HOMEWORK 5

Jackson Hellmers
9075444662

11/07/21

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. See piazza post @114 for some recommendations on how to write the answers.
- Submit code for programming exercises. Though we provide a base code with python (jupyter notebook), you can use any programming language you like as long as you use the same model and dataset.
- Submit all the material on time.

1 Implementation: GAN (40 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 epoch 1, 50, 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 $\{z^{(1)}, z^{(2)}, \cdots, z^{(n_z)}\}$ from noise prior $p_g(z)$
Sample minibatch of $\{x^{(1)}, x^{(2)}, \cdots, x^{(m)}\}$
Update the discriminator by ascending its stochastic gradient:

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

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

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

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)

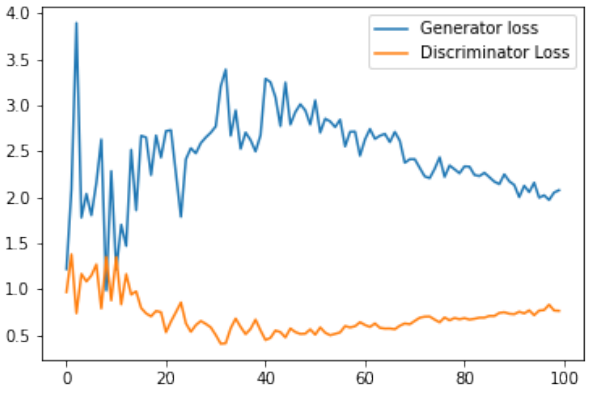
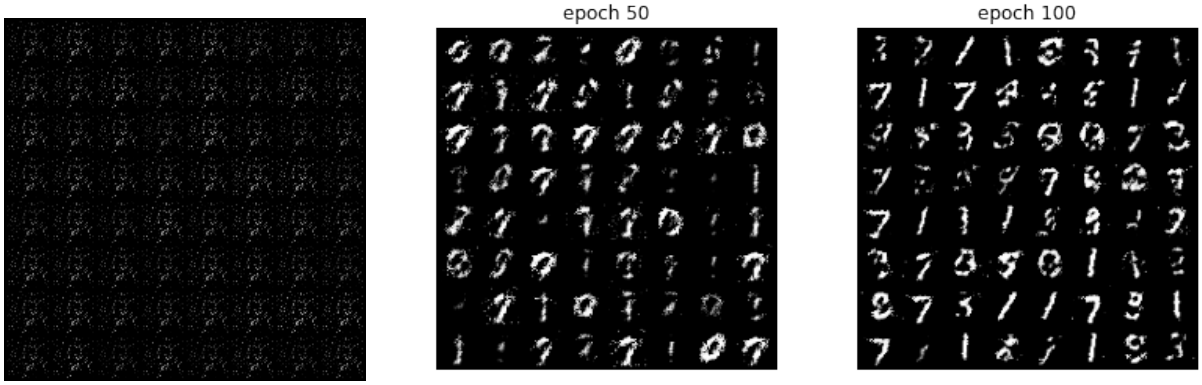


Figure 1: Personal Learning curve



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

Figure 2: Personal Generated images by G

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

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

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

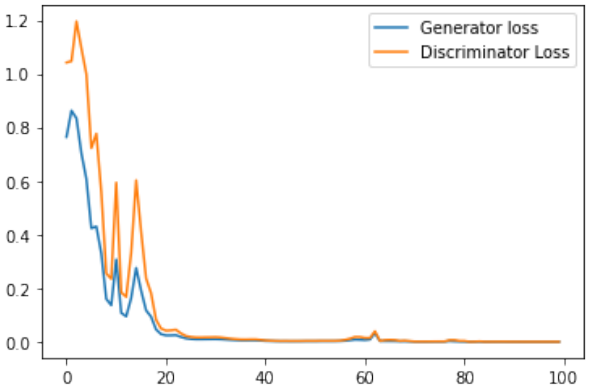
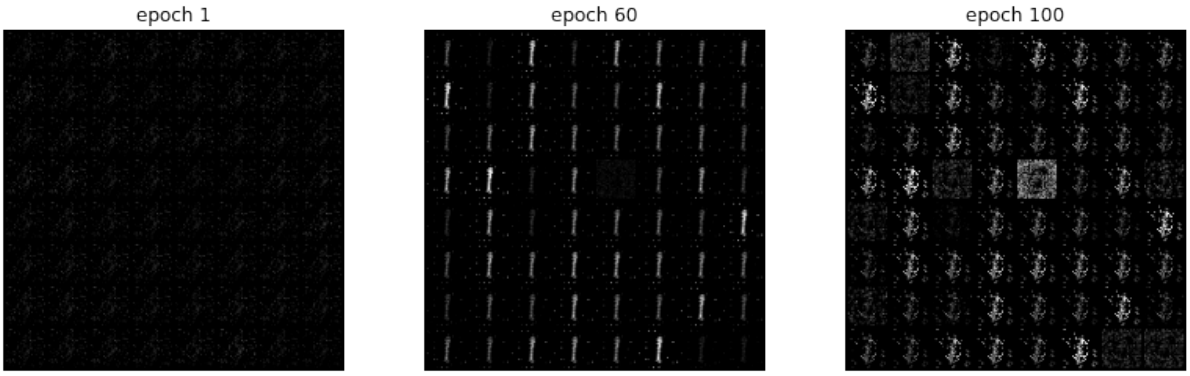


Figure 3: Personal Learning curve



(a) epoch 1 (b) epoch 60 (c) epoch 100

Figure 4: Personal Generated images by G

I showed epoch 60 instead of 50 as it was the epoch that was closest to converging to an actual solution (though it looks like it would have mode collapsed).

Looking at the new update rule for the generator we see that we are taking the log of $(1 - D(G(z))$. Our $G(z)$ generates fake images so we know its true label is 0. The discriminator network attempts to predict the label of the given image yielding 0 when it thinks the given image is fake and 1 when real. We are attempting to fool the discriminator so we hope that the generated image yields a 1. Since we are taking the log when the discriminator predicts a 1 we get $log(1 - 1) = log(0) = -\infty$ and when the discriminator predicts a 0 we get $log(1 - 0) = log(1) = 0$. So our GD only takes a step when the discriminator predicts a 1 which is opposite of the behavior we desire.

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

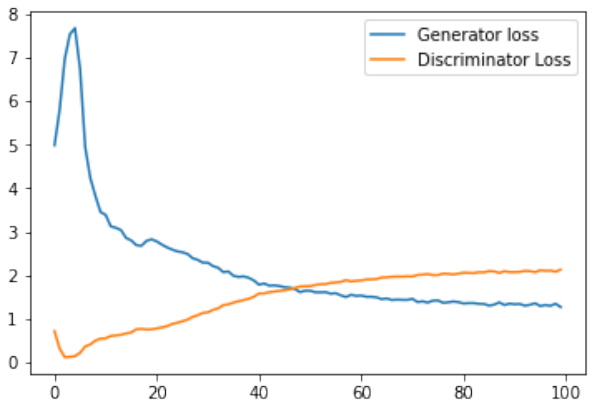


Figure 5: Personal Learning curve

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

Figure 6: Personal Generated images by G

There are a lot of hyper parameters we could attempt to optimize over such as the value of k (discriminator update steps), the optimizer learning rates and the latent vector size (nz). The images shown above were generated by updating the value of k to 2 ($k = 2$). This however created some instability to the training so I also had to adjust the normalization transform parameters and the learning rates. I included the 10th epoch instead of the first to show how the model was able to overcome the mode collapse.

2 Ridge regression [15 pts]

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

$$\min_{\beta} \left(\frac{1}{n} \sum_{i=1}^{n} (z_i^\top \beta - y_i)^2 \right) + \lambda \|\beta\|_A^2$$

where

$$\|\beta\|_A^2 := \beta^\top A \beta$$

and

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

This 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.

$\nabla \frac{1}{n}(Z^T\beta - Y)^2 + \lambda\beta^T A\beta = 0$
$\nabla(Z\beta - Y)^T(Z\beta - Y) + \lambda\beta^T A\beta = 0$
$Z^TY = (Z^TZ + \lambda A)\beta \;\rightarrow\; (Z^TZ + \lambda A)^{-1}Z^TY = \beta$

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)

$$X = \frac{U+V}{2} \quad Y = \frac{U-V}{2}$$

$$\frac{\partial x}{\partial u} = \frac{1}{2} \quad \frac{\partial x}{\partial v} = \frac{1}{2} \quad \frac{\partial y}{\partial u} = \frac{1}{2} \quad \frac{\partial y}{\partial v} = \frac{-1}{2}$$

$$J = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{-1}{2} \end{bmatrix}$$

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

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

(10 pts)

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

$|J| = |-0.5| = 0.5$

$f_{U,V}(u,v) = f_{X,Y}(\frac{u+v}{2}, \frac{u-v}{2})|J|$

This is where the two $\sqrt{2}$s come from as $\frac{1}{\sqrt{2}} * \frac{1}{\sqrt{2}} = 0.5$

$f_{U,V} = 0.5 * f_{X,Y} = 0.5 * \left(\frac{1}{\sqrt{2\pi}} exp(-(\frac{u+v}{2})^2/2)\right)\left(\frac{1}{\sqrt{2\pi}} exp(-(\frac{u-v}{2})^2/2)\right)$

$= \left(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-u^2/4)\right)\left(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-v^2/4)\right)$

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

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

(10 pts)

$|J|^{-1} = |-0.5|^{-1} = 2$

$f_{X,Y} = 2 * f_{U,V} = 2 * \left(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-(x+y)^2/4)\right)\left(\frac{1}{\sqrt{2\pi}\sqrt{2}} exp(-(x-y)^2/4)\right)$

$= \left(\frac{1}{\sqrt{2\pi}} exp(-x^2/2)\right)\left(\frac{1}{\sqrt{2\pi}} exp(-y^2/2)\right)$

4 Directed Graphical Model [20 points]

Consider the directed graphical model (aka Bayesian network) in Figure 7.

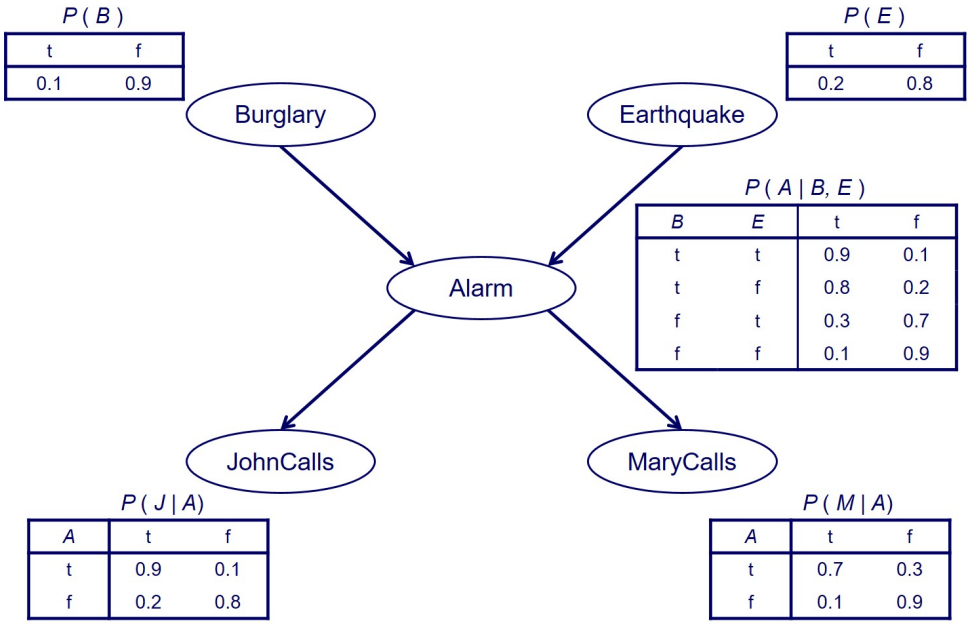


Figure 7: A Bayesian Network example.

Compute $P(B = t \mid E = f, J = t, M = t)$ and $P(B = t \mid E = t, J = t, M = t)$. These are the conditional probabilities of a burglar in your house (yikes!) when both of your neighbors John and Mary call you and say they hear an alarm in your house, but without or with an earthquake also going on in that area (what a busy day), respectively.

$$P(B = t, E = f, J = t, M = t) = P(B)P(E)P(A \mid B, E)P(J \mid A)P(M \mid A)$$

$$P(B = t \mid E = f, J = t, M = t) = \frac{P(B = t, E = f, J = t, M = t)}{P(E = f, J = t, M = t)} = \frac{0.04064}{0.04064 + 0.0583} = 0.4107$$

$$P(B = t \mid E = t, J = t, M = t) = \frac{P(B = t, E = t, J = t, M = t)}{P(E = t, J = t, M = t)} = \frac{0.0114}{0.0114 + 0.0365} = 0.2380$$

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*.