Final Project Proposal Group 2
Anime Face Generator using GAN

Team Member: Chen Chen, Hao Ning, Hungchun Lin

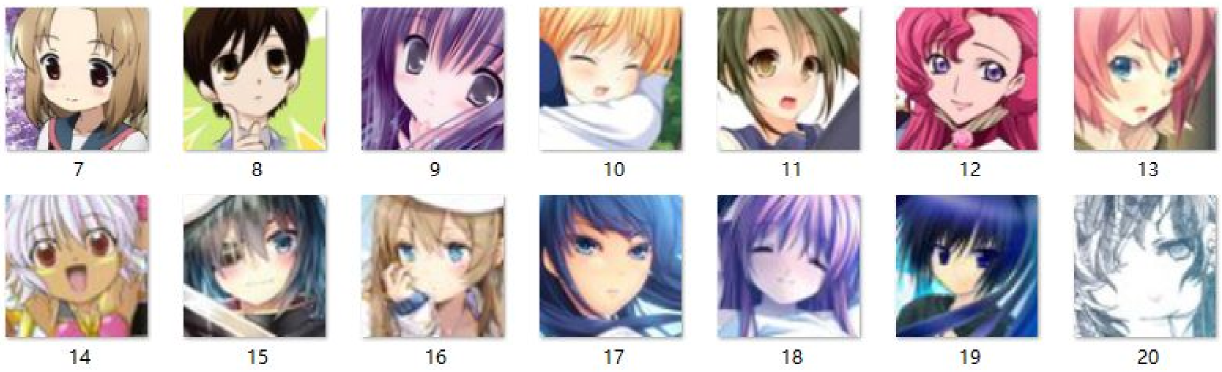
Description of the Problem

Generative Adversarial Network(GAN) is a hot topic in machine learning in recent years, it was invented by Ian Goodfellow and his colleagues in 2014[1]. It has many real life applications in Fashion, Art, Advertising, Science,Video games, etc[2]. Image generation is one of the amazing practices of GAN. In this project, a generator that produce various types of anime faces will be developed using Deep convolutional GAN, then the performance of the generator is optimized by Wasserstein GAN (WGAN) and the improved WGAN, which is the WGAN with Gradient Penalty (WGAN-GP) .

Data Source

There is an image archive and a tag archive in the original source. Image data and tag data are collected by students from NTU, downloaded from google drive, each consists of 33431 instances. There's extra_data from this google drive.

Animation images example:



All images are colored with RGB channels, and with the same size of 96x96. The main object of all the images is the animation portrait.

This project will not use tag archive. But tags may be used for further study of CGAN.

[1] Goodfellow, Ian, etc. (2014). *Generative Adversarial Networks*

[2] Generative adversarial network wikipedia, https://en.wikipedia.org/wiki/Generative_adversarial_network

Framework to Implement the Network

GAN has two key components: generator and discriminator, and both are neural networks. The generator trains the input images and generates the new images. Discriminator learns to evaluate the output images of the generator and assign high scores to images it defines as real images and low scores it defines as generated images. Generator always wants to improve the performance to “fool” the discriminator. When the generator improves, there must be a loss increase in discriminator. And the goal is to find the lowest aggregate loss of generator and discriminator.

Deep Network Selection

A generative adversarial network (GAN), is an architecture for training deep learning-based generative models. The architecture of GAN consists of a generator and a discriminator model. Based on the original GAN, the original GAN can be improved by changing the loss function. The Wasserstein GAN (WGAN), was introduced by Martin Arjovsky, et al., which is an extension of the original GAN that both improves the stability when training the model and proposes a new loss function that correlates with the quality of generated images. Based on the original WGAN, a paper "Improved Training of Wasserstein GANs" written by Ishaan Gulrajani, et al. proposed to expose penalty on the norm of weights from the critic network, which is the WGAN with gradient penalty(WGAN-GP) in this project.

Reference Materials and Background Support

Compared to traditional GAN, Wasserstein GAN approach can improve the learning stability, avoid problems such as mode collapse, and provide meaningful learning curves useful for debugging and hyperparameter searches[3]. WGAN suggests to use Wasserstein distance to evaluate the distance between real and generated samples instead of JS convergence in GAN. WGAN uses weight clipping to ‘fulfill’ the constraint requirement of 1-lipschitz function. However, the WGAN approach sometimes fails to converge and will only generate poor samples, which is usually caused by the use of weight clipping in WGAN. An improved WGAN is proposed by penalizing the norm of gradient of the critic with respect to its input for clipping weights (WGAN_GP)[4]. WGAN and WGAN_GP are modifying the loss function of GAN to get better results. There are also many other tips and tricks to train a GAN and make it perform better, such as avoid Sparse Gradients: ReLU, MaxPool, use a spherical Z etc[5]. Tuning different parameters will be applied in this project to get optimized performance.

[3] Wasserstein GAN. https://arxiv.org/abs/1701.07875

[4] Improved Training of Wasserstein GANs. https://arxiv.org/abs/1704.00028

[5] https://github.com/soumith/ganhacks

Evaluation

Since the project goal is to generate various anime faces, the evaluation of the results is straightforward. The most obvious way is just to compare the quality of the generated faces from different models. We will generate batches of images for every training epoch and track the optimization process. The number of good samples and bad samples will be compared, the yield rate will be our standard for evaluation.

Working Schedule

Check Point Date	**Milestone**
03/13/2020 - 03/27/2020	Topic Selection, Background Research
03/27/2020 - 04/03/2020	Project Proposal
04/04/2020 - 04/11/2020	Data Import and Pre-processing
04/12/2020 - 04/18/2020	Network and Framework Development
04/19/2020 - 04/25/2020	Finalize Training and Evaluation
04/26/2020 - 05/01/2020	Project Presentation and Report