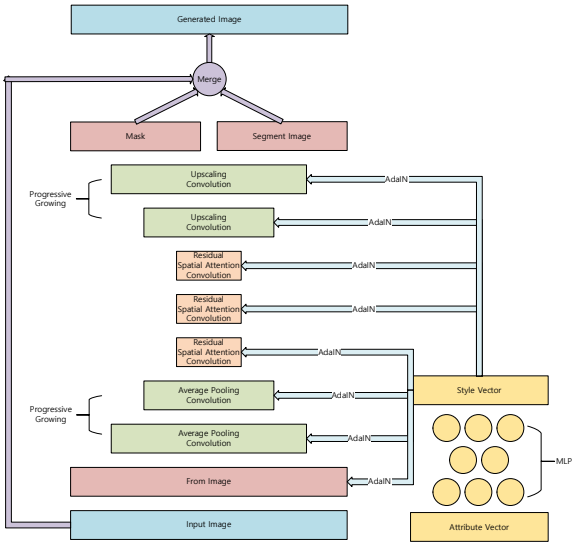

Fig5. Generator Architecture

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.

2.4.2 Discriminator

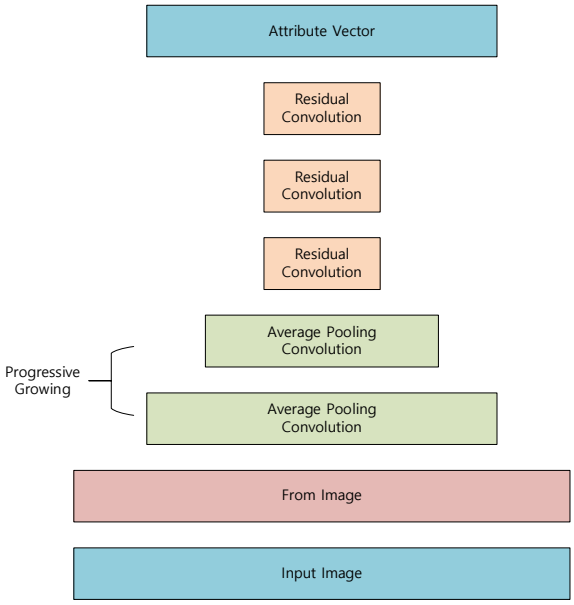

Fig6. Discriminator Architecture

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 [9] 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 6 attributes:

black hair, blond hair, brown hair, smiling, mouth slightly open, rosy cheeks. Used two rtx2080ti with Tensorflow 2.0. Image resolution is 144 by 176.

Used Adam [10] optimizer. Does not used progressive-growing generator. However, the architecture is a four-growth architecture: start from 9 by 11 resolution.

$$discriminator\ and\ generator\ optimizer = adam\ optimizer\begin{pmatrix} learning\ rate = 3e-6, \\ \beta_1 = 0.9, \\ \beta_2 = 0.999, \\ decay = 0 \end{pmatrix}$$

$$attribute\ loss\ weight = 1$$

$$simplified\ content\ loss\ weight = 1e-9$$

$$batch\ size = 16\ for\ each\ GPU$$

$$image\ frame\ pixel\ size = 4$$

Trained 62 epochs for 31 hours.

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.

4.1 Good cases

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

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

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

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

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

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

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

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

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

4.2 Bad cases

4.2.1 Attribute ignored cases

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

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

Target attributes: brown hair, mouth slightly open, smiling, rosy cheeks

In these cases, one or more target attributes were ignored

4.2.2 Frame problem cases

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

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

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

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

4.2.3 Unnatural cases

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

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

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

In these cases, images do not look natural.

4.2.4 Mistake cases

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

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

In these cases, the generator misrecognized the part that is not a face.

4.2.5 Corrupted cases

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

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

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

In these cases, the image is completely corrupted.

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.

4.3 Speed comparison

	Unit	Simplified content loss			Reconstruction loss			Re-reconstruction loss		
Resolution	Pixel	176x144	88x72	44x36	176x144	88x72	44x36	176x144	88x72	44x36
Epoch 1	Sec	14.5639	10.0948	7.8109	20.0343	13.4815	9.2857	25.1710	16.2387	11.1615
Epoch 2	Sec	14.8430	10.1875	7.6440	20.1300	13.3820	9.3325	25.6395	16.3295	11.0112
Average	Sec	14.7035	10.1412	7.7275	20.0822	13.4318	9.3091	25.4053	16.2841	11.0864

Fig7. Speed comparison table

Used 1% of Celeb_A train dataset. Other hyperparameters are same as in training. Re-reconstruction loss uses reconstructed image of reconstructed image.

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

Re-reconstruction loss has no meaning for training but experimented with it for speed comparison. Also compared the speed when the GAN did not grow completely. As the number of times the image passes through the generator increases, it can be seen that the training time increases linearly. In high resolution, Simplified content loss is almost 32~37% faster than reconstruction loss per epoch.

5. Discussion and Future works

Experiments show that Simplified content loss can replace reconstruction loss. Although whether the simplified content loss improved the training speed is not tested, reduction in training time per epoch was observed. A further experiment is needed to compare actual training speed with reconstruction loss and with simplified content loss.

Although image framing can prevent background distortion, there is a problem that pixels that are in the frame or near frame do not change properly.

Some images convert correctly, but some do not. In particular, when many attributes are changed, the target attributes are ignored or generate odd images. Research on how to correctly convert any image is necessary.

Unlike progressive-growing GAN [5], used the bi-directional progressive growing generator to improve speed. However, this paper did not experiment whether bi-directional progressive-growing GAN improves training speed. An experiment comparing three types of generator: non-progressive growing, progressive growing, and bi-directional progressive growing, is necessary.

6. Funding

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

7. Appendix

7.1 Generator Architecture

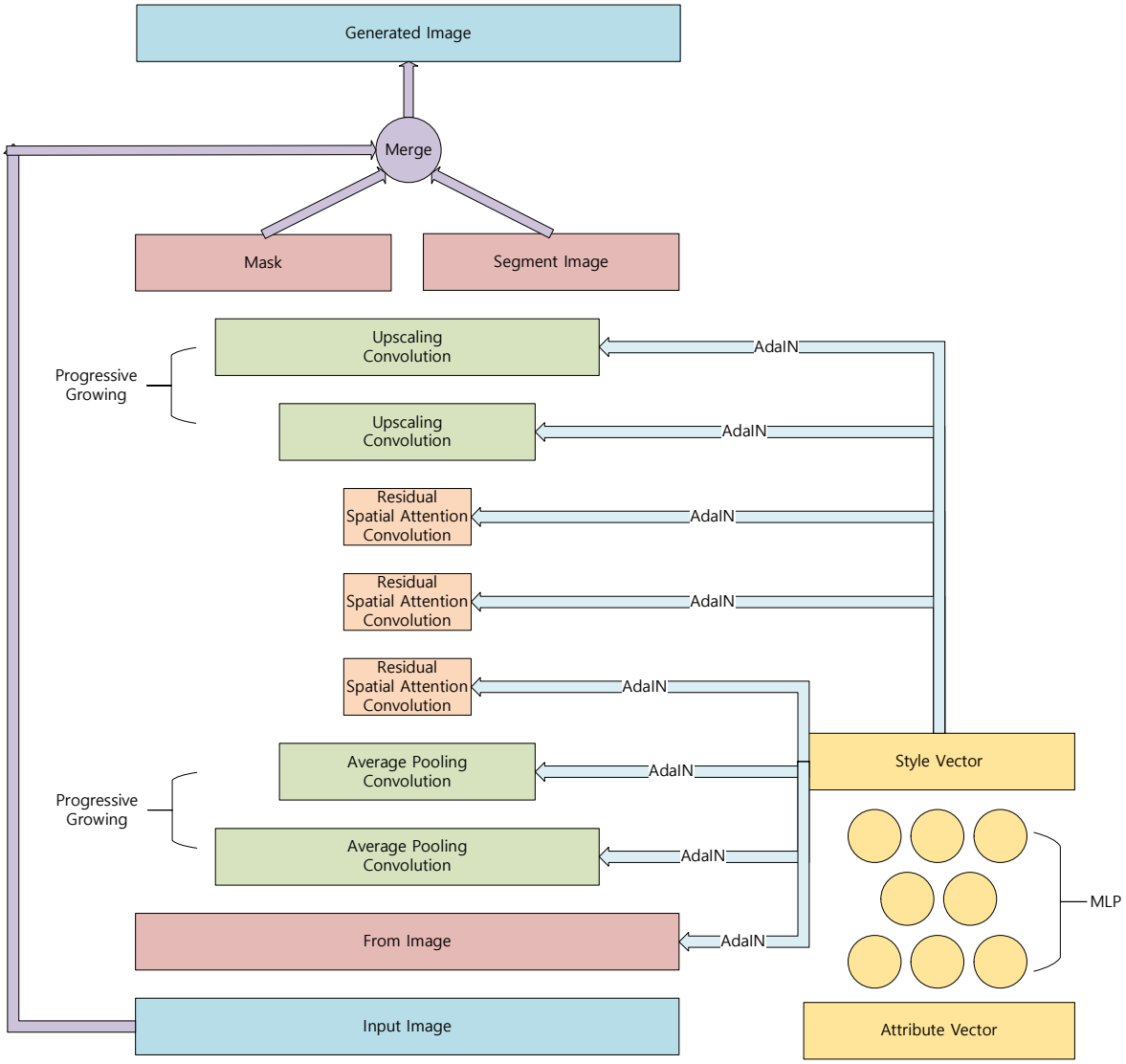

Fig8. Generator Architecture

Generated Image = Input Image * Mask + Segment Image * (1 - Mask). Unlike CAGAN, mask is three channels.

Used bi-directional progressive growing generator architecture.

7.1.1 Residual spatial attention convolution

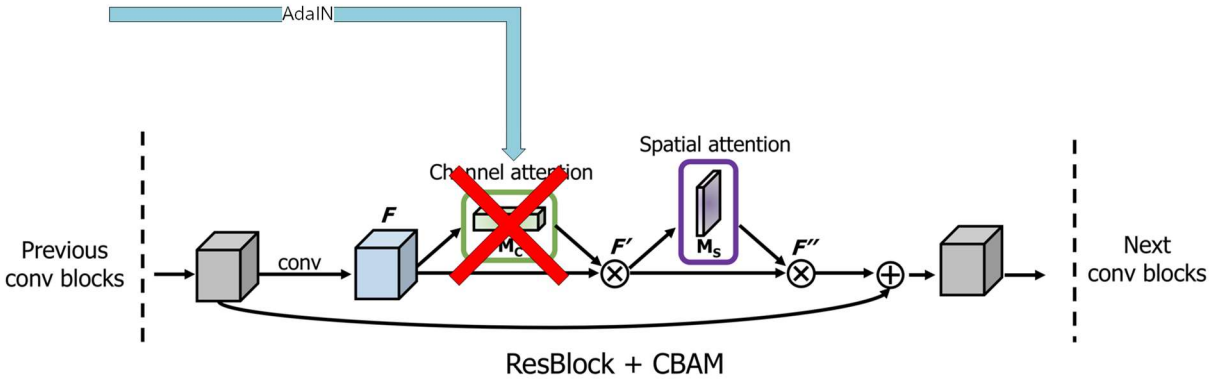

Fig9. Residual spatial attention convolution

Because the roles of AdaIN and channel attention overlap, AdaIN module replaced channel attention module in CBAM.

7.2 Discriminator architecture

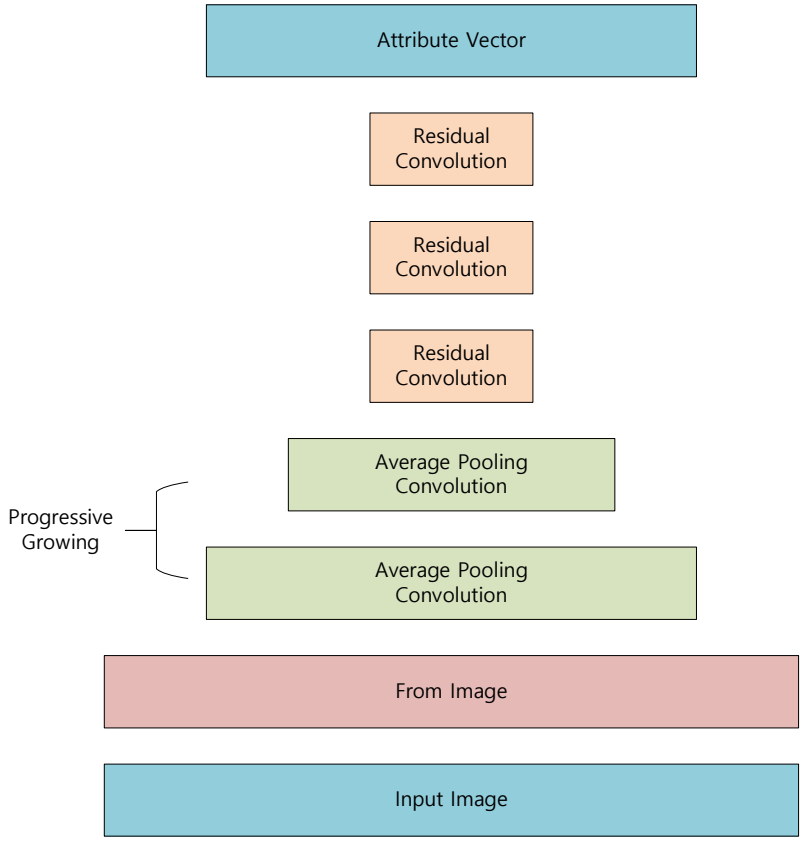

Fig10. Discriminator architecture

8. 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] JeongIk Cho

Conditional Activation GAN: Improved Auxiliary Classifier GAN

http://vixra.org/abs/1912.0204

[3] Augustus Odena, Christopher Olah, Jonathon Shlens

Conditional Image Synthesis With Auxiliary Classifier GANs

https://arxiv.org/abs/1610.09585

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

Globally and locally consistent image completion

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

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

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

https://arxiv.org/abs/1710.10196

[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] Diederik P. Kingma, Jimmy Ba

Adam: A Method for Stochastic Optimization

https://arxiv.org/abs/1412.6980