Deep Learning Assignment IV: DC-GAN

Karim Armanious

May 17, 2021

1 Introduction

DC-GAN is a foundational adversarial framework developed in 2015.

It had a major contribution in streamlining the process of designing adversarial frameworks and visualizing intermediate representations, thus, making GANs more accessible to both researchers and practitioners. This was achieved by enhancing the concept of adversarial training (introduced by Ian Goodfellow one year prior) with then-state-of-the-art advances in deep learning such as strided and fractional-strided convolutions, batch normalization and LeakyReLU activations.

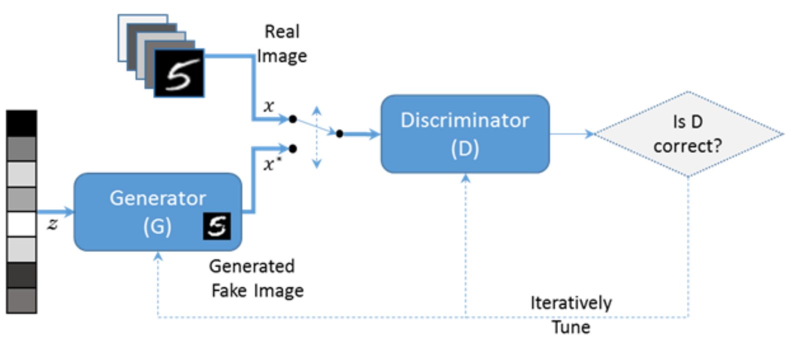

Figure 1: An overview of the DC-GAN framework

2 DC-GANs

In this programming exercise, you are tasking with creating a miniature Deep Convolutional Generative Adversarial Network (DC-GAN) framework for the generation of MNIST digits. The goal is to bridge the gap between the theoretical concept and the practical implementation of GANs.

The desired DC-GAN network should consist of two principal components: the generator G and the discriminator D. The generator should receive as input a 100-dimensional random noise vector z and outputs a synthetically generated MNIST digit $G(z)$ of pixel size $28 \times 28 \times 1$. As the adversarial training continues over time, the output digits should increasingly resemble handwritten digits as shown below.

Figure 2: Sample output images from the MNIST dataset

The discriminator network receives both the synthetically generated digits as well as ground-truth MNIST digits x as inputs. D is trained as a binary classifier. In other words, it is trained to assign the correct label (real vs fake) to both sets of input images. On the other hand side, G is motivated to fool the discriminator into making a false decision by implicitly improving the quality of the output synthetic image. This adversarial training procedure, where both networks are trained with opposing goals, is represented by the following min-max optimization task:

$$\min_G \max_D \mathcal{L}_{\text{adv}} = \min_G \max_D \mathbb{E}_x \left[\log D(x)\right] + \mathbb{E}_z \left[\log\left(1 - D\left(G(z)\right)\right)\right] \tag{1}$$

3 Task

In order to complete this assignment, please fill out the code skeleton provided with this task description and solve the following tasks.

1. Complete the data loading pipeline by normalizing and shuffling the data-samples appropriately.
2. Determine the layers of both the generator and discriminator architectures using the given information in the code skeleton.

3. Define the adversarial loss components used for training both the generator and the discriminator.

4. Define the adversarial training loop.

5. Create an output GIF file to illustrate the progress throughout the training in the resultant quality of the generated images.

4 Further Practice

How does the generated digits compare with the original MNIST? Optimize the network design and training hyperparameters further for better results.

Repeat the above steps for other similar datasets such as Fashion-MNIST or expand the capacities of the network appropriately to suit larger datasets such as the Large-scale Celeb Faces Attributes (CelebA) dataset.