CS230

Avatar Artist Using GAN

Hui Su, Jin Fang
Stanford University
{huisu, jinf9812}-@stanford.edu

Figure 1

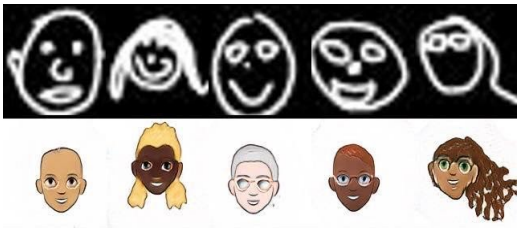

Figure 2

(a) human quickly draw sketches of human faces (b) our algorithms learn this sketch to anime transformation and automatically generate to two different types of anime Figure 1 and Figure 2

Abstract

human sketches would be expressive and abstract at the same time. Generating anime avatars from simple or even bad face drawing is an interesting area. Lots of related work has been done such as auto-coloring sketches to anime or transforming real photos to anime. However, there aren't many interesting works yet to show how to generate anime avatars from just some simple drawing input. In this project, we propose using GAN to generate anime avatars from sketches.

1 Introduction

Human stick figures are always very intuitive and abstract. We found solving the problem of transforming human face sketches into anime avatars would be an interesting area. One application could be developing a kid friendly app that records and converts a kid's hand draw sketches into anime or cartoon images. In this project, the input to our algorithm is an image of a human face sketch, and the output would be an anime picture of the face. We use GAN as our algorithms to generate the output. The ultimate goal is that output avatars should look similar with input features and meanwhile look authentic.

2 Related work

There are existing similar great applications. From the perspective of functionality, there are applications such as coloring sketches to anime such as [1][2], transforming real photos to sketches such as [3][4], and transforming sketches to real photos such as [6] and transforming styles between magato-anime such as [5].

From the perspective of approaches, most of the work mentioned above use GAN. [1] uses their noval AC-GAN [3]so the algorithm is applicable to different specific art styles and [2] focuses on a 2-stage architecture to make the coloring more natural-looking. [3] proposed noval APDrawingGAN to optimize for the loss function when transfering from real-photos to sketches. [4] uses Conditional GAN. [5][6] use CycleGAN for their applications.

In general, [1] and [3] are using more advanced algorithms by either improving on the architecture of loss function. While [4][5][6] uses more basic versions of GANs to tackle more specific problems but also achieve reasonably good results.

By comparing all existing relative work, we have not found any application that is specifically on our problem, which is to transform human drawing sketches to anime. In this project, we mainly adopted CycleGAN [7] as the final result with exploration of DiscoGAN[8] and XGAN[9]. We used both paired and unpaired data to learn the transformation, which is explained in the next section in detail.

3 Dataset and Features

In the experiments, Anime sketch data and Quick, Draw! data [10] are used as the input, which are human face sketches. Danbooru dataset[11] and Cartoon Set [12] are used as output, which are anime domain data. They are the expected output avatar domain styles.

2.1 Danbooru dataset 2019

We used Danbooru dataset 2019 [11] as the starting point for anime domain images, which contains over 3 million anime images with 512px. And we extracted 50k avatar images with 96px as our experiments inputs, see examples in Figure 3. We split 47k of them as the training set and another 2k as the test set. We found this dataset is lacking some of the diversity as many of the figures have similar eyes, nose, and hairstyle.

2.2 Danbooru Sketch Avatar dataset 2019

For sketch avatar data, we wrote edge-detection algorithms to generate the corresponding sketch version of the same from avatar images [10], see examples in Figure 4. As a result, the datasets from anime avatar domain and sketch domain are pair-wised.

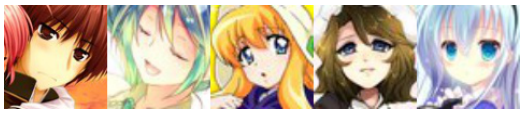

Figure 3

Figure 4

2.3 Quick, Draw! Dataset

We used Quick Draw dataset [10] with resolution of 28*28 px from online real human sketches。 Examples are demonstrated in Figure 5. We used 7k of quick draw data for training and 2k of them as testing. The dataset is closer to human hand drawing sketches.

2.4 Cartoon Set

Since the gap between anime avatar and Quick, Draw! Data is big. Humans even feel hard to imagine mapping from human hand drawing to anime. We also introduced another type of avatar image - Cartoon set[12], which is more diverse in terms of face size, facial features such as shape of eyes, nose and hairstyle, etc as shown in Figure 6. A total number of 10k images with resolution of 128px are used in our experiment. We used 1.5k of them as testing data, and 8.5k of them as training.

Figure 5

Figure 6

4 Methods

We think GAN is our direction to tackle this project since it is one of the most popular image generation techniques. We have explored the options of CycleGAN, DiscoGAN, and Unit. Based on the results, we chose CycleGAN. But we will talk about all of them briefly in the following section.

4.1 CycleGAN

We train the CycleGAN using the implementation of [7] to generate the mappings from anime avatar to sketch and also sketch to anime avatar. Cycle GAN can learn the mapping functions between two domains X and Y given training sets X and Y.

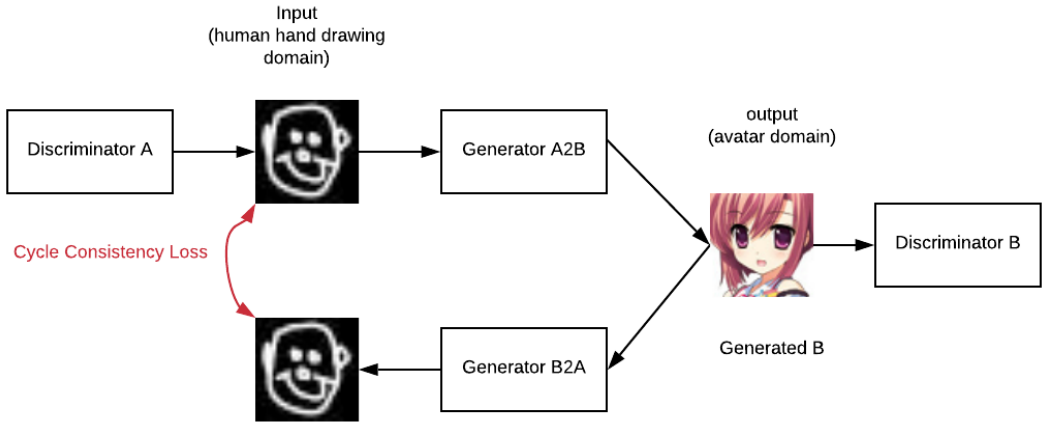

Figure 7

In the CycleGAN, we have the Adversarial Loss from X to Y domain. We also have the same loss from another direction.

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

We also use Cycle Consistency Loss to implies that generators should be able to bring x or y back to the original input,

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

Then we can get the full loss function,

$$L(G, F, D_X, D_Y) = L_{GAN}(G, D_Y, X, Y) + L_{GAN}(F, D_X, Y, X) + \lambda L_{cyc}(G, F)$$

The optimization goal is,

$$G^*, F^* = \min_{G,F} \max_{D_X, D_Y} L(G, F, D_X, D_Y)$$

We use the implementation provided by Erik Linder-Norén[15]. In this implementation, the generators are ResNet. PatchGAN is used in Discriminators.

4.2 UNIT

UNIT[9] is another common method for unsupervised condition generation. CycleGAN directly transforms images in domain A to domain B. UNIT projects both images in two domains to the

same common space. Add cycle consistency or semantic consistency to make sure images in both domains can convert back to origin.

4.3 DiscoGAN

DiscoGAN[8] is another method based on GAN that learns to discover relations between different domains with unpaired data. It successfully transfers style from one domain to another while preserving key attributes such as orientation and face identity. We found the loss function and architecture is very similar to CycleGAN except for generators.

5 Experiments/Results/Discussion

In our experiment, for CycleGAN we used hyperparameters of lr =0.0002 , adam_optimizer, n_residual_blocks=9, batch_size = 64

In DiscoGAN, we used lr =0.0002 , adam_optimizer, n_residual_blocks=9, batch_size = 64

In Unit, we used lr = 0.0002, batch_size = 64

We learnt from other work that setting lr=0.0002 is reasonably good for similar problems. And the batch size we chose is the largest we could fit in memory for our machines so we can fasten the training as much as possible.

5.1 Experiment 1 (pairwise, anime avatar and sketch data)

In the first experiment , we used anime avatars and sketch dataset as two different domains. We have run 20 epochs(about 14700 iteration) using cycleGAN. Since most sketch images are very detailed, the problem is more close to a coloring problem. We can get very good coloring results in the first several epochs. Figure 8 and Figure 9 are the outputs in 5th epoch and 20th epoch. In this experiment, we find it is relatively easy to learn the mapping relationship from sketch to anime. It is because the edge detection algorithm is a rule-based algorithm.

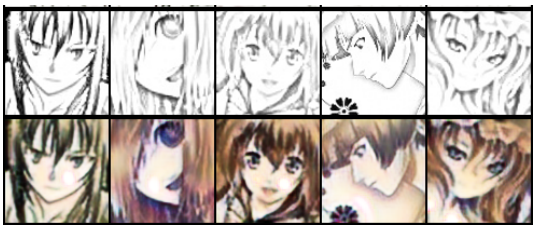

Figure 8

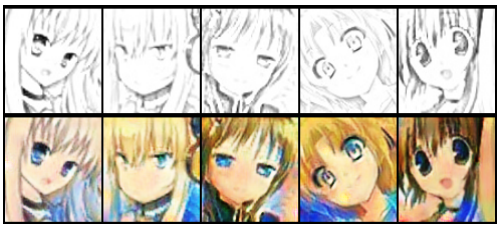

Figure 9

5.2 Experiment 2(unpairwise, anime avatar and quick draw face data)

In the second experiment, we use Quick, Draw! dataset as sketch style data and both Danbooru anime avatar and Cartoon Set as avatar style data. In this experiment the two style datasets are not pairwise. We run the experiment with CycleGAN, DiscoGAN, and UNIT. We find CycleGAN slightly outperforms the other two. In our experiment, we find the mode collapse problem is hard to avoid after hundreds of epochs in all approaches. We find 15-40 epochs are ideal in the experiment. We try to use the minibatch discriminator[14] and ensemble multiple models to partially solve mode collapse and mode dropping problems. Figure 10 and Figure 11 are the ensemble results of CycleGAN. In Figure 12, we plot the loss of the first 20 epochs in CycleGAN, the loss functions start to be flatten.

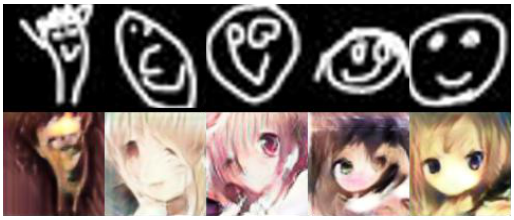

Figure 10

Figure 11

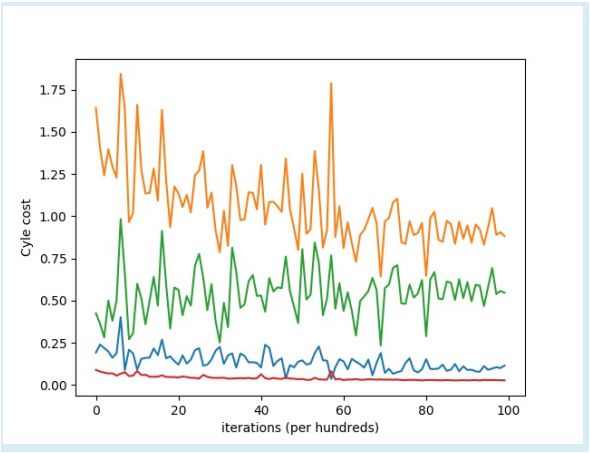

Figure 12(Orange - average Adversarial Loss, Green - cost of generator, Blue - cost of Discriminator, Red - cost of identity)

6 Conclusion/Future Work

We used CycleGAN, DiscoGAN, and Unit. CycleGAN and DiscoGAN are pretty similar in terms of loss function but their neutral network structures are different. CycleGAN uses ResNet and DiscoGAN uses U-Net. In general, we found CycleGAN slightly outperforms DiscoGAN with the same number of epochs. DiscoGAN is more vulnerable from suffering and stuck in mode collapse.

Also, Unit has similar performance compared to CycleGAN but in the early stage of learning, its learning speed is slower thus needed more epochs to achieve the same performance.

In terms of datasets, we found that the current Cycle GAN performans pretty well on Danbooru and its sketch dataset, but not optimal for quick draw dataset. Some of the causes might be

1) due to the simplicity of quick draw data, the facial features of eyes, nose, etc are underrepresented for most of the time. So even for human beings, it would be a hard task to image the transformation to cartoon.

2) In Cartoon Set, all mouths represented in the images are the same shape and size. So naturally, the output of our algorithm also has one type of mouth. It is hard for models to learn different facial expressions without exposure of diverse data distribution on the training data.

In both experiments, mode collapse/dropping is a very common issue we found when using GAN to generate domain B pictures. Each checkpoint of the model would generate very similar results on different input data. We addressed mode drop by two solutions one is using a minibatch discriminator. Instead of sending a single image to the discriminator, we send a batch of real images or fake images to the discriminator, to help the discriminator recognize the image distribution problem. Another is running the test by ensemble multiple generators.

7 Contributions

Hui and Jin, the two contributors to this project, almost separated the work equally along the project, including dataset preparation, model implementation, modeling, results analysis, etc.

Github link: https://github.com/diandiansu/anime-artist

References

[1] Zhang, Lvmin, et al. "Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan." *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*. IEEE, 2017.

[2] Chen, Yu-shun, "2 stage conditional gan for sketch auto-coloring". CS230. Stanford University 2018

[3] Yi, Ran, et al. "APDrawingGAN: Generating Artistic Portrait Drawings from Face Photos with Hierarchical GANs." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

[4] He, Yipeng, et al. "deep sketch drawer" CS230. Stanford University. 2018

[5] Griffin, Jonathan, et al. "Mega-to-anime translation using cycle-consistent generative adversarial networks". CS230. Stanford University. 2019

[6] Meng, Jerry, et al. "Sketch2Face: Using CycleGAN to Produce Photo-like Images from Unpaired Sketches". CS230. Stanford University. 2018

[7] Almahairi, Amjad, et al. "Augmented cyclegan: Learning many-to-many mappings from unpaired data." *arXiv preprint arXiv:1802.10151* (2018).

[8] Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

[9] Liu, Ming-Yu, Thomas Breuel, and Jan Kautz. "Unsupervised image-to-image translation networks." *Advances in neural information processing systems*. 2017.

[10] Danbooru Dataset: Retrieved from https://www.gwern.net/Danbooru2019

[11] Quick, Draw! Dataset: Retrieved from https://quickdraw.withgoogle.com/data/face

[12] Cartoon Set: Retrieved from https://google.github.io/cartoonset/index.html

[13] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." *Proceedings of the IEEE international conference on computer vision*. 2017.

[14] Salimans, Tim, et al. "Improved techniques for training gans." *Advances in neural information* processing systems. 2016.

[15] PyTorch-GAN https://github.com/eriklindernoren/PyTorch-GAN