Assignment (3)

24789 - Special Topics: Deep Learning for Engineers (Spring 2022)

Out Date: 2023/3/28 (Tue)
Due Date: 2023/4/05 (Wed) @ 11:59 pm EST

All exercises should be submitted to Gradescope under **24-789 Homework 3**, where you will submit a pdf file of your jupyter notebook. You will also need to submit a zip file of your notebook and additional code under **24-789 Homework 3 Programming**. Please remember to restart and run all and collapse your training cell output before submission. Note: Diffusion models are notoriously slow to train. It is highly recommended to use a GPU. Please start early.

Submission file structure

```
andrewID-HW3.zip
                    MNIST_Diffusion.ipynb
                    unet.py
```

You can refer to Python3 tutorial, Numpy documentation and PyTorch documentation while working on this assignment. This tutorial is also very helpful. Any deviations from the submission structure shown above would attract penalty to the assignment score. Please use Pizza for any questions on the assignment.

Total Points (50 points)

PROBLEM 1

Denoising Diffusion Probabilistic Model

In this assignment you will implement a Denoising Diffusion Probabilistic Model (DDPM) for 1s in MNIST data. This model works by taking images, and first adding a small amount of noise until the image is completely noisy. Then, a trainable model, in our case a UNet, is used to learn how to do this noising process in reverse, seen in figure 1. Your implementation should be similar to this paper.

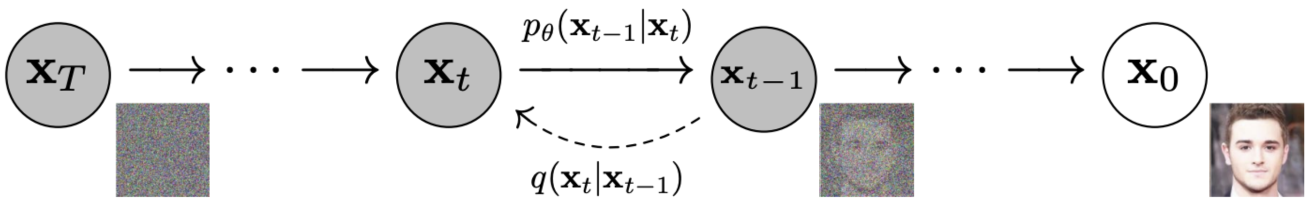
$\mathbf{x}_T$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_t$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$

Fig. 1: Denoising Diffusion Probabilistic Model (DDPM). Image source.

a) Model Initialization 10 points
For this section you will need to implement the `__init__` function in the `MNISTDiffusion` class. This class should contain the β from the provided `_cosine_variance_schedule` function, α derived from β, and initializing the internal UNet model.

b) Forward Diffusion 10 points
Using the reparameterization trick, implement the forward pass of the model:

$$q\left(x_t|x_0\right) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, \left(1-\bar{\alpha}_t\mathbf{I}\right)\right)$$

In order to do this, you will need to implement the `forward` and `_forward_diffusion` functions. You should sample a single t uniformly from your timesteps for each training sample.

c) Reverse Diffusion 10 points
Reverse diffusion will use trainable mean and variance parameters that come from the internal UNet model.

$$p\left(x_{t-1}|x_t\right) = \mathcal{N}\left(x_{t-1}|; \mu_\theta\left(x_t,t\right), \Sigma_\theta\left(x_t,t\right)\right)$$

You will implement this in the `_reverse_diffusion` function.

d) Image Sampling 10 points
Now you will implement the sampling procedure. In order to accomplish this you will need to implement the `sampling` function. This function takes in the number of samples (default is 36 in our case), generates noise, and runs the reverse diffusion process for each sample. You should return the entire trajectory.

e) Model Training 5 points
Now time to put it all together for model training. You will need to carefully select hyperparameters for your model. This includes standard hyperparameters like `learning_rate`, `batch_size`, `epochs`, etc., and also, crucially, `timesteps`. `timesteps` controls how many noising and denoising steps your model takes during the diffusion process. You will also need to select an optimizer, loss function, and learning rate scheduler. A relatively large value is recommended, on the order of 10^4. You should display 36 final samples using the provided code.

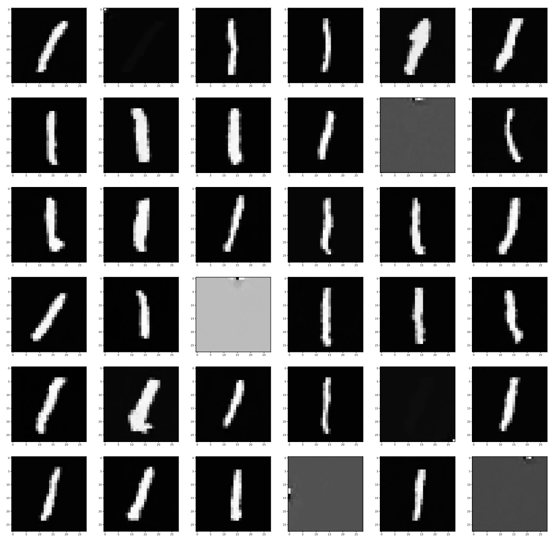

Fig. 2: An example of diffusion output that would receive full credit.

Points will be given as follows:

- 10 points (5 bonus) 36 clear samples, at least two numbers (you will need to modify the data loader code)
- 7 points (2 bonus) 36 clear samples
- 5 points >24 clear samples (as in figure 2)
- 3 points >12 clear samples
- 0 points <12 clear samples

f) Diffusion Visualization 5 points
You will also need to visualize a sample diffusion trajectory that shows noise turn into a clean sample, like in figure 3. Make sure to label each image with the corresponding step of the denoising process.

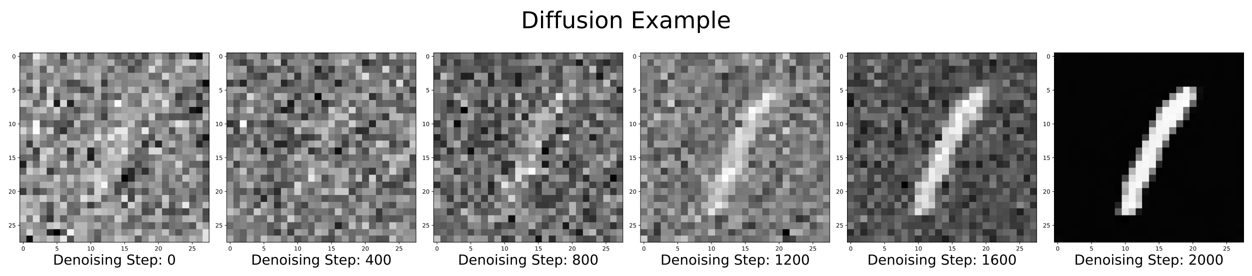

Fig. 3: Diffusion from noise to clean sample.