Assignment 2 – GAN

Due Date: Nov 1, 2022 Midnight

In this assignment we will implement a Generative Adversarial Network to generate samples like the digits in the MNIST dataset. MNIST is a popular digit dataset that is available with most of the deep learning frameworks like PyTorch, TensorFlow, etc. Or, feel free to use the `digits.py` file provided with Assignment 1 to download the MNIST dataset. It consists of 60,000 grayscale images, each 28x28 pixels. The dataset has 10 categories of digits, {0,1,…,9}. **Normalize the pixel values to [-1,1]**. Use the entire dataset of 60,000 images to train the GAN. The architecture of the neural network is as follows:

1. The Generator takes in a vector $z \in \mathbb{R}^d$ which is sampled from a Gaussian distribution $N(0,1)$ with $d = 100$. The Generator is a fully connected neural network with the following architecture:

   z → 256 → 512 → 1024 → 784

   Use a Leaky ReLU as activation for the layers (for e.g., LeakyReLU with negative slope 0.2), expect for the last layer, where you use a tanh() activation. The output of the last layer (784 dimensions) is converted to an image 28 x 28 pixels.

2. The Discriminator is also a fully connected network with LeakyReLU activations except for the last layer, which uses Sigmoid activation. The architecture is as follows:

   784 → 512 → 256 → 1

   The Discriminator takes as input real MNIST images and generated MNIST images (output of the Generator) and classifies them using a binary classifier.

3. Label Smoothing: We will use Label Smoothing when training the Discriminator and Generator. When training the Discriminator, the labels are usually 0 for generated/fake images and 1 for real images. With Label Smoothing, the labels for real images are random values between (0.8, 1) and labels for generated images are random values between (0, 0.2). You can use Label Smoothing when training the Generator as well.

   Submission Format and Grading:
   1. Implement the assignment in Python and submit a Jupyter Notebook file with the following name format `Assignment2_FirstName_LastName.ipynb`
   2. Convert your notebook to pdf after saving it with all the outputs and save the file as `Assignment2_FirstName_LastName.pdf`
   3. Generator implemented with the prescribed architecture (20 pts)
   4. Discriminator implemented with the prescribed architecture (20 pts)
   5. Label Smoothing (20 pts)
   6. When executed the notebook should output the following:
      a. Training curves (Discriminator/Generator loss vs epochs) (20 pts)

b. A 10 x 10 grid plot of 100 randomly generated images which are the outputs of the Generator. (20 pts)

Your submissions will be executed to verify your solutions. The code and report will be evaluated for plagiarism. Ensure the notebook will execute on a generic Google Colab environment without the need for change/debugging. If it is not possible to execute your code, it is not possible to verify your results. The submission will also be evaluated for the quality of images in each of these outputs.