Final Project Report for CS 175, Spring 2018
Project Title: SketchGAN
Project Team: 2
Student Name(s):
Shengquan Ni, 46564157, shengqun@uci.edu
Yangguang He, 50947171, yangguah@uci.edu
Changchuan Shen, 83371717, changchs@uci.edu

1. Introduction and Problem Statement

In this project, we use the Generative Adversarial Networks to complete a style transfer task between pictures and sketches using the CUHK student data set, with evaluation using classification accuracy and user studies. The input of the picture is not limited. It can be arbitrary, such as a landscape photo, portrait etc. Our goal is to transfer the picture into a sketch which contains almost all the details of the original picture. It is a style transfer task. People used to use filtering and other image processing techniques. (e.g. "XDoG: Advanced Image Stylization with extended Difference-of-Gaussians" [2]) or VGG neural network. Recently, people have invented the Generative Adversarial Networks, which contains a "Generator" and a "Discriminator". The "Generator" aims for generating "fake" samples and fooling the "Discriminator". And the job of the "Discriminator" is to recognize those "fake" samples as accurate as possible. We heard that GAN is useful in generating images. Thus, we wonder if GAN can do the same style transfer task.

After testing with several architectures, we found GAN did well in the style transfer task after training for 300-400 epochs with our dataset, and its result was similar to the result of VGG neural network but loses some details.

2. Related Work:

In the past people use filtering to achieve the goal. The standard Difference-of-Gaussians edge detection operator is one of the techniques, it increases the accuracy of sketch depiction and it is able to withstand the noise. The DoG operator supports many different styles such as pencil shading, pastel and etc [2].

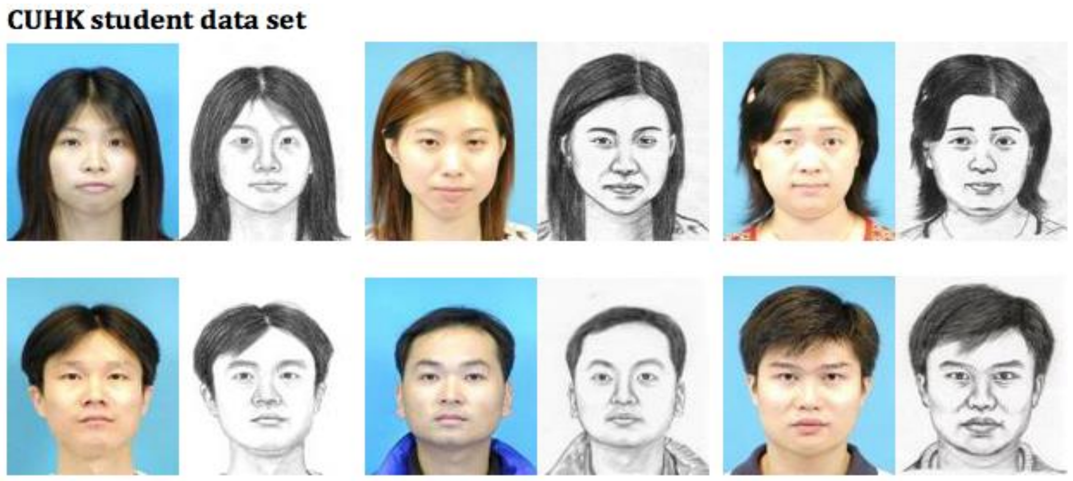
People also use a fully convolutional neural network (FCNN) to first create the outlines of faces and some key facial features such as eyes and mouth and then they proposed to use pyramid column feature to add textures and shadings [1]. Also, the quantitative and qualitative evaluations they used promises the framework works better than other artistic method given on different images as input.

Researchers also have tried some specific GAN structure to image style transfer. CycleGAN is one of them. It learns two mapping, one is from sample space X to Y, the other is from Y to X, at the same time. The author introduces "Cycle consistency loss", which prevents mode collapse [3]. Our approach is similar to CycleGAN but we modified the loss, the architecture and the way of training it.

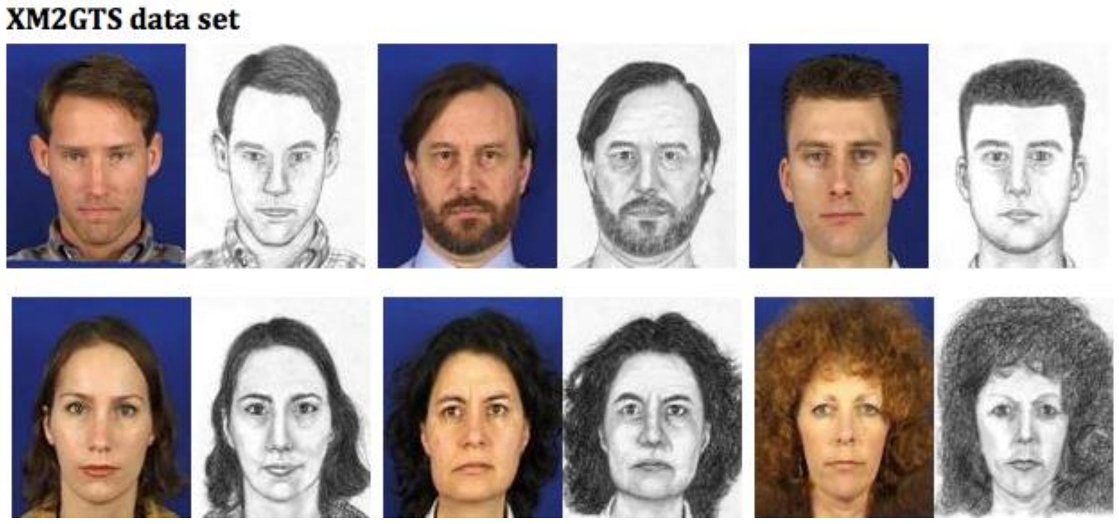
3. Data Sets

We used CUHK Face Sketch database (CUFS) for the face sketch recognition. This database has 188 faces from the Chinese University of Hong Kong, 123 faces from the AR database and 295 faces from the XM2VTS database. Each face has a sketch drawn by an artist based on a photo taken in a frontal pose, under normal lighting condition, and with a neutral

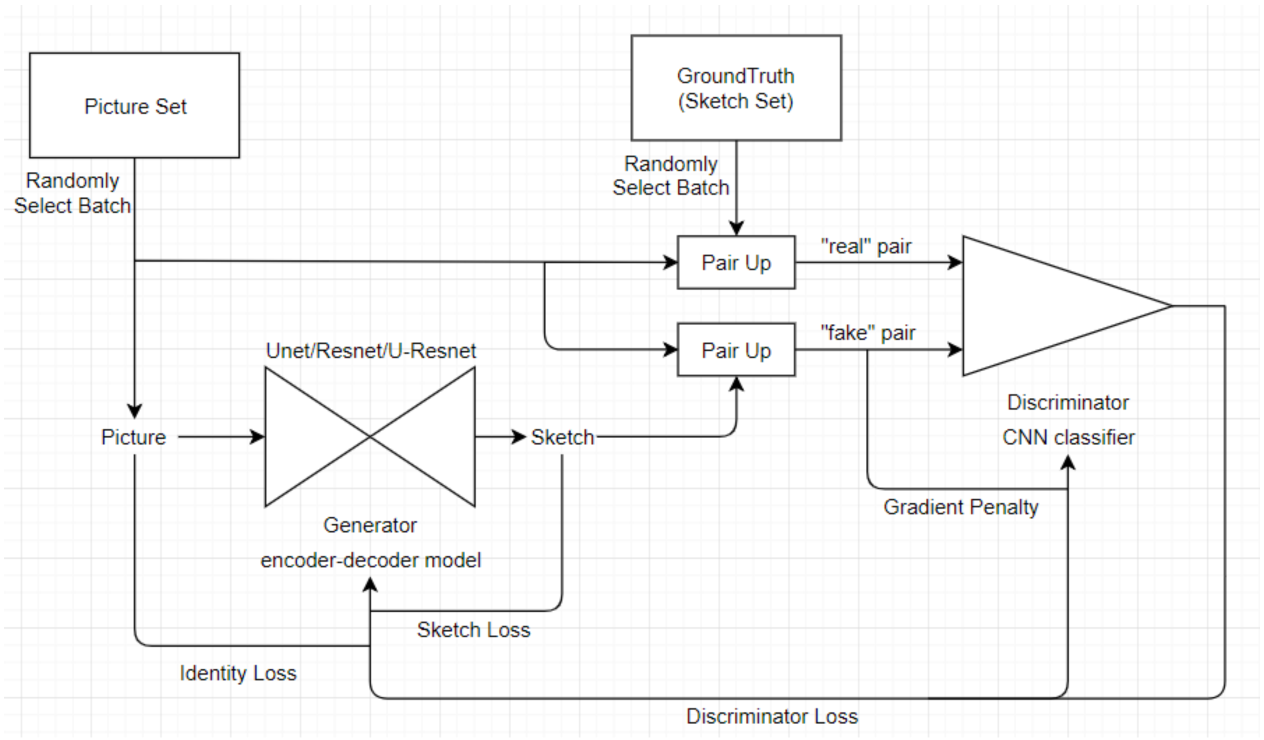
expression. Each one of the 3 databases contain uncropped and cropped photos, uncropped and cropped sketches and fiducial points of photos and sketches. Since we were doing style transfer task, we didn't need fiducial points. We chose cropped photos and sketches from CUHK and XM2VTS database because AR database was not available. These images all had 3 channels (for RGB) and the same resolution, which is 200x250.
Here are 6 samples from CUHK student dataset:



Here are 6 samples from XM2GTS dataset:



4. Description of Technical Approach
Architecture of SketchGAN:



GAN has two major component, the Generator and the Discriminator. The Generator is supposed to generate “fake” data and fool the Discriminator, but the Discriminator needs to classify the “real” and “fake” data as accurate as possible. This process called adversarial learning, that why this network called Generative Adversarial Networks. The original GAN uses Sigmoid in the last layer of the Discriminator to shrink the result between 0 and 1 because the task for the Discriminator is a binary classification. And original GAN uses log loss function. However, the original architecture makes GAN unstable. We must train the Discriminator carefully, the more accurate Discriminator gets, the more serious the gradients vanish. That’s make training original GAN a difficult problem. Thus, what we are using in our project is one improved type of Generative Adversarial Networks(GAN) [5], which called Wasserstein GAN with Gradient Penalty(WGAN-GP) [4]. WGAN-GP removes the Sigmoid layer from the Discriminator and the log function from the loss, which make the binary classification problem become a regression problem. In addition, WGAN-GP uses Wasserstein metric instead of JS divergence or KL divergence, which can be derived from the loss of the original GAN. The reason is that both JS and KL divergence are not a reasonable distance when measuring the difference between “real” and “fake” distributions, they will cause gradients unstable and mode collapse. In most cases, especially the early stage of training, the “real” and “fake” distribution has little or no intersection, so both divergence will make no sense. However, Wasserstein metric can reflect the distance between two distributions even they have no intersection. In the WGAN-GP paper, we need to satisfied Lipschitz continuity to calculate the Wasserstein metric. So, we calculate the gradient penalty between “real” and “fake” sample and add it to our loss function to make it satisfied Lipschitz continuity. We also introduced “Sketch Loss” in the loss of our Generator. “Sketch Loss” is the L1 loss between the generated image and the edges of the original image. We believe that a good sketch must contain all the edge information of the original image, so we add it into our Generator loss function. In addition, we add "Identity Mapping Loss" [3], which is introduced in CycleGAN paper. It is the L1 loss between the generated image and the original image itself. Thus, the final loss functions are like:

$L_{Discriminator} = \lambda GradientPenalty + Discriminator(fake) - Discriminator(real)$

$L_{Generator} = -Discriminator(fake) + \lambda SketchLoss + \lambda IdentityMappingLoss$

The architecture of our Discriminator is the same as CycleGAN, just 4 layers of LeakyReLU-Conv-BatchNorm block. However, we have 3 choices of our Generator, they are Unet, Resnet and U-Resnet. We try the same architecture of Unet and Resnet in CycleGAN, and both work very well. We also implemented a U-Renset, which is a mix of Unet and Resnet, it has Unet architecture in surface layers and successive residual blocks in the deepest layer. After doing some experiments, we find it works even better than Unet and Resnet. We also try to replace the ReLUs with ELU, which can speed up learning. However, we find the learning speed is not changing so much as the paper described [6].

5. Software

code we wrote:

U-Resnet Generator, an encoder-decoder model which takes an image and output a "fake" sketch

The training process for WGAN-GP, including calculating gradient penalty and loss, printing result for every k epochs.

Tool functions, including showing images, edge detections, saving/loading models…

Preprocessing functions

code for testing models

code from other people that we used:

Unet Generator

Resnet Generator

N-Layer Discriminator

Residual Block

Third-party Libraries:

pytorch

torchvision

matplotlib.pyplot

numpy

OpenCV

PIL

CUDA

6. Experiments and Evaluation

We have trained our SketchGAN using 3 different architectures of the Generator with Wasserstein matric as the measurement of the training process.

We used learning rate=1e-5 for both Discriminator and Generator

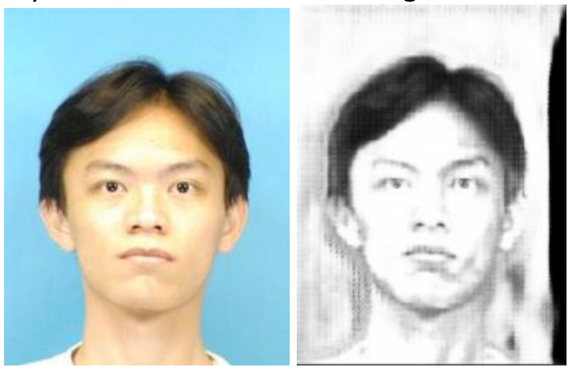
We used λ=0.5 for both loss functions.

We used batchsize=4

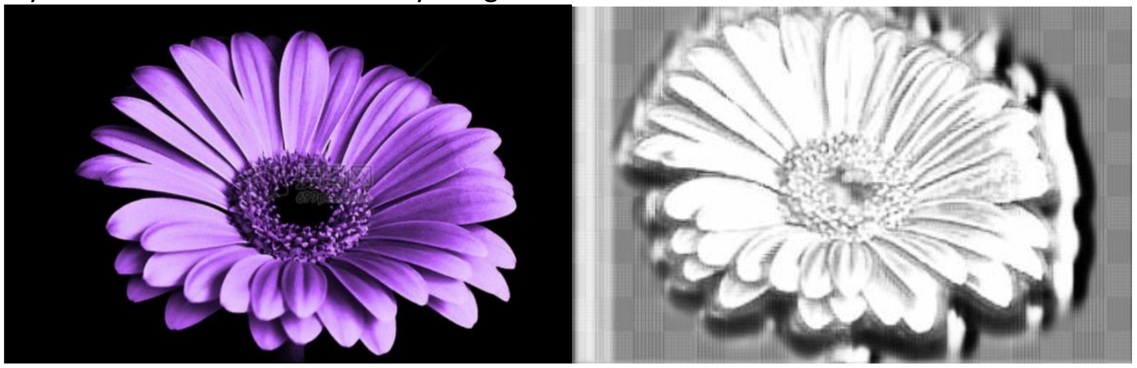
After training 300 epochs on the Resnet Generator, we got:

Wasserstein matric=0.1044-0.0884=0.016, which means the distribution of "fake" and

“real” samples are very close.
Generator Loss fluctuated between -0.1 and -0.25, we believed the model is already optimized (reached its maximum accuracy and cannot be better)
style transformation on training dataset:

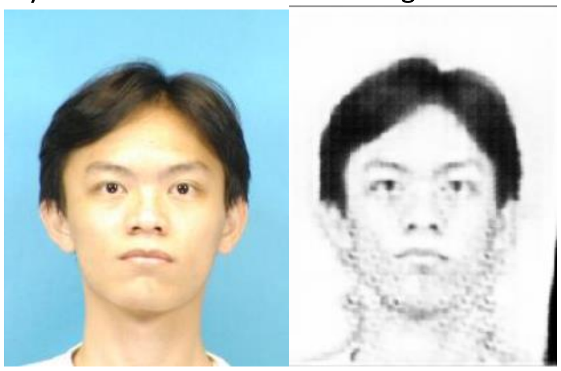


style transformation on arbitrary image:

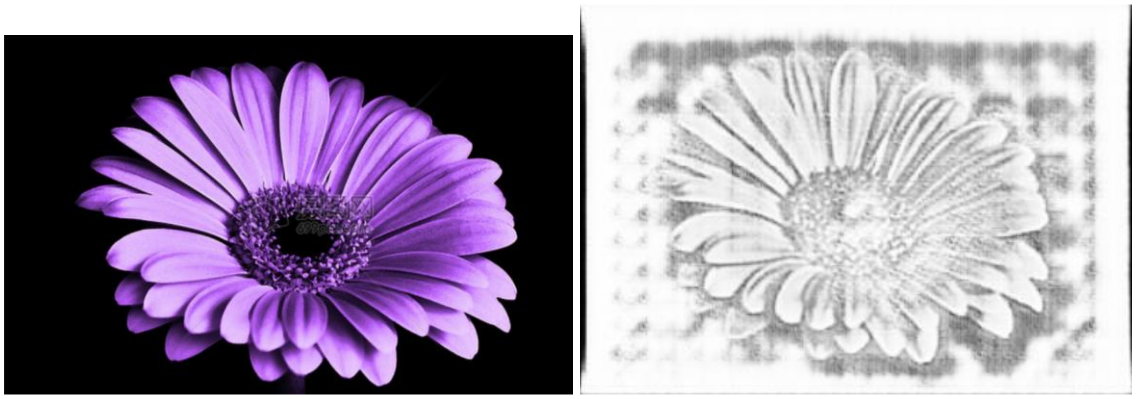


After training 300 epochs on the Unet Generator, we got:

Wasserstein matric=-0.0899-(-0.1023)=0.0124, which means the distribution of “fake” and “real” samples are very close.
Generator Loss fluctuated between -0.12 and -0.16, we believed the model is already optimized (reached its maximum accuracy and cannot be better)
style transformation on training dataset:



style transformation on arbitrary image:

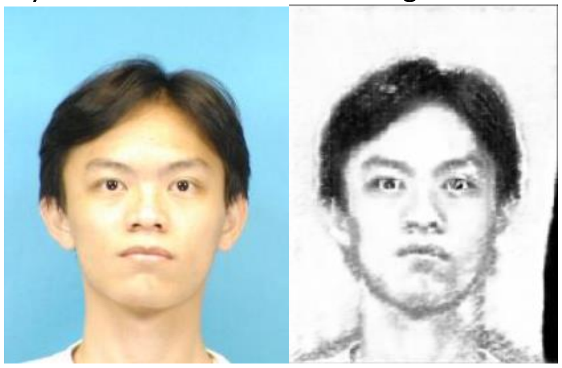


After training 300 epochs on the U-Resnet Generator, we got:

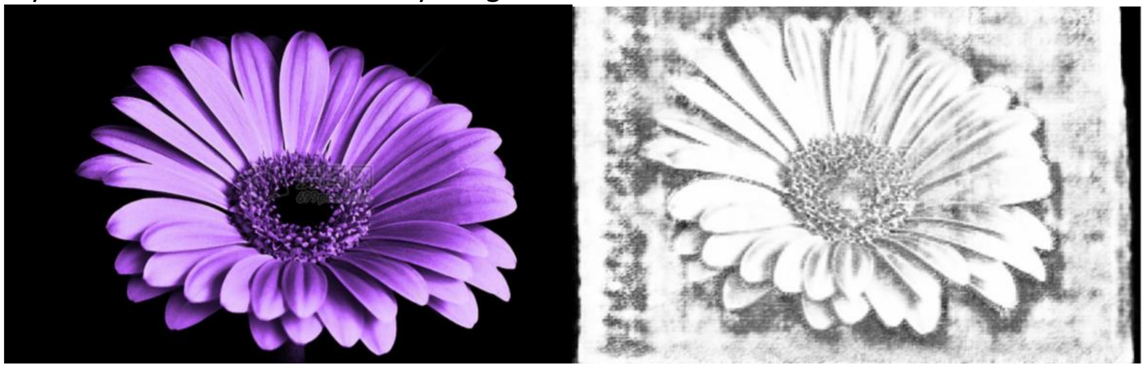
Wasserstein matric=-0.0881-(-0.0921)=0.004, which means the distribution of “fake” and “real” samples are very close.

Generator Loss fluctuated between -0.23 and -0.35, we believed the model is already optimized (reached its maximum accuracy and cannot be better)

style transformation on training dataset:



style transformation on arbitrary image:



7. Discussion and Conclusion

In this project, we learned the GAN in depth and make use of it. It is very useful for generating images. The training process of GAN is different from what we learned in any ML classes, it is Unsupervised. We also know how to build neural networks from scratch. We read a lot research papers and understand different types of GANs. The result is agreed our expectations and we believe our Sketch GAN can do the style transfer task between images and sketches, but we are still wondering why the generated images are blurred. One possible reason we come up with is that our loss functions are not good enough. We decide to continue our research on the problem, developing our loss functions and models (maybe add some architecture of super-resolution GAN [7]).

References

[1] C. Chen, X. Tan and K. Y. K. Wong, "Face Sketch Synthesis with Style Transfer Using Pyramid Column Feature," *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Lake Tahoe, NV, 2018, pp. 485-493.

[2] Winnemoller, H., Kyprianidis, J., & Olsen, S. (2012). Xdog: An extended difference-of-gaussians compendium including advanced image stylization. *Computers & Graphics*, *36*(6), 740–753.

[3] Jun-Yan Zhu*, Taesung Park*, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", in IEEE International Conference on Computer Vision (ICCV), 2017.

[4] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville. “Improved Training of Wasserstein GANs”, arXiv:1704.00028v3

[5] Goodfellow, Ian; Pouget-Abadie, Jean; Mirza, Mehdi; Xu, Bing; Warde-Farley, David; Ozair, Sherjil; Courville, Aaron; Bengio, Joshua (2014). "Generative Adversarial Networks". arXiv:1406.2661

[6] Djork-Arne Clevert, Thomas Unterthiner & Sepp Hochreiter “Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)”. arXiv:1511.07289v5

[7] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, ´ Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi Twitter. “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”. arXiv:1609.04802v5