Assignment #6.1 – Simple diffusion

https://github.com/Lemorita95/1FA006/tree/main/6_diffusion-models/simple-diffusion

Written summary

Also available in https://github.com/Lemorita95/1FA006/blob/main/6_diffusion-models/simple-diffusion/README.md

approach

1. hyperparameters:

    batch_size = 64

    learning_rate = 0.8e-4

    num_epochs = 1000

    diffusion $ \beta_1 $ = 0.02

    diffusion time_steps = 250

2. Define the target data_distribution that we want to predict
3. Sample train and validation dataset from it;
4. Create model DiffusionModel() - used a simple MLP;
5. The model takes 2 inputs: current sample value and the time step and outputs: amount of noise added in timestep t;
6. Loss function is defined as MSE;
7. Training loop:
    1. sample a random timestep (t);
    2. sample a random from from a standard normal distribution (e);
    3. estimate noise (e(xt, t)) for the 'clean' data (x0) at timestep (t);
    4. compute loss between sampled random noise (e) and estimated loss (e(xt, t));
8. Generate new data with sample_reverse();
9. Compare with data_distribution;

results

1. the training loop for the hyperparameters took around 45 seconds;

2. the training had convergence;
3. fast to produce new data for this simple exemple, 100k samples in 3.7 seconds;
4. testes with two diffent distributions and worked well;

challenges

1. high initial loss but reduces fast, not much improvement after 500 epochs;

Result plots

This section present the training loss and the predictions made with the trained network.

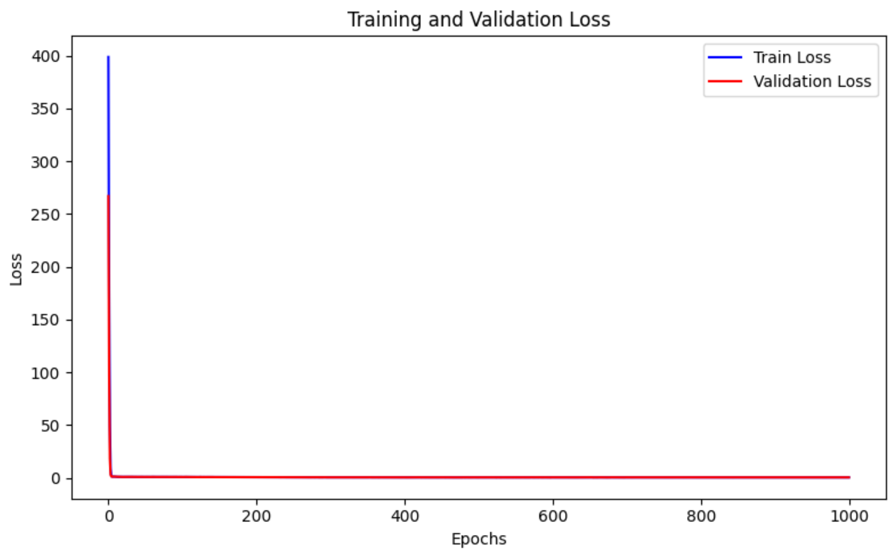

Figure 1: Train and validation loss.

The information on Figure 1 does not highlight the nuances of the training and validation loss throughout epochs, however, the training converged. Similar result was achieved when the training dataset was composed of three gaussians (not shown).

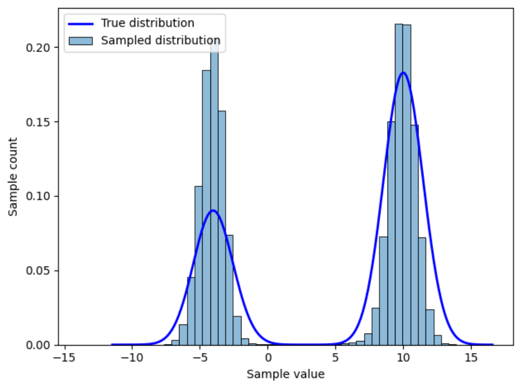

Figure 2: Generated samples for two gaussians dataset.

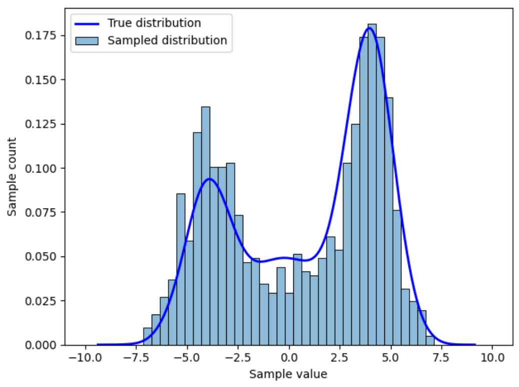

Figure 3: Generated samples for three gaussians dataset.

Figure 2 and Figure 3 shows data the trained network learned to predict the distribution of the training dataset.