Vanilla GAN/DC GAN using PyTorch

Pseudocode

Pre-Requisites:

- **Setup and data loading**:
    - Import necessary libraries (PyTorch, nn, optim, torchvision, utils).
    - Define the data folder path.
    - Define a function to load MNIST data with transformations.
    - Load the MNIST dataset and create a data loader.
    - Calculate the number of batches.

Network Definition:

- **Network architecture**:
    - Define the Discriminator Network (DiscriminatorNet) with multiple linear layers, LeakyReLU activations, and Dropout.
    - Define the Generator Network (GeneratorNet) with multiple linear layers and LeakyReLU activations, sigmoid activations,ending with a Tanh activation.
    - Define helper functions to convert images to vectors and vectors to images.

Main Execution:

- **Initialization and optimization**:
    - Initialize the Discriminator and Generator networks.
    - Move the networks to GPU if available.
    - Define the optimizers (Adam) for both networks.
    - Define the loss function (BCELoss).
    - Set the number of discriminator steps and training epochs.
- **Training helper functions**:

    - Define a function to create real data targets (ones).
    - Define a function to create fake data targets (zeros).
    - Define a function to train the Discriminator network.
    - Define a function to train the Generator network.
- **Training loop**:
    - Initialize a logger for tracking training progress.
    - Iterate through epochs and batches.
    - Train the Discriminator on real and fake data.
    - Train the Generator on fake data.
    - Log errors and display training progress.
    - Save model checkpoints periodically.
- **Testing and visualization**:
    - Generate test noise for sample generation.
    - Generate and log sample images from the generator during training.