2019/11/07
0780828
Alisher Mukashev (穆安里)

HOMEWORK #2

Goals

Train your generative model using CELEBA dataset to generate images.

Introduction

In this homework, I am going to generate realistic looking faces. In order to do so, we are going to use Generative Adversarial Networks (GANs) [1], and more specifically Deep Convolutional Generative Adversarial Networks (DCGANs) [2]. The underlying idea behind GAN is that it contains two neural networks that compete against each other in a zero-sum game framework, i.e. generator and a discriminator.

The code based on Github repository [3].

Generator

The Generator takes random noise as an input and generates samples as an output. Its goal is to generate such samples that will fool the Discriminator to think that it is seeing real images while actually seeing fakes. We can think of the Generator as a counterfeit.

Discriminator

Discriminator takes both real images from the input dataset and fake images from the Generator and outputs a verdict whether a given image is legit or not. We can think of the Discriminator as a policeman trying to catch the bad guys while letting the good guys free.

Methodology

Preprocessing

Because of the limits the resolution of the model, I resized the image from dataset into 112x112. Celeba dataset has unnecessary pixels in images that doesn't consist faces, hence before resizing I cropped the images.

Loss functions

Our training process consists of the "battle" between the discriminator and generator. They both strive to reduce their losses. If they are well-balanced they will both tend towards some convergence points. Our perfect final state would look like this:

1. Generated samples look good and reflect the input dataset.
2. Discriminator converges to 0.5 - 50% accuracy, discriminator does not know how to distinguish between real inputs and fake ones.
3. Generator converges to 1.0 - 100% accuracy, all of its samples are so good that discriminator considers them as reals.

Model Architecture

In this homework, I am going to use a well-tested model architecture by Radford et al., 2015 that you can see in Figure 1:

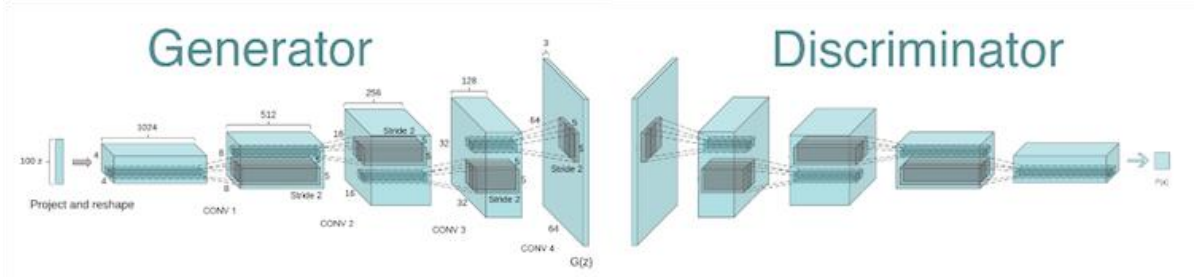


Figure 1

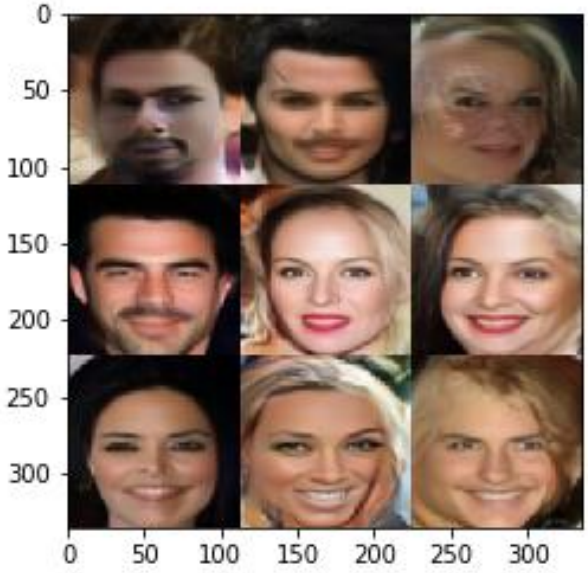
As you can see in the above visualization. Generator and Discriminator have almost the same architectures, but reflected.

Hyperparameters

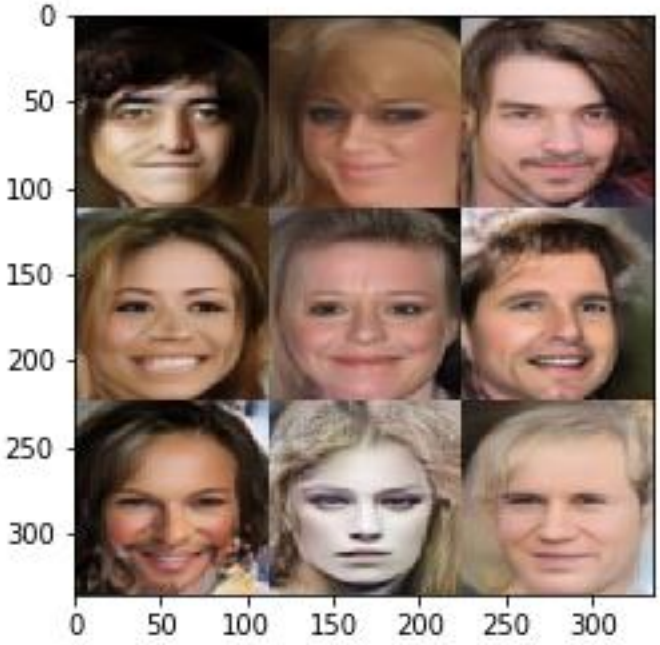
As an optimizer for all models “Adam” optimizer was chosen. The learning rate for discriminator was 0.00004 chosen. The learning rate for generator was 0.0002 chosen. The batchsizes 32 and 64 were tested and finally 64 was chosen. The number of epochs in the range (10:30) were checked and the best results were shown when choosing 20. The input image size in the range (56:112) were tested and finally, 112x112 size were selected. Beta1 for “Adam” optimizer was 0.5 chosen. Noise size was 100 chosen.

Results

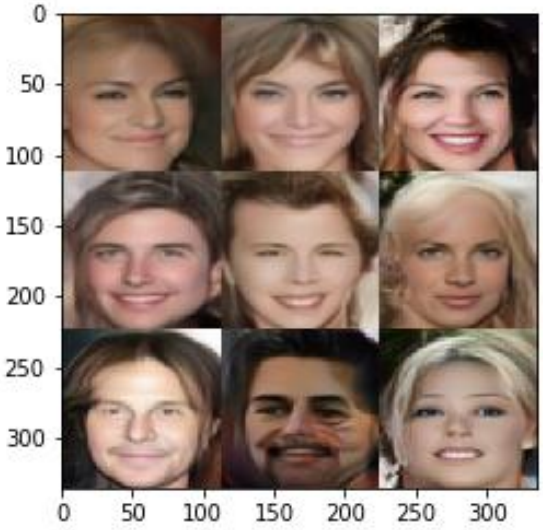
After 10 epochs



After 15 epochs



After 20 epochs



Summary

Obtained model [4] did not show the best results, as, for example, the Style Gan could show. I've tried to change dimension of images in StyleGan, but it didn't go well.

It was not so clear to me which distribution I should use. Which one will work better, uniform or normal? Hence, I just used uniform distribution.

References

1. *https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf*
2. *https://arxiv.org/pdf/1511.06434.pdf*
3. *https://github.com/HACKERSHUBH/Face-Genaration-using-Generative-Adversarial-Network*
4. *https://github.com/alishsuper/Selected-Topics-in-Visual-Recognition-using-Deep-Learning/tree/master/HW2*