HOMEWORK 6

Mingcong Cao
9084259218

Instructions: Use this latex file as a template to develop your homework. Submit your homework on time as a single pdf file. Please wrap your code and upload to a public GitHub repo, then attach the link below the instructions so that we can access it. Answers to the questions that are not within the pdf are not accepted. This includes external links or answers attached to the code implementation. Late submissions may not be accepted. You can choose any programming language (i.e. python, R, or MATLAB). Please check Piazza for updates about the homework. It is ok to share the results of the experiments and compare them with each other.
https://github.com/Mingcong-Cao/ECE760_Homework6.git

1 Implementation: GAN (50 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:

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

  \# 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)

Solutions are shown below

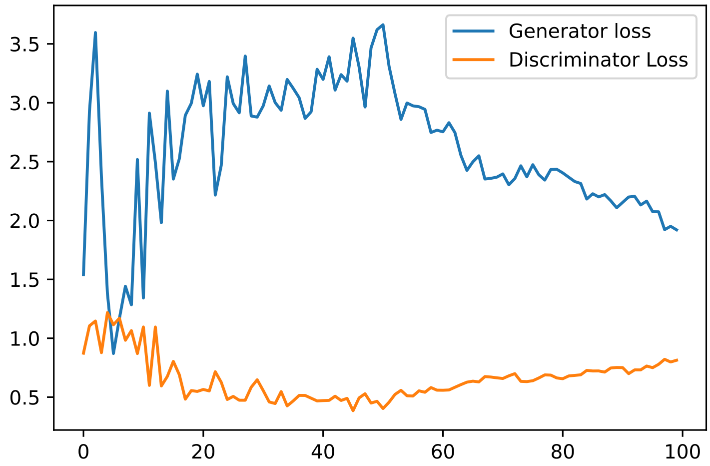
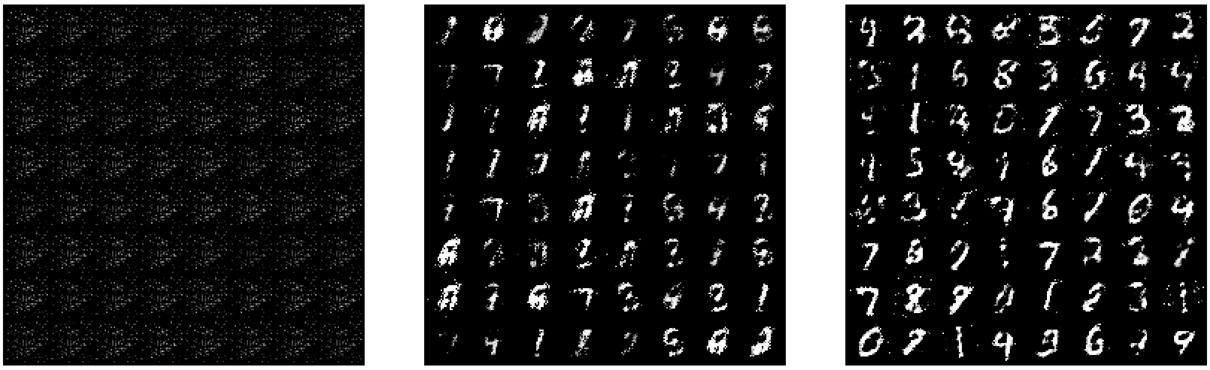


Figure 1: Learning curve



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

Figure 2: 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.
You may find this helpful: https://jonathan-hui.medium.com/gan-what-is-wrong-with-the-gan-cost-function-6f594162ce01 (10 pts)

According to the post, when the discriminator becomes nearly perfect, the gradient of the cost function in (b) will be almost zero everywhere, leading to failure to train the network. As can be observed in figure 3, the loss quickly converges to zero.

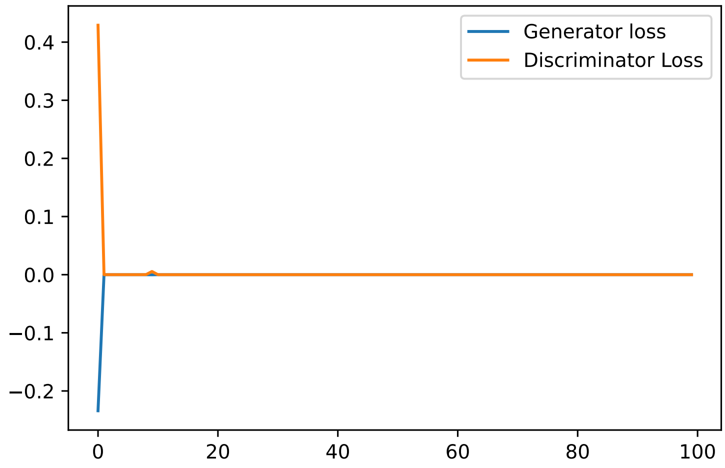
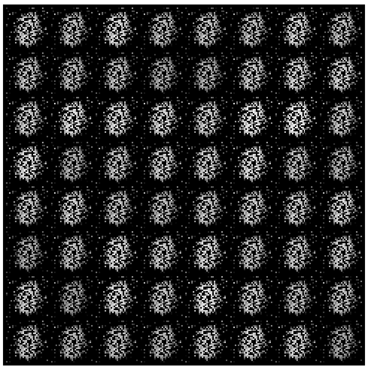
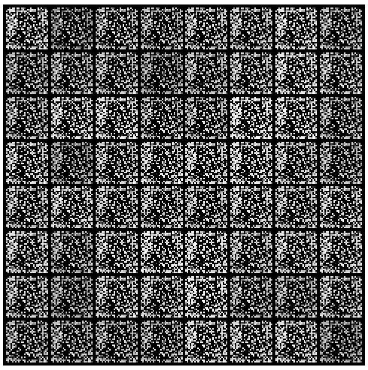


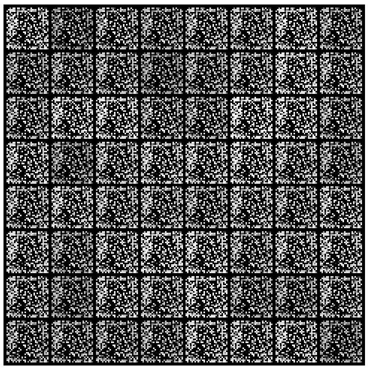
Figure 3: Learning curve



(a) epoch 1



(b) epoch 50



(c) epoch 100

Figure 4: Generated images by G

(c) Except the method that we used in (a), how can we improve training for GAN? Implement that and report your setup, learning curves, and generated images in epoch 1, 50, 100. This question is an open-ended question and you can choose whichever method you want. (20 pts)

I implemented conditional GAN, where the label is one-hot encoded and passed to both the generator and discriminator. Noise with $\sigma = 0.1$ is added. Now the generation is controlled by the label, we could see each row corresponds to one digit.

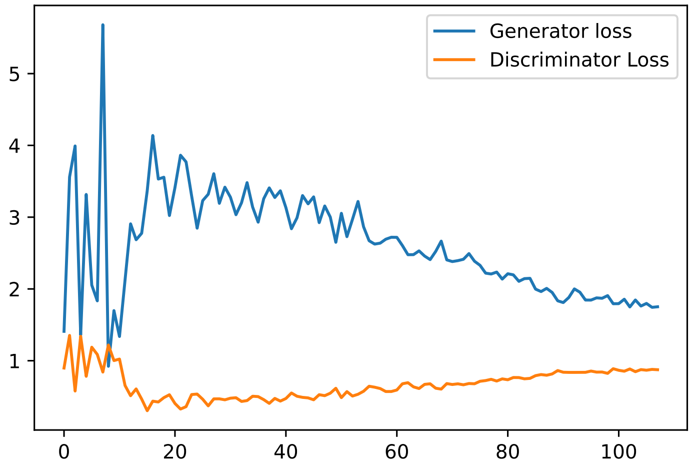
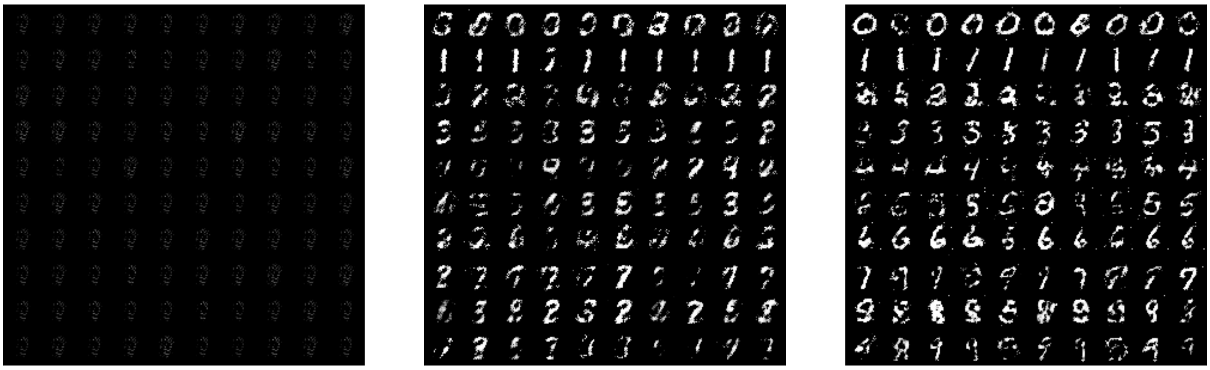


Figure 5: Learning curve for CGAN



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

Figure 6: Generated images by $CGAN$

2 Directed Graphical Model [25 points]

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

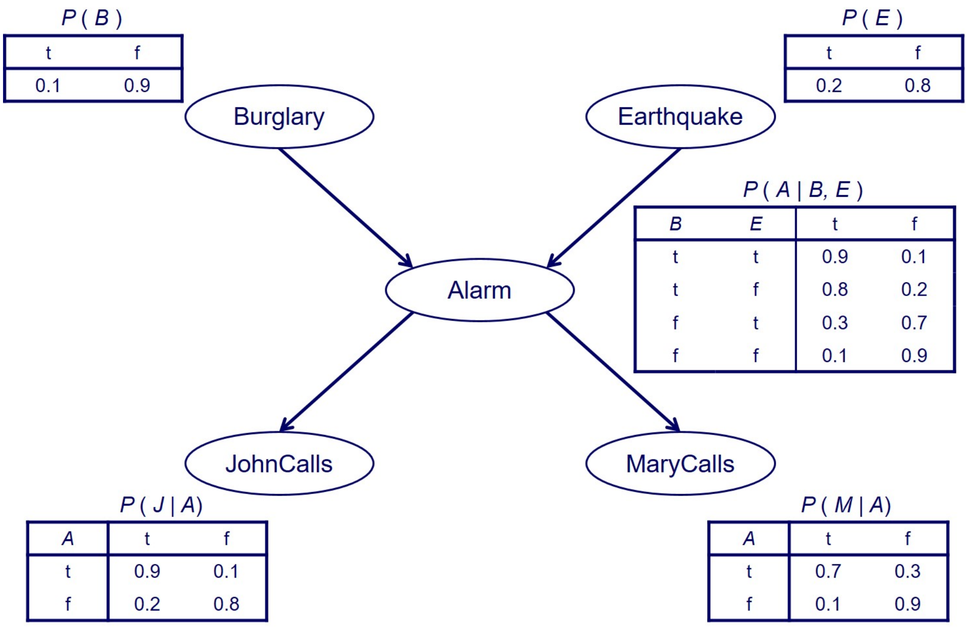


Figure 7: A Bayesian Network example.

Compute $P(B = t \mid E = f, J = t, M = t)$ and $P(B = t \mid E = t, J = t, M = t)$. (10 points for each) 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.

First question:

$P(B = t, E = f, J = t, M = t) = \sum_A P(B = t)P(E = f)P(A|B = t, E = f)P(J = t|A)P(M = t|A) = 0.1 * 0.8 * (0.8 * 0.9 * 0.7 + 0.2 * 0.2 * 0.1) = 0.04064$

$P(B = f, E = f, J = t, M = t) = \sum_A P(B = f)P(E = f)P(A|B = t, E = f)P(J = t|A)P(M = t|A) = 0.9 * 0.8 * (0.1 * 0.9 * 0.7 + 0.9 * 0.2 * 0.1) = 0.05832$

$P(B = t \mid E = f, J = t, M = t) = \frac{P(B=t,E=f,J=t,M=t)}{P(B=t,E=f,J=t,M=t)+P(B=f,E=f,J=t,M=t)} = 0.411$

Second question:

$P(B = t, E = t, J = t, M = t) = \sum_A P(B = t)P(E = f)P(A|B = t, E = t)P(J = t|A)P(M = t|A) = 0.1 * 0.2 * (0.9 * 0.9 * 0.7 + 0.1 * 0.2 * 0.1) = 0.01138$

$P(B = f, E = t, J = t, M = t) = \sum_A P(B = t)P(E = f)P(A|B = t, E = t)P(J = t|A)P(M = t|A) = 0.9 * 0.2 * (0.3 * 0.9 * 0.7 + 0.7 * 0.2 * 0.1) = 0.03654$

$P(B = t \mid E = t, J = t, M = t) = \frac{P(B=t,E=t,J=t,M=t)}{P(B=t,E=f,J=t,M=t)+P(B=f,E=f,J=t,M=t)} = 0.237$

3 Chow-Liu Algorithm [25 pts]

Suppose we wish to construct a directed graphical model for 3 features X, Y, and Z using the Chow-Liu algorithm. We are given data from 100 independent experiments where each feature is binary and takes value T or F. Below is a table summarizing the observations of the experiment:

X	Y	Z	Count
T	T	T	36
T	T	F	4
T	F	T	2
T	F	F	8
F	T	T	9
F	T	F	1
F	F	T	8
F	F	F	32

1. Compute the mutual information $I(X, Y)$ based on the frequencies observed in the data. (5 pts)

   all log functions have base 2
   $I(X,Y) = H(X) - H(X|Y) = (-2 * 0.5 * log(0.5)) - (0.4 * (-0.1 * log(0.1) - 0.9 * log(0.9)) + 0.6 * (-0.167 * log(0.167) - 0.833 * log(0.833))) = 0.422$

2. Compute the mutual information $I(X, Z)$ based on the frequencies observed in the data. (5 pts) $I(X,Z) = H(X) - H(X|Z) = (-2 * 0.5 * log(0.5)) - (0.55 * (-0.691 * log(0.691) - 0.309 * log(0.309)) + 0.45 * (-0.267 * log(0.267) - 0.733 * log(0.733))) = 0.133$

3. Compute the mutual information $I(Z, Y)$ based on the frequencies observed in the data. (5 pts) $I(Z,Y) = H(Z) - H(Z|Y) = (-0.45 * log(0.45) - 0.55 * log(0.55)) - (0.5 * (-0.9 * log(0.9) - 0.1 * log(0.1)) + 0.5 * (-0.8 * log(0.8) - 0.2 * log(0.2))) = 0.397$

4. Which undirected edges will be selected by the Chow-Liu algorithm as the maximum spanning tree? (5 pts)
   X-Y and Y-Z will be chosen since the mutual information between them are larger.

5. Root your tree at node X, assign directions to the selected edges. (5 pts)
   The direction will be directed from root. $X \rightarrow Y$ and $Y \rightarrow Z$

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*.