We know what Glasses are good for you

Image-to-Image generation using cycle GAN

Hyukhun Koh **Junbeom Kwon**

Machine Learning Programming term project, Sogang University

Figure 1. Each person in the figure is Hyukhun Koh and Junbeom Kwon. Images on the right side were generated by cycleGAN.

Abstract

Choosing the best glasses is essential for making a good impression. However, previous attempts to recommend glasses have been insufficient. To get recommendations about glasses from AI , we used cycleGAN, one of the most prominent image-to-image translation models. Using four types of datasets, we trained our model to add realistic and fashionable glasses on the faces. In our experiment, we found that the model trained with both aihub and kaggle dataset achieved the best performance.

1. Introduction

Have you ever gone through the hassle of finding the best glasses for you? It is hard to decide what glasses to buy because there are many characteristics that have an influence on face styles related to glasses and faces. For example, people with round faces do not prefer to put on round glasses because that kind of glasses can make their round faces stand out. In addition, other attributes such as transparency, thickness, and color contribute to the overall impression of a person. In many cases, opticians or ophthalmologists give some guidance about what glasses to buy. However, they usually have knowledge only about vision, not fashion, which can result in unpleasant decisions about glasses.

What if we can get guided by AI? We tried to tackle this task with GAN(Generative Adversarial network), a generative model that can generate new images by estimating the distribution of the training datasets. GAN is adversarial in that it has two competing model components, generator and discriminator. The generator is trained to make realistic fake images similar to real images. The discriminator is trained to distinguish real images from fake images. In this way, we can generate the most seemingly realistic images.

Among the various tasks using GAN, our task can be defined as image-to-image translation. Image-to-image translation is converting the style of a given image to another style. Whereas we

cannot control the style of images in vanilla GAN, other GANs for image-to-image translation can generate specific styles of images by specifying some conditions for the task.

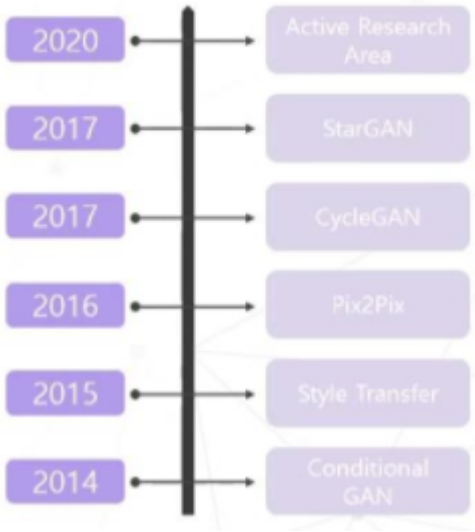

figure 2. Image-to-Image Translation Timeline

In 2014, Conditional GAN was introduced to use conditions for image generation. Replacing labels for each image to embedding vectors, it could effectively generate images corresponding to the labels. After that, GANs for image-to-image translation targeted on style transfer, in which a model can change the style of given images to another style. In this stage, researchers had produced remarkable translation modules in supervised settings, where they used paired datasets to train their models.

However, obtaining paired datasets was challenging for early models like pix2pix. Only few datasets were available for the tasks, so configuring early models was highly limited. In addition, when the generated output was not firmly defined, it was difficult to apply pix2pix. For example, it is hard to make certain transferred images using input images with respect to a Gogh-style. In this case, the pix2pix model is not applicable.

For this reason, we decided to use a more recent model, cycleGAN. By using cycleGAN, we could save much time and effort but with higher quality. We could collect much larger amounts of datasets from various sources without seeking paired images of the same person. (Figure3)

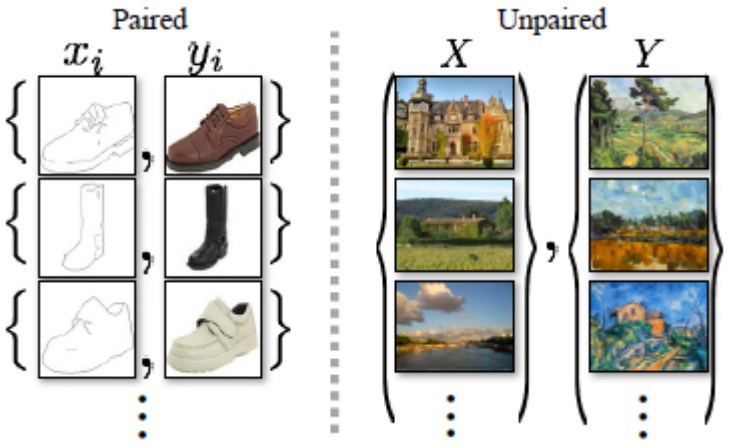

Figure 3. the figure from the original cycleGAN paper that compares paired and unpaired images.

In this paper, we will share our experience of applying cycleGAN to generate pictures of faces with glasses. Dealing with many kinds of datasets and models, we have overcome several issues for GAN. We strongly believe that our project not only has practical meaning, but also gives us an opportunity to get a deep understanding of the state-of-art image generation model.

2. Data collection and preprocess

Now, we will cover how we collected our datasets and how we processed them. To make an image-to-image translation model, we needed paired/unpaired image datasets. After searching for some images in google or open source in any image competition, we determined to use four sources of images. We included images from “Kaggle glass classification”, “AI hub facial images”, open source images “celebA”, and finally, “google images” which we collected by ourselves. In addition, when we gathered the image, we did not consider the size or the resolution. Since, in our model, it processes the image to the standard size which is 286*286. Moreover, In the process of image, we randomly cropped to be 256*256 size. Therefore, our final input image size would be 256*256 and the size of generated images also would be identical as input image’s.

All of our collected datasets had different characteristics. So, we will introduce sample datasets and explain how we processed the images such as deleting or adding. The description below says requirements for preprocessing the images.

1. glasses should have full form (not the sunglasses)
2. each image should have one person with two eyes
3. The face should account for three quarters of the picture
4. faces with mask or any coverage should be excluded

1) Kaggle dataset

Figure 4. four samples from kaggle datasets(Glasses or No Glasses)

Total number of kaggle datasets was 5000 with 1024*1024 resolution. There were four types of images, face with glasses, face with sunglasses, face with no glasses and noisy images where some parts of glasses were omitted. So, we personally looked through all of the data to filter out the sunglass images and noise images. We filtered out the sunglasses because features of sunglasses were not the same as those of normal glasses. Including distinct sunglasses and glasses together was problematic because cycleGAN cannot separate multiple styles, which may mix features of sunglasses and glasses recklessly. Entirely, the faces accounted for most parts of the pictures with one face per image. In addition, every image had no coverage.

The problem with the Kaggle dataset was that most of those images are western people. Also, they only show smiley facial expressions. Then if we make the model with only those datasets, the model would not translate the faces of other races or faces with other facial expressions. To target these

issues, we tried to find asian images and finally decided to use an AI hub image dataset.

2) AI hub

AI hub was collected by the government, which required permission to access the datasets. This dataset was composed of 400 people with multiple versions - 3 types of resolution(low-moderate-high), 6 types of accessory, various types of light brightness and camera angles, 3 face styles. Since we wanted to use datasets similar to kaggle datasets, we selected high resolution-glasses/no glasses-highest brightness-3 face styles-centered images. The reason why we chose the highest resolution and brightness was that the model's performance was proportional to resolution and clearness.

Figure 5. one subject in AI hub with different settings

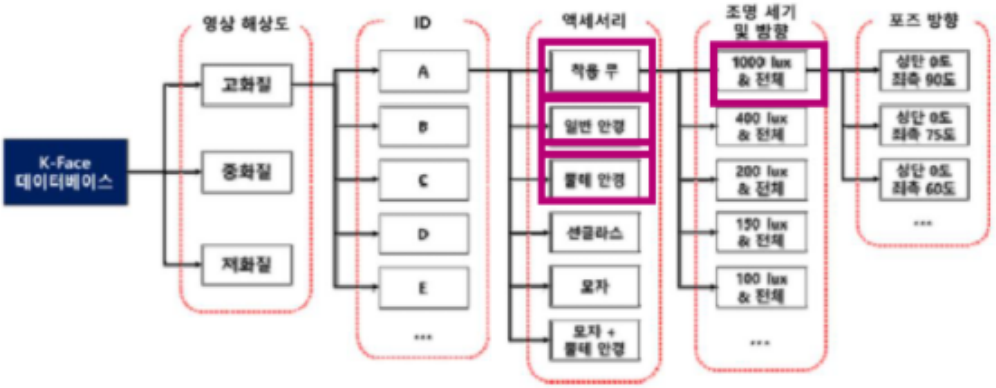

Figure 6. the data structure of AI hub

Pictures in Figure5 are images from the same person with different styles. Collecting images from a lot of folders was confusing and time consuming, so we automated the process with some python codes(crawling_extract.ipynb). We inspected the output images to check whether they met our four criteria. Images had one person without sunglasses or coverage. Then additional preprocessing was unnecessary.

3) celebA

Even though we found several images from AI hub and Kaggle, there were still problems. In general, to train the deep learning models, we need more than 10 thousand samples. That is, more images were required to make our model more general. So we tried to find the huge data. Eventually, we discovered celebA which has 10,177 number of identities, 202,599 number of face images, and 5 landmark locations, 40 binary attributes annotations per image.

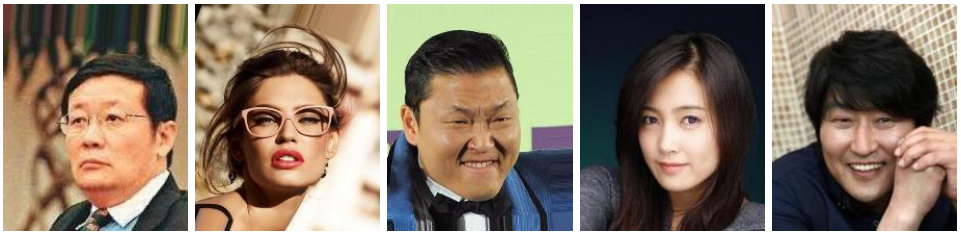

Figure 7. five images from open source "celebA"

Here are samples for celebA data. As you can see, it has various ethnic groups and meaningful face sizes with various facial expressions. Besides, the condition of "one person per image" and "no coverage" was satisfied. But the problem with these images was that their resolution was smaller than other datasets. Therefore, we decided to use those images as a last resort if the model trained with the former two datasets showed bad performances.

4) google crawling

Figure 8. two images crawled from google image search

Finally, we tried to include the latest images. Therefore, we collected the data from google with crawling code(crawling_extract.ipynb). Because most of the images were not suitable for our dataset, we strived to find the appropriate keyword to efficiently preprocess the images. Owing to time restrictions, provided that we cannot preprocess all the images, we just added some good images which met four criteria to the existing dataset as much as we could.

3. Model Architecture

Now, it is time to explain the architecture of cycleGAN. As we discussed earlier, cycleGAN's most useful advantage is that it can be trained with unpaired dataset. To know why cycleGAN can achieve this, we need to understand the details of cycleGAN.

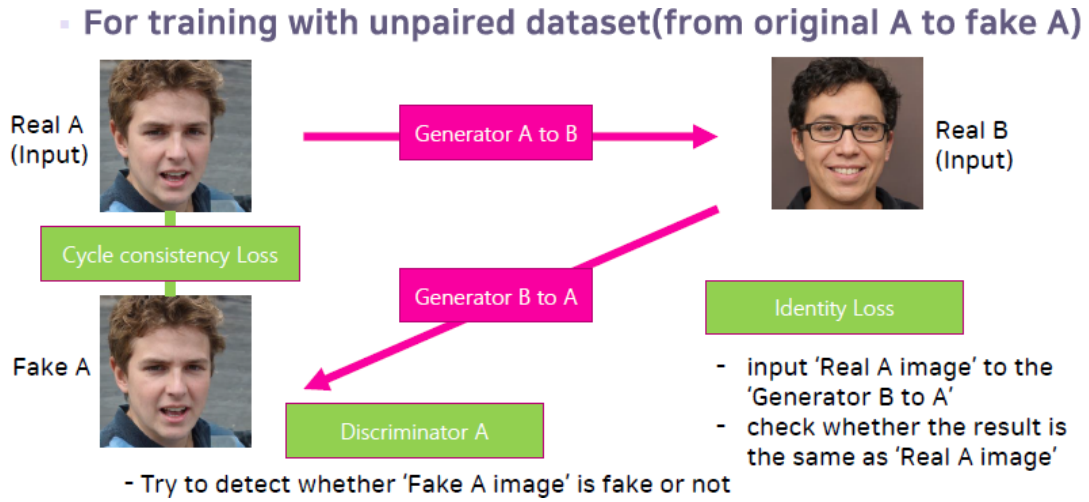

Figure 9. training process of cycleGAN(generating a fake A from a real A)

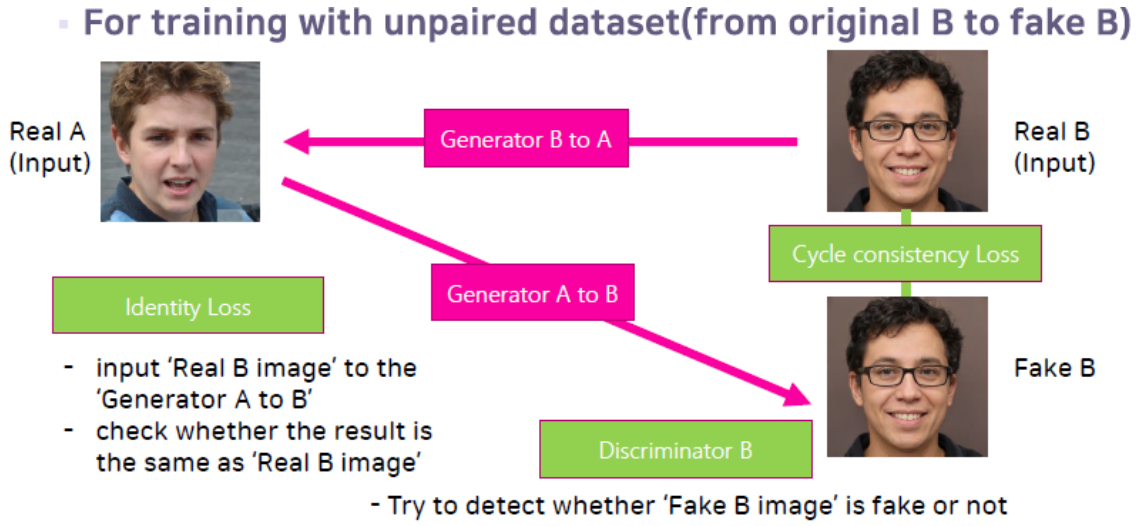

Figure 10. training process of cycleGAN(generating a fake B from a real B)

Our task is to make the model learn how the different domains, X and Y, are mapped. In our project, we can define that domain X includes faces with glasses and domain Y includes faces without glasses. We want to interchange domain X and Y. To learn the relationship between two domains, cycleGAN has a cycle-like training process.(figure 10, 11) CycleGAN consists of two generators and two discriminators. Also, what is newly added compared to typical GAN models is cycle-consistency loss and identity loss. Let's deep dive into each component.

- Generators

CycleGAN consists of two generators. The generators play a role as the domain changer, which generates the domain X from domain Y or in the opposite direction.

Generator consisted of encoder and decoder. Encoder was used to extract meaningful information from the original image. We implemented the encoders using the CBR2d class, composed of Reflection Pad – Conv2d – InstanceNorm(IN) – Relu. We constructed the encoders with the three CBR2d classes. Furthermore, we added some resnet blocks. ResBlocks are nothing but the series of CBR2d with concatenating input and output. We set the num_resblock parameters with 9.

After the encoder, we placed a decoder to reconstruct the image from the output of the encoder. We

composed a decoder with the 2 DCBR2d classes, each class having Conv2d_transpose-IN-Relu. To calculate faster, we added a bottleneck layer with 1 CBR2d layer and a tanh layer.

- Discriminators

To deal with two generators, we needed two discriminators. We implemented the discriminators for each generator. The structure of the discriminators were simpler than generators. We used only CBR2d classes. We constructed discriminators with 5 CBR2d to extract the important features. Then, we added a sigmoid function to discern whether the generated image is real or not. Besides, We initialized the weight parameters using Kaiming He's method and adopted the PatchGAN model for 70*70.

- Loss

After we constructed two generators and discriminators, we made a loss object. In cycleGAN, there are three types of loss objects such as Cycle consistency loss, Adversarial loss, Identity loss.

1) Adversarial Loss

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))],$$

Figure 11. formula of Adversarial loss. G denotes generator and D denotes discriminator. Small Y notation means that the discriminator tries to discriminate between the fake Y and the real Y.

We applied the typical GAN adversarial loss. G has a goal to deceive the D while D tries to catch a fake image. To train two generators, we have to implement two adversarial losses consisting of G_loss and D_loss. In fact, in the perspective of G, the losses should be minimized. On the other hand, in the perspective of D, the objective function should be maximized. To measure the adversarial losses, we used the Binary Cross Entropy function.

2) Cycle Consistency Loss

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].$$

Figure 12. formula of Cycle Consistency Loss.

When we used adversarial loss alone, it was hard to make the fake images similar to the original images. Then, we imported another loss which can ensure that the output image(fake image) was from the input image(real image). That is Cycle Consistency Loss. Its idea is to make output with a generator and input this output into another generator doing the opposite operation. then compare this new output with original images. Because we have two domains, we need to check the bidirectional way. That is why in the formula we have two terms. To measure the difference between original image and derived images, we used L1 Norm.

3) Identity Loss

$$\mathcal{L}_{identity}(G, F) = \mathbb{E}_{y \sim p_{data}(y)}[\|G(y) - y\|_1] + \mathbb{E}_{x \sim p_{data}(x)}[\|F(x) - x\|_1]$$

Figure 13. formula of Identity Loss

The idea of Identity loss is similar to the back translation method in natural language processing. If we input the x to G(y->x), then x must not be changed. It ensures that the generator does not affect the images in a domain other than the target domain. To gauge the loss, we also used L1 Norm.

These are three types of losses used for optimizing cycleGAN. From the cycleGAN paper, we brought in learning rate parameters and lambda weight for cycle consistency loss. We set 0.0002 for learning rate and 100 epochs for training. Also, we applied 0.1 weight for cycle loss and 5e-1 for identity loss.

4. Results

1) metrics

There are numerous metrics for measuring the accuracy according to images. We did not have any answer label for input image because cycleGAN uses unpaired data to train the model. It is the reason why we could not use measurements such as pixel distance and distribution similarity. As a result, one of the metrics we chose was the Human eYe Perceptual Evaluation(HYPE). To calculate the accuracy embracing the quality of output images, we came up with the four criteria. We gave 1 point for each criteria. If the target image meets all the requirements, it gets 4 scores. The higher the score is, The better the image is. By the HYPE, we could check the adequacy of glasses and overall quality of the images.

1. the time to distinguish whether the image is fake or real.
2. full form of glasses
3. keep the original tone(face color)
4. keep the entire shape of image(nothing cut)

2) interpretation

In common, all models did not distort any face shape or entire image shape. Besides, they were specialized in processing the square images rather than the rectangular images. On top of that, all models could not predict the blurred or low resolution images. They represented lots of differences in the level of completion and capacity to deal with various races.

First, the model with the kaggle dataset operated well only for the white, not for the other ethnic groups. Also, it could not predict the blurred image. However, few glasses seemed artificial, which means that it successfully worked for the white people.

Second, the model with the aihub dataset was terrible. It was overly fitted to the aihub dataset. When we tried the different types of images, the overall tone of the images became brighter than before. Even some of the images appeared to be animated.

Lastly, the model with the combined dataset works well for diverse ethnic groups. It could put glasses on all races except extremely black people. It basically didn't work well because we did not have enough black race images in our datasets.

3) HYPE of each image (images of ours not included)

			kaggle	aihub	all
		ID	total	total	total
black	men	1	**4**	1	**4**
black	men	2	2	1	**3**
black	men	3	1	1	**3**
black	men	4	**4**	1	3
black	men	5	3	1	**4**
black	women	1	2	1	**3**
black	women	2	**3**	1	**3**
black	women	3	2	2	**3**
black	women	4	**3**	1	**3**
black	women	5	**4**	1	**4**
white	men	1	2	1	**4**
white	men	2	2	3	**4**
white	men	3	**3**	1	**3**
white	men	4	**4**	1	**4**
white	men	5	**3**	1	**3**
white	women	1	2	1	**3**
white	women	2	**2**	1	**2**
white	women	3	3	1	**4**
white	women	4	**4**	1	**4**
white	women	5	**4**	1	**4**
yellow	men	1	2	1	**4**
yellow	men	2	2	1	**3**
yellow	men	3	1	**4**	**4**
yellow	men	4	1	**4**	**4**
yellow	men	5	1	**4**	**4**
yellow	women	1	2	**3**	**3**
yellow	women	2	1	**4**	3
yellow	women	3	1	**4**	**4**
yellow	women	4	1	**4**	**4**
yellow	women	5	1	**4**	**4**
		total	2.3	1.9	**3.5**

4) outputs

- **kaggle**

Figure 14. images generated by cycleGAN with kaggle dataset(left : real, right : fake)

- **aihub**

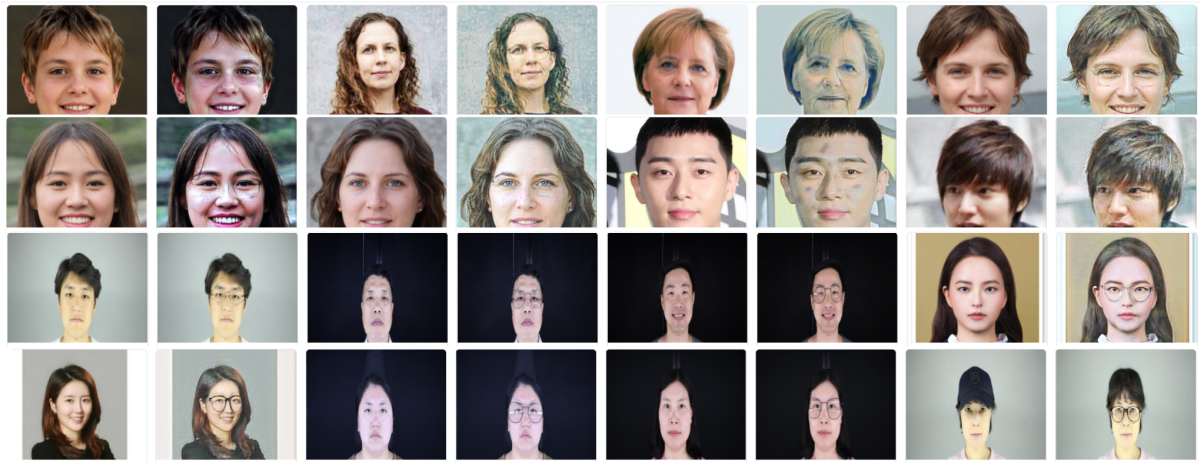

Figure 15. images generated by cycleGAN with ai hub dataset(left : real, right : fake)

- **all (kaggle+aihub)**

Figure 16. images generated by cycleGAN with all dataset(left : real, right : fake)

5. Applications

The glass generator can have a tremendous impact on the glasses industry. For users who hope to find the best fitting glasses, this application can help them choose the glasses. The sellers of the glasses can establish a recommendation system based on our application, which will increase the profits by suggesting the optimal glasses for the customers. In addition, we do not need complicated graphic editor programs like photoshop to add or remove glasses from our pictures. By simply putting their pictures into the AI applications, they can change their faces immediately without any effort. We highly recommend the developers of camera applications to use our application. By including our application in camera applications such as B612 and SNOW, we can provide convenient face editing services related with glasses.

Furthermore, we can expand the function of applications from adding any styles of glasses to changing a particular style to another style of glasses by using multi-domain models such as StarGAN. By assigning various styles of glasses to multiple domains, we can control the style of the glasses, not generating a randomly selected style of glasses. For the project, we must classify the glasses into several groups, which would be our further task.

6. Limitations

1) limitation of cycleGAN

First, we can only get one recommendation per person. That is, it is impossible to make various kinds of glasses for each person. We noticed that the model cannot generate various images because the weight of generators is fixed already once the training is over. Besides, the kind of glasses were somewhat limited.

Second, its performance depended on the resolution of the image. Specifically, if we transferred an image in domain X to domain Y, not only the resolution of the image became lower but also the tone of image became slightly changed. In our dataset, the color of face images accounted for the most part of the input image. As a result, the hue of images became closer to the color of the faces as a whole. For instance, the background became brighter by faces. Conversely, faces became darker because of the background.

Third, when the model generated the faces with glasses, the shape and color of eyes changed according to the glasses. Too far, some people seemed like different people in the output images. It may reflect the actual effect of glasses including refraction.

Finally, there were some biases in samples. Our model showed comparatively low performance in generating Asians and men with glasses. It worked well for western people and women. We guessed that the problem was caused by biased numbers in sex and races in our datasets.

2) Trials to overcome limitations

a) changing model

To recommend the various glasses, we realized that we need to add some style layer which recombines the style and randomly generated. But we didn’t have enough time to implement the new model and validate it. Instead, we tried to find the model using style code or styleGAN method online. Then we discovered StarGANV2 developed by Naver Cloba AI Lab. It used the reference image to generate new images. By using reference images, we could add specific styles of glasses to the faces!

STARGANV2 makes it possible with mapping network and style encoder. StarGANV2 doesn't extract image features naively. Instead, it derives the style code from randomly generated latent vectors and it maps them with images. In addition, the model trains the style encoder itself only to generate proper style code. Likewise, this model generates multiple images for one image. Also, in the perspective of loss, the model contains style reconstruction loss and diversification loss to make the image diversified. As a result, the model could be applied to multiple domains. Specifically, if we classify the glasses into different styles such as thick glasses or thin glasses, we could achieve our goal.

However, when we were about to adopt and implement, we suddenly faced a big problem. The model was so huge that our devices could not deal with many parameters and out of memory issues happened. Therefore, we will hopefully try STARGAN in the future.

b) including various kinds of images

What we have done to improve the performance for Asian men is to collect more datasets. This is why we used various sources of images such as google images and ai hub. Because most models trained with datasets from western countries are optimized for their country, we tried to collect Asian face images as much as possible.

7. Discussion

a) hyperparameter tuning

In our cycleGAN implementation, we could not apply recent deep learning techniques. We had a resource limitation in training the model. In fact, the available gpu resources were 2070 super or collab. That's why we could not find the best parameter for the model. When we trained the model with 10000 images, it took about 1 day and several hours. Also, we were worried about memory overflow. If we had a better GPU, we would apply the cycle_fit method or any lambda parameter decay for cycle loss and identity loss. We could add or reduce the blocks to find the best structure for our tasks. Moreover, we could train more epochs or increase the batch sizes. It was sad that we could not try many hyperparameters because of limited resources.

b) resolution of images

Often, there were no higher resolution images online. For instance, celebA and google images had low resolution. Whenever we attempted to get high quality images, most companies required us to pay a fee to use them. If we are able to collect images meeting higher standards, we will earn a more accurate model.

8. Reference

1. arXiv:1703.10593
2. arXiv:1912.01865
3. hanyoseob/pytorch-CycleGAN: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks
4. https://junyanz.github.io/CycleGAN/
5. https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix
6. https://phillipi.github.io/pix2pix/
7. Large-scale CelebFaces Attributes (CelebA) Dataset
8. https://aihub.or.kr/
9. https://velog.io/@tobigs-gm1/Style-GAN
10. https://fastcampus.co.kr/
11. http://www.kwangsiklee.com/2018/03/cyclegan%EC%9D%B4-%EB%AC%B4%EC%97%87%EC%9D%B8%EC%A7%80-%EC%95%8C%EC%95%84%EB%B3%B4%EC%9E%90/