EE5179 : Deep Learning for Imaging

Programming Assignment 5: Variational Autoencoders

Due Date: 23rd October, 2025

Instructions

1. Program in python for this assignment.
2. Post any doubts you have on moodle. This will be helpful to your peers as well.
3. Submit the codes (the actual notebook and a PDF version) and your assignment report in a zip file titled PA5 RollNumber.zip in the submission link provided on moodle.

Preliminaries

1. It is recommended to use Google Colab (as in the tutorials) or Jupyter/iPython notebooks for the assignment. The notebook format is very convenient to work (and evaluate!) with and additionally, Colab provides GPU access as well.
2. The dataset can be downloaded from here or you can make use of the inbuilt dataset from Pytorch.

Note: Check if the labels are in one-hot format, or appropriately convert them to one-hot format before training and testing the network.

---

1 Variational Autoencoders

A variational autoencoder (VAE) generates data from the desired distribution of ε. Below is the design of VAE and loss function to be used for training. The first term in the loss function is the reconstruction loss ensuring the input and generated images are the same. The second term ensures the abstract representation of encoder follows the desired distribution using KL divergence loss.

$$\mathcal{L} = -\mathbb{E}_{z\sim q(z|x)}[\log p_\theta(x|z)] + D_{KL}(q_\phi(z|x) \,\|\, p_\theta(z))$$

1. Implement VAEs on MNIST:
   - Use latent dimension 20.
   - Train three separate VAEs with only fully connected layers, each with two hidden layers: $(100, 100)$ for the first, $(200, 200)$ for the second, and $(300, 300)$ for the third. Train the models for 25 epochs.
   - Plot the loss and visualize 15 generated images from random samples for each of the three models.

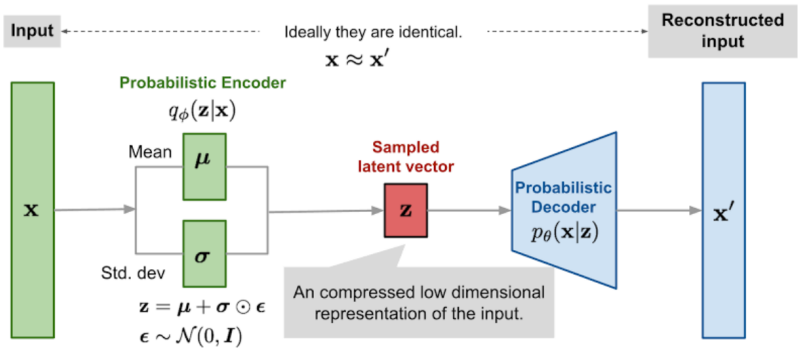

Figure 1: Structure of variational auto encoder

2. Design a convolutional VAE with two convolution layers and two fully connected layers.

   - Input (28×28 grayscale) → Conv1 (32 filters, 4×4, stride 2, padding 1) → Conv2 (64 filters, 4×4, stride 2, padding 1) → Flatten ($64 \times 7 \times 7$) → FC1 (256) → FC2 (128) → FC_μ (latent dim), FC_{σ^2} (latent dim) [Encoded Representation] → FC3 (128) → FC4 (256) → FC5 ($64 \times 7 \times 7$) → Deconv1 (32 filters, 4×4, stride 2, padding 1) → Deconv2 (1 filter, 4×4, stride 2, padding 1) → Output (28×28).

3. Save the trained models. Sample 15 random latent vectors from a standard Gaussian, decode them, and display the reconstructed images.

4. Sample latent vectors from the prior $\mathcal{N}(0, I)$ and decode them using the trained convolutional VAE.

   - Classify the decoded images using a pretrained classifier and visualize them in 2D using t-SNE, coloring each point by the predicted digit.
   - Select two latent vectors corresponding to different digit classes (e.g; digits 4 and 7), perform linear interpolation between them in latent space, decode the interpolated vectors, and display the resulting images in sequence.
   - Write short observations on the latent space structure and how digits change during interpolation.