Lab 4: InfoGAN

Lab objective

In this lab, you will learn the classical form of GAN loss, and you will need to implement InfoGAN trained on MNIST, which adopts adversarial loss to generate realistic images and learns the disentangled representations by maximizing mutual information.

Important Date

1. Deadline: 8/15 (Thu.) 11:59
2. Demo date: 8/15 (Thu)

Turn in

1. Experimental Report (.pdf) and Source code

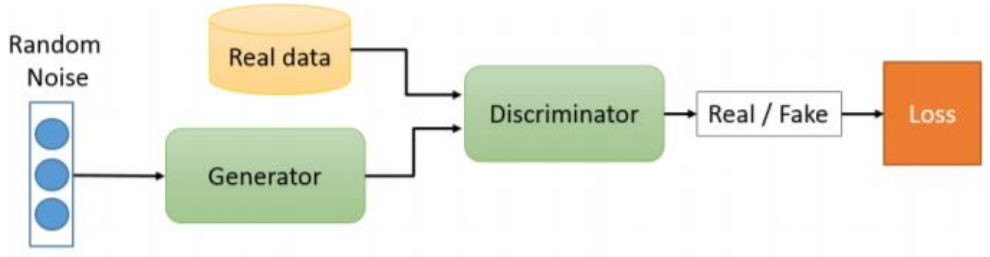
Notice: zip all files in one file and name it like DLP_LAB4_your studentID_name.zip. e.g. DLP_LAB4_0756051_李仕柏.zip

Requirements

1. Modify the generator and the discriminator in InfoGAN (https://github.com/pianomania/infoGAN-pytorch)
2. Adopt traditional generator and discriminator loss. Maximize the mutual information between generated images and discrete one-hot vector
3. Show the generated images of the specific number
4. Plot the loss curves of the generator and the discriminator while training

Implementation details

1. Generative Adversarial Network (GAN)
   A. GAN consists of two major components: a generator and a discriminator. The discriminator, which is a binary classifier, tries to distinguish fake inputs from real inputs, while the generator tries to fool the discriminator.



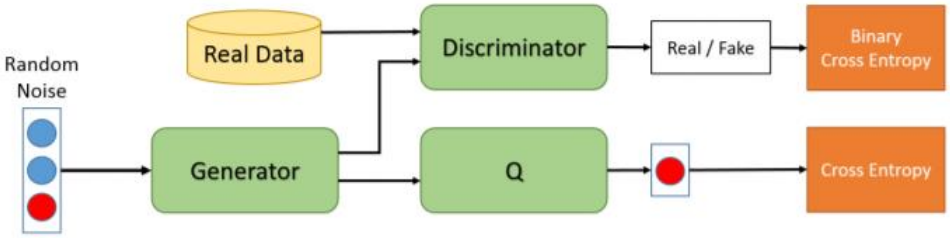
   B. The loss function of the discriminator

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

   C. The loss function of the generator

$$\mathcal{L}_{\mathrm{G}} = E_{x\sim p_g}[-\log D(x)]$$

2. Deep Convolution Generative Adversarial Network (DCGAN)
   A. DCGAN is an extension of classical GAN. The main contributions of DCGAN includes:
      - Remove all fully-connected layers
      - Add de-convolutional layers to generator (https://datascience.stackexchange.com/questions/6107/what-aredeconvolutional-layers)
      - Add batch normalize layers to both generators and discriminators
      - Replace pooling layers with stride convolutional layers
      - For more details, please refer to the reference
   B. Sample code: https://github.com/pytorch/examples/tree/master/dcgan
3. InfoGAN (https://arxiv.org/abs/1606.03657)
   A. Overview



   B. Random noise in this lab consists of:
      - 54-D continuous noise drawn from standard normal distribution (zero mean and unit variance)
      - 10-D discrete one hot vector. E.g., (0, 1, 0, 0, 0, 0, 0, 0, 0, 0)
   C. Maximizing mutual information to force models to use useful information

$$
\begin{aligned}
I(c; G(z,c)) &= H(c) - H(c|G(z,c)) \\
&= \mathbb{E}_{x\sim G(z,c)}[\mathbb{E}_{c'\sim P(c|x)}[\log P(c'|x)]] + H(c) \\
&= \mathbb{E}_{x\sim G(z,c)}[\underbrace{D_{\mathrm{KL}}(P(\cdot|x) \parallel Q(\cdot|x))}_{\geq 0} + \mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]] + H(c) \\
&\geq \mathbb{E}_{x\sim G(z,c)}[\mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]] + H(c)
\end{aligned}
$$

   D. Lower bound of the mutual information

$$
\begin{aligned}
L_I(G,Q) &= \boxed{E_{c\sim P(c),x\sim G(z,c)}[\log Q(c|x)] + H(c)} \\
&= E_{x\sim G(z,c)}[\mathbb{E}_{c'\sim P(c|x)}[\log Q(c'|x)]] + H(c) \\
&\leq I(c; G(z,c))
\end{aligned}
$$

   E. The loss function of the generator now becomes:

$$\mathcal{L}_G = E_{x\sim p_g}[-\log D(x)] + \mathrm{L_I}(G,Q)$$

- **Output example**

  e.g. one-hot vector = [1,0,0,0,0,0,0,0,0,0]

- **Scoring Criteria**

1. Report (60%)
   A. Introduction (10%)
   B. Implementation details (30%)
      - Describe your generator and discriminator architectures. You must adopt other different model architecture (different channel size, activation function, normalization, layer numbers, etc). Network Q can simply follow the reference code. (20%)
      - Specify the hyperparameters and setting (e.g. channels, learning rate, optimizer, epochs, etc.) (10%)
   C. Results and discussion (20%)
      - Show your results of all MNIST numbers (10%)
      - Discuss your training process the results of the loss curves (10%)
2. Demo (50%)
   A. The TA will give you 2 numbers and you have to generate 10 same numbers for each of them but with different styles (noises) (10%+10%)
      - 10 correct numbers ---- 100%
      - 9 correct numbers ---- 90%
      - > 7 correct numbers ---- 80%
      - Otherwise ---- 0%
   B. Questions (20%)

- **Reference**

  Here are some model architectures that you can use as your models

1. https://github.com/tensorlayer/dcgan
2. https://github.com/nashory/pggan-pytorch