Homework 5

>>NAME HERE<<
>>ID HERE<<

Instructions:

- Please submit your answers in a single pdf file and your code in a zip file. pdf preferably made using latex. No need to submit latex code.
- Submit code for programming exercises. Though we provide a base code with python (jupyter notebook, you can import it in colab and choose GPU as the runtime environment), you can use any programming language you like as long as you use the same model and dataset.

1 Implementation: GAN (55 pts)

In this part, you are expected to implement GAN with MNIST dataset. We have provided a base jupyter notebook (gan-base.ipynb) for you to start with, which provides a model setup and training configurations to train GAN with MNIST dataset.

(a) Implement training loop and report learning curves and generated images in epochs 1, 50, and 100. Note that drawing learning curves and visualization of images are already implemented in provided jupyter notebook. (20 pts)

Procedure 1 Training GAN, modified from Goodfellow et al. (2014)

Input: m: real data batch size, n_z: fake data batch size
Output: Discriminator D, Generator G
for number of training iterations **do**
    # Training discriminator
    Sample minibatch of n_z noise samples $\{\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \cdots, \mathbf{z}^{(n_z)}\}$ from noise prior $p_g(\mathbf{z})$
    Sample minibatch of $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(m)}\}$
    Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d}\big(\frac{1}{m}\sum_{i=1}^{m}\log D(\mathbf{x}^{(i)}) + \frac{1}{n_z}\sum_{i=1}^{n_z}\log(1 - D(G(\mathbf{z}^{(i)})))\big)$$

    # Training generator
    Sample minibatch of n_z noise samples $\{\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \cdots, \mathbf{z}^{(n_z)}\}$ from noise prior $p_g(\mathbf{z})$
    Update the generator by ascending its stochastic gradient:

$$\nabla_{\theta_g}\frac{1}{n_z}\sum_{i=1}^{n_z}\log D(G(\mathbf{z}^{(i)}))\big)$$

end for
# The gradient-based updates can use any standard gradient-based learning rule. In the base code, we are using Adam optimizer (Kingma and Ba, 2014)

The expected results are as follows.

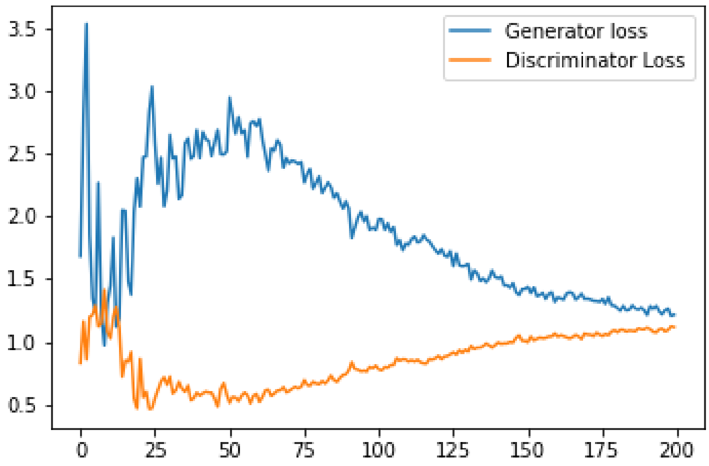
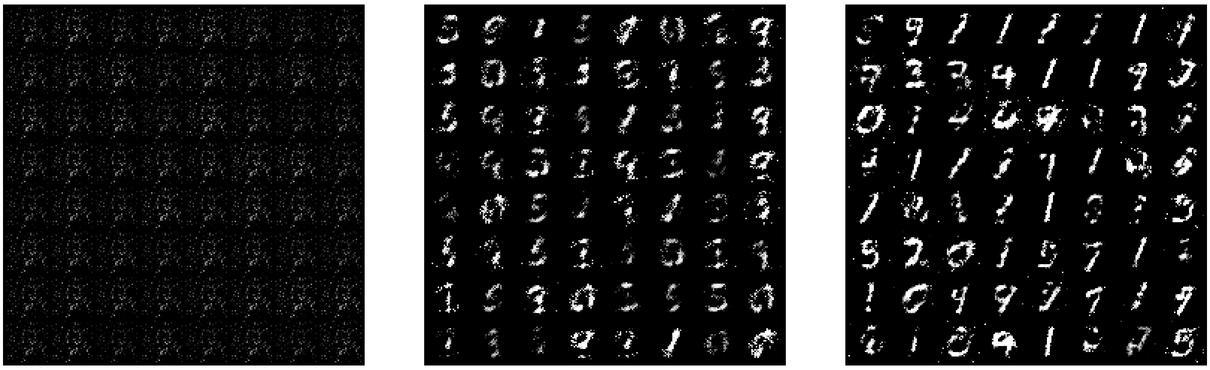


Figure 1: Learning curve



(a) epoch 1 (b) epoch 50 (c) epoch 100

Figure 2: Generated images by G

Solution goes here.

(b) Replace the generator update rule as the original one in the slide,
"Update the generator by descending its stochastic gradient:"

$$\nabla_{\theta_g} \frac{1}{n_z} \sum_{i=1}^{n_z} \log(1 - D(G(\mathbf{z}^{(i)}))) \ ,$$

and report learning curves and generated images in epochs 1, 50, and 100. Compare the result with (a). Note that it may not work. If training does not work, explain why it does not work. (10 pts)

Solution goes here.

(c) Except for the method that we used in (a), how can we improve training for GAN? Implement that and report learning curves and generated images in epochs 1, 50, and 100. (10 pts)

Solution goes here.

2 Ridge regression [20 pts]

Derive the closed-form solution in matrix form for the ridge regression problem:

$$\min_{\boldsymbol{\beta}} \left(\frac{1}{n} \sum_{i=1}^{n} (\mathbf{z}_i^\top \boldsymbol{\beta} - y_i)^2 \right) + \lambda \|\boldsymbol{\beta}\|_{\mathbf{A}}^2$$

where

$$\|\boldsymbol{\beta}\|_{\mathbf{A}}^2 := \boldsymbol{\beta}^\top \mathbf{A} \boldsymbol{\beta}$$

and

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

This $\mathbf{A}$ matrix has the effect of NOT regularizing the bias β_0, which is standard practice in ridge regression. Note: Derive the closed-form solution, do not blindly copy lecture notes.
Solution goes here.

3 Review the change of variable in probability density function [25 pts]

In Flow based generative model, we have seen $p_\theta(x) = p(f_\theta(x))|\frac{\partial f_\theta(x)}{\partial x}|$. As a hands-on (fixed parameter) example, consider the following setting.

Let X and Y be independent, standard normal random variables. Consider the transformation $U = X + Y$ and $V = X - Y$. In the notation used above, $U = g_1(X, Y)$ where $g_1(X, Y)$ where $g_1(x, y) = x + y$ and $V = g_2(X, Y)$ where $g_2(x, y) = x - y$. The joint pdf of X and Y is $f_{X,Y} = (2\pi)^{-1} exp(-x^2/2) exp(-y^2/2), -\infty < x < \infty, -\infty < y < \infty$. Then, we can determine u, v values by x, y, i.e. $\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$.

(a) (5 pts) Compute Jacobian matrix

$$J = \begin{bmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \end{bmatrix}$$

(5 pts)

Solution goes here.

(b) (Forward) Show that the joint pdf of U, V is

$$f_{U,V}(u, v) = \big(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-u^2/4)\big)\big(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-v^2/4)\big)$$

(10 pts)

(Hint: $f_{U,V}(u, v) = f_{X,Y}(?, ?)|det(J)|$)

Solution goes here.

(c) (Inverse) Check whether the following equation holds or not.

$$f_{X,Y}(x, y) = f_{U,V}(x + y, x - y)|det(J)^{-1}|$$

(10 pts)

Solution goes here.

References

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.