Individual Final Project Report

HungChun Lin

Introduction

The knowledge of neural networks covered in Machine Learning II, for now, only has been utilized on supervised learning. This project will develop more in-depth knowledge and more application based on the knowledge already been covered in the Machine Learning II course. After searching about the different applications of deep learning, Generative Adversarial Networks (GAN) is an interesting field, which combined the multilayer perceptron (MLP) and convolutional neural network (CNN) that had been covered in the course, but it is an unsupervised learning method in deep learning, which our team wants to explore. Generating images is an amazing demonstration of Deep Learning, in this project, the images generating problem will be solved by Deep convolutional GAN, wasserstein GAN (WGAN) and the improved WGAN, which is the WGAN with Gradient Penalty (WGAN-GP).

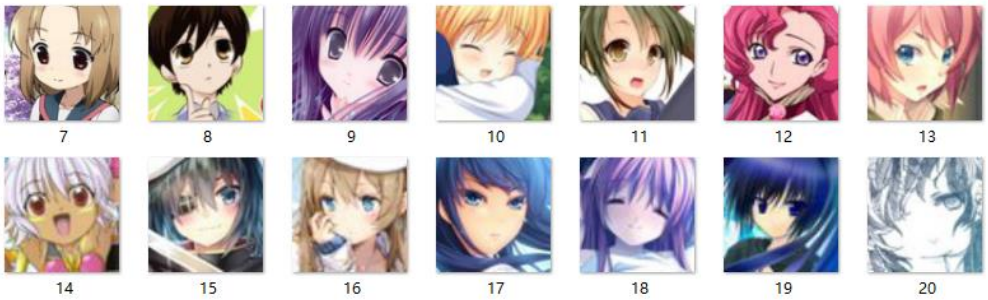
Background of GAN

Generative Adversarial Networks (GANs), which are a method for generative modeling based on deep learning, it was first described in the 2014 paper by Ian Goodfellow, et al. titled "Generative Adversarial Networks."Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data. From this approach, the model can be used to generate new examples that could have been drawn from the original dataset. GAN now is a hot topic for research, there have been many different types of GAN implementation and named with suffix plus GAN, such as DCGAN, LAPGAN, infoGAN and Cycle GAN, etc.

Description of the data

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.

Description of Individual work:

Main contribution:

Do the research on the algorithm and structure of DCGAN

Find out how the mode collapse happens with different epoch in DCGAN

Partial contribution:

Data collection

Data preprocessing

Research on WGAN and WGAN-GP model

DCGAN

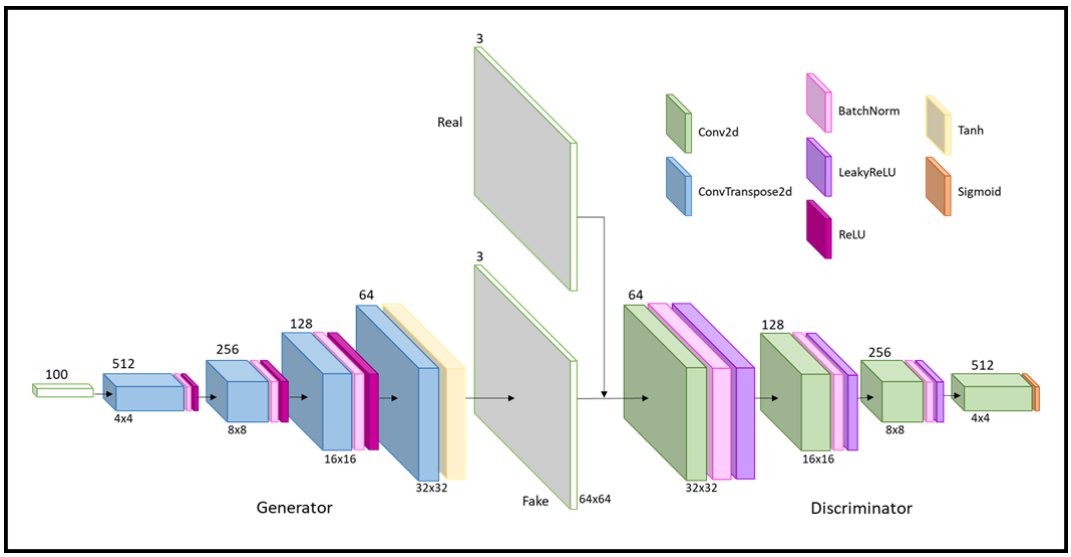
Algorithm

Deep Convolutional GAN (DCGAN) is one of the most popular GAN. It is composed of convolution networks in place of multi-layer perceptrons. DCGAN uses a standard CNN architecture on the discriminative model. For the generator, convolutions are replaced with upconvolutions, so the representation at each layer of the generator is actually successively larger, as it makes from a low-dimensional latent vector onto a high-dimensional image. The convolution networks implemented without max pooling, which is replaced by convolutional stride. And for the DCGAN, it utilized leaky ReLU for discriminator, because Leaky ReLU allows the pass of a small gradient signal for negative values, it makes the gradients from the discriminator flows stronger into the generator.

Math theory:

- x is original data, z is noise, θ_d is the weight of discriminator, θ_g is the weight of generator
    - Sample m examples $\{x^1, x^2, \ldots, x^m\}$ from database
    - Sample m noise samples $\{z^1, z^2, \ldots, z^m\}$ from a distribution
    - Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m\}$, $\tilde{x}^i = \mathrm{G}(z^i)$
    - Update discriminator parameters θ_d to maximize
        - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} logD(x^i) + \frac{1}{m}\sum_{i=1}^{m}(1 - logD\left(\mathrm{G}(z^i)\right))$
        - $\theta_d \leftarrow \theta_d + \eta\nabla\tilde{V}(\theta_d)$
    - Sample m noise samples $\{z^1, z^2, \ldots, z^m\}$ from a distribution
    - Update generator parameters θ_g to minimize
        - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} \log(1 - D\left(G(z^i)\right))$
        - $\theta_g \leftarrow \theta_g - \eta\nabla\tilde{V}(\theta_g)$

Model Setup



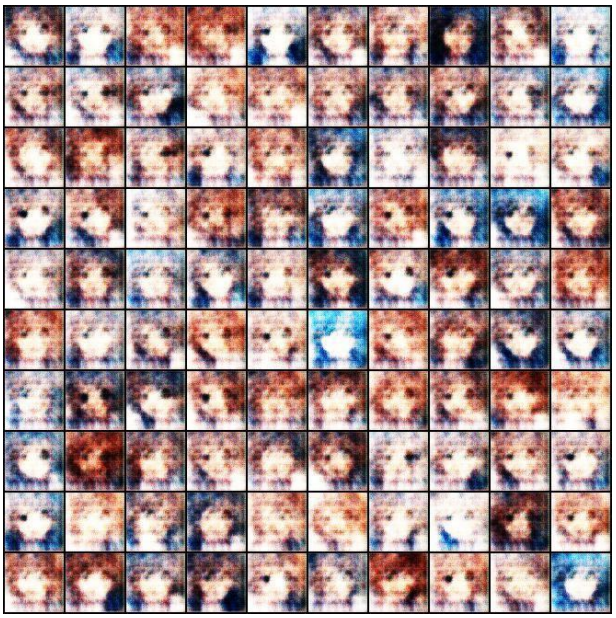
Generator Structure

Generator	Output size
Input noise vector: Z(100)	
Linear(100) + BN + ReLu	(512,4,4)
Conv2d_transpose(512, (5,5), stride = 2) + BN+ ReLu	(256,8,8)
Conv2d_transpose(256, (5,5), stride = 2) + BN+ ReLu	(128,16,16)
Conv2d_transpose(128, (5,5), stride = 2) + BN+ ReLu	(64,32,32)
Conv2d_transpose(64, (5,5), stride = 2) + Tanh	(3,64,64)

Discriminator Structure

Discriminator	Output size
Input: Image(64*64*3)	
Conv2d (3, (5,5), stride = 2) + BN + LeakyReLu	(64, 32, 32)
Conv2d (64, (5,5), stride = 2) + BN + LeakyReLu	(128, 16, 16)
Conv2d (128, (5,5), stride = 2) + BN + LeakyReLu	(256, 8, 8)
Conv (256, (5,5), stride = 2) + BN + LeakyReLu	(512, 4, 4)
Conv (512, (4,4), stride = 1) + Sigmoid	(1)

Model Performance



Left: DCGAN epoch 1

Right: DCGAN epoch 25

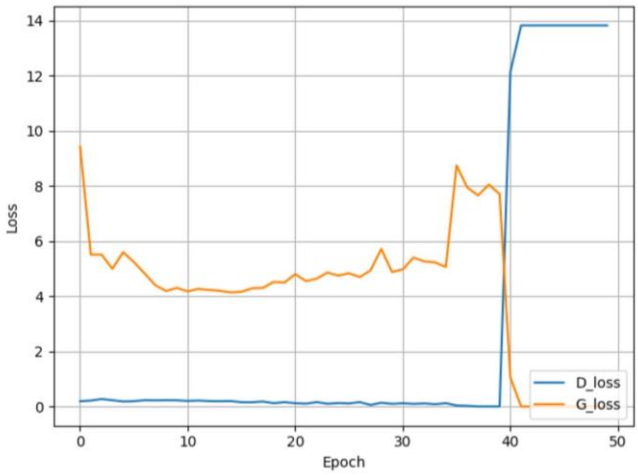
Left: DCGAN epoch 35 Mode Collapse)

Right: DCGAN epoch 50 (Mode Collapse)

The left image is the result of epoch 25, and the right image is the result of epoch 50 which is also an example of mode collapse.

Mode collapse

Mode collapse is when the generator generates a low-diversity of samples, or even the same sample, no matter how the different inputs, the output does not change. In real life, the data should be multimodal, but when the mode collapse happens, it only generates one mode of data. Mode collapse is a hard problem for training GAN, it may happen only partially or not at all. The reason for mode collapse is that a generated image converges to a x*, that fool the discriminator the most. In this situation, x* will be independent of the input.



As the loss history plot, we could find out that the loss of generator suddenly soar from the epoch 32, it is also the timing that mode collapse star happening, and around the epoch 40, the loss of G becomes zero and the loss of discriminator soar extremely.

For the data preprocessing, we did the horizontal flip for data augmentation without any other rotate since if we rotate the images, there will be some black part into the images, it would cause the training focusing on the black part and produce fail images.

DCGAN Summary

In this project, while the epoch went over 30 epoch, the mode collapse happens frequently. DCGAN is popular in the GAN technics, it is the basis of other further GAN. However, DCGAN has some problems, the first is the mode collapse we have introduced, and another problem is that the training for DCGAN is very unstable. If the discriminator gets stronger quickly, then the gradient of the loss function become zero, then the learning stopped, but when the discriminator gets too weakly, then the generator does not have good feedback from discriminator so the loss of generator is hard to improve.

Code

Less than 20% is from/copied from the internet.

Reference

1. Hongyi Lee, GAN 2018: https://www.youtube.com/watch?v=DQNNMiAP5lw&list=PLJV_el3uVTsMq6JEFPW35BCiOQTsoqwNw
2. DCGAN: https://github.com/carpedm20/DCGAN-tensorflow
3. GANHACKS: https://github.com/soumith/ganhacks
4. Wasserstein GAN: https://arxiv.org/abs/1701.07875
5. Improved Training of Wasserstein GANs: https://arxiv.org/abs/1704.00028
6. Machine Learning Mastery: https://machinelearningmastery.com/category/generative-adversarial-networks/