Lab6

鄭余玄

1 Introduction

In this lab, we are going to implement an InfoGAN based on DCGAN using the MNIST dataset. The training and evaluation method is described in Section 2, and the setup of the training is described in Section 3.

2 Method

The main structure of InfoGAN is almost the same as DCGAN, except for the mutual information loss. The input dataset is MNIST, which will be described in Subsection 2.1. The loss function of the discriminator and the mutual information will be shown in Subsection 2.2. Finally, the method of evaluation is described in Subsection 2.3.

2.1 MNIST

The MNIST dataset is loaded from `torchvision`, and images are preprocessed the same as in the DCGAN example. The image is resized to 64×64 and normalized. However, in `InfoGAN` we do not need the ground-truth label, and thus I discard those to save the memory of GPU.

2.2 InfoGAN

The generator and discriminator are almost the same as those in DCGAN, except for the extra `Q` network structure. The `Q` network shares the front of the neural network with the discriminator.

The discriminator is trained first, which is the same as in DCGAN, and its loss function is

$$\mathcal{L}_D = -\mathbb{E}_{x \sim p_r}[\log D(x)] - \mathbb{E}_{x \sim p_g}[\log (1 - D(x)]$$

where p_r is sampled from real data and p_g is sampled from the generator. The input of the generator is a gaussian noise vector with zero mean and unit variance.

The discriminator is trained after, and its loss function is

$$\mathcal{L}_G = -\mathbb{E}_{x \sim p_g}[\log D(x)]$$

.

The mutual information lower bound is finally optimized, and its loss function is

$$\mathcal{L}_I(G, Q) = -\mathbb{E}_{c \sim p_c, x \sim p_g}[\log Q(c|x)]$$

where p_c is sampled uniformly from each possible categories, that is a one-hot multinomial distribution with equal probability.

2.3 Evaluation

The evaluation fixed either one of the input gaussian noise z or the category c and varies another one. For each row, z is fixed and the category c varies, which results in different digits. On the other hand, for each column, c is fixed and the input gaussian noise z varies, which results in the same digit but with different shapes.

3 Result

The setup of the optimizer (`Adam`) and the criterion (`BCELoss`) is the same as in DCGAN. The batch size is 64 and the final epoch is 200.

The result samples at the 200-th epoch are shown in Figure 1. Since InfoGAN is unsupervised, the category c is not necessary the same as the displayed digit. Most digit can be distinguished visually in spite of some digits are defective.

Figure 2 shows the loss during training. The loss of the mutual information (`Loss_Q`) decreases to zero about 100 epochs, and the loss of discriminator (`Loss_D`) gradually decreases to zero except for some periodic increment. In comparison, the loss of generator (`Loss_G`) fluctuates more after 100-th epoch, and therefore I think it was caused by balancing the generator and the discriminator loss but under the condition that the mutual information of the `Q` and the condition `c` had been learned.

Figure 3 shows the probability of real data and fake data. Both real and fake can be discriminated well, and the probability becomes lower after an update.

Figure 1: Sample at 200-th epoch

4 Discussion

In the very beginning of my implementation, it was almost using the DCGAN structure, and thus the resulting sample image looks much like the transposed one of my current final result, and even produce weird symbols. This result could be viewed as a contrast to my current correct implementation, which ensures the mutual information between the `Q` and the condition c is maximized.

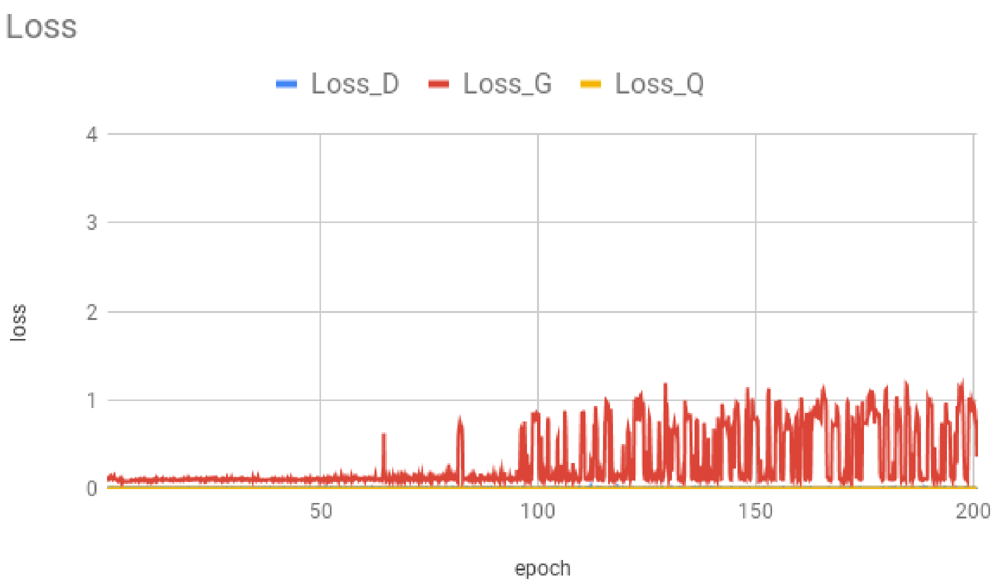

Figure 2: Loss

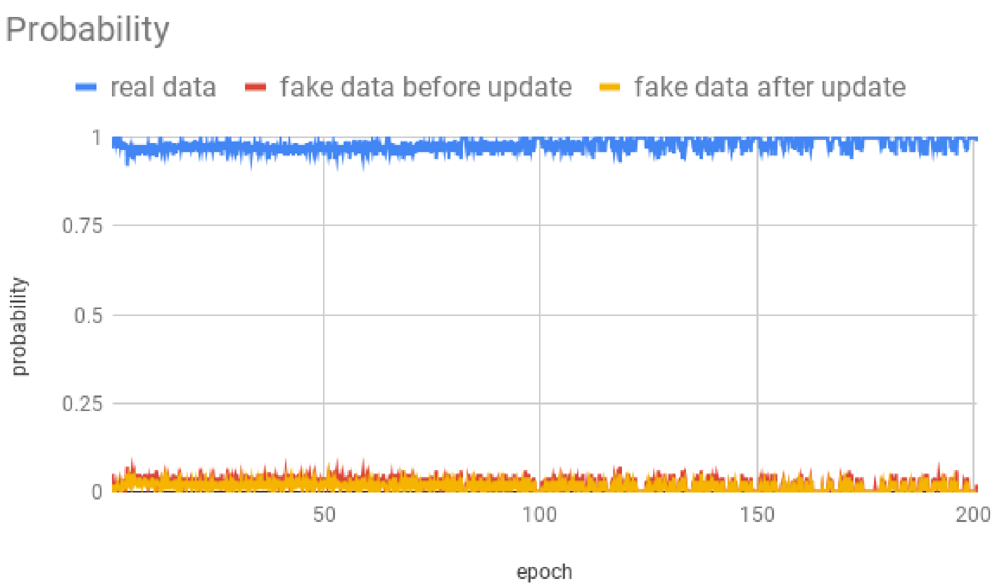

Figure 3: Probability