GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL of ELECTRICAL and COMPUTER ENGINEERING

ECE 8803-GGDL Fall 2023
Problem Set #3

Assigned: 20 Oct
Due Date: 30 Oct

Please contact the TAs for clarification on the instructions in the homework assignments.

Problem 1: GANs on MNIST (50 points). You will implement a Generative Adversarial Network model and evaluate it under different conditions. You are provided with a notebook `gan-mnist.ipynb`, in which part of the necessary code has already been filled out for you. Pay attention to the cells marked **YOUR CODE HERE**: these are the only cells you need to edit. For those using the PACE clusters, a conda environment has been provided for you with the necessary packages for this notebook: `hw3-p1`.

a. Part 1: Libraries and utility functions. Read through the header sections of the notebook and run the corresponding cells to initialize the necessary libraries and functions (you do not need to edit any code here). Why do you think it is convenient to normalize the images to the range [-1,1] instead of [0,1]?

- (bonus points) How does the minimax game defined by the objective function of a GAN relate to minimizing the Jensen-Shanon divergence between the training data distribution and the samples obtained from the generator? See `https://arxiv.org/abs/1406.2661`.
- (bonus points) Why is maximizing the probability of the discriminator making the INCORRECT choice the most practical way to update the generator? See paper linked above.

b. Part 2: Vanilla GAN. Implement the noise sampling function as well as the generator and discriminator networks according to the parameters specified in the notebook.

c. Part 3: GAN Loss. Implement the necessary losses for both networks.

d. Part 4: Training Function. Implement the optimizer construction, review and run the training loop cells. How do the generated samples look?

e. Part 5: Least Squares GAN. Implement the Least-Squares loss function for the constructed networks and evaluate them. How does the resulting generated samples compare to those produced by the Vanilla GAN? Why do you think this is the case?

f. Part 6: Deeply Convolutional GANs. Implement the generator and discriminator architectures including the convolutional layers specified in the notebook and evaluate the resulting model. How do the generated samples compare to the two previous configurations? Why does adding convolutional layers have this effect?

g. Part 7: Compare generated images. Evaluate the generated samples from the three previous models against the real MNIST images. Can you distinguish the original from the fake ones in all three cases? Do you think an equilibrium exists where the generator wins, i.e. the discriminator ends up unable to distinguish the two distributions on finite samples? Explain why you believe this would or would not be possible.

Problem 2: EBMs on MNIST (50 points). You will implement an Energy Based Model model and evaluate it for generative performance and anomaly detection capacity. You are provided with a notebook `ebm-mnist.ipynb`, in which part of the necessary code has already been filled out for you. Pay attention to the cells marked **YOUR CODE HERE**: these are the only cells you need to edit. For those using the PACE clusters, a conda environment has been provided for you with the necessary packages for this notebook: `hw3-p2`.

a. Part 0/1: Libraries and utility functions. Read through the header sections of the notebook and run the corresponding cells to initialize the necessary libraries and functions (you do not need to edit any code here). How does the contrastive divergence objective change the energy of the correct/wrong samples? What do you think would need to happen for the push/pull forces on the diagram to balance out?

b. Part 2: Model construction and training. Implement the model construction, MCMC sampling steps and loss calculation in the corresponding cells, and run the implemented model.

- Why do you think that the Swish activation function is more convenient than the standard ReLU for this model and task? *Hint: Try to visualize the shape of both activation functions and think about how they would affect the (backward) propagation of the gradient through the network.*
- What is the role of the regularization loss? How do you think the resulting data would look if we were not using it?

c. Part 3: Analysis. Sample a few images using the MCMC method. How many MCMC steps are required to sample reasonable images from the trained model? Is this number different for different digits? Now complete the transformation cells in order to test the out-of-distribution performance of the model.

- For pure random noise inputs: are the scores in line with what would be expected for a good anomaly detection method? Why do you think the score for the 'true' images is so close to zero?
- For the image transformations: do the scores of the transformed images differ significantly from the original ones? Do you think this model would make a good out-of distribution detector? Provide a detailed explanation of why this would or would not be the case.