Report: AI assignment 3 (part 2)

Component 1: Variational Auto Encoder
The VAE class is a Convolutional VAE, its structure is organized into three main parts: the encoder, reparameterization trick, and decoder, and lastly the forward pass

The encoder maps the input image to a latent representation, parameterized by a mean (mu) and log variance (logvar). The convolutional layer comprises 2 CNN layers, CNN1 has 16 filters, and CNN2 has 32 filters, both with a stride of 2 for downsampling. The output of the convolutions is flattened and then passed through two fully connected layers to further reduce dimensionality and prepare the data for generating the latent representation. These layers output the mean (mu) and log variance (logvar) for the latent space, defining the parameters for a normal distribution.

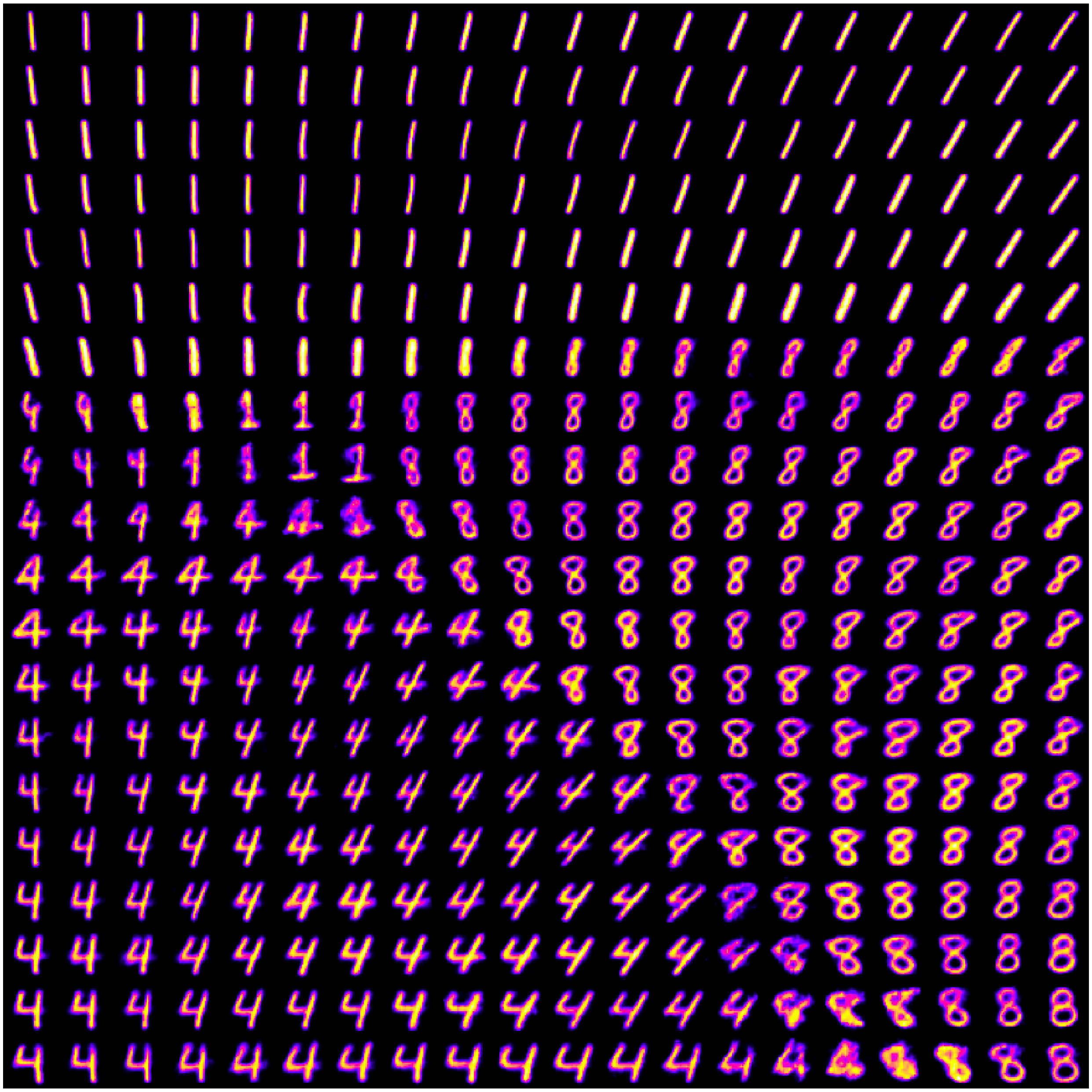
The Reparameterization trick suggests backpropagation through the stochastic sampling process. It takes the mean (mu) and log variance (logvar) and generates a sample from the latent distribution. It calculates the standard deviation from logvar, samples a random noise vector (eps), and combines these to obtain z, a sample from the Normal distribution.

The decoder reconstructs the input from the latent sample to create an output image that resembles the input. The latent variable z is passed through a series of fully connected layers to increase dimensionality and reshape it to match the structure required for the transposed convolutions. These layers upsample the output back to the original image dimensions. The final layer has one output channel (matching the grayscale input) and uses a sigmoid activation to produce pixel values between 0 and 1.

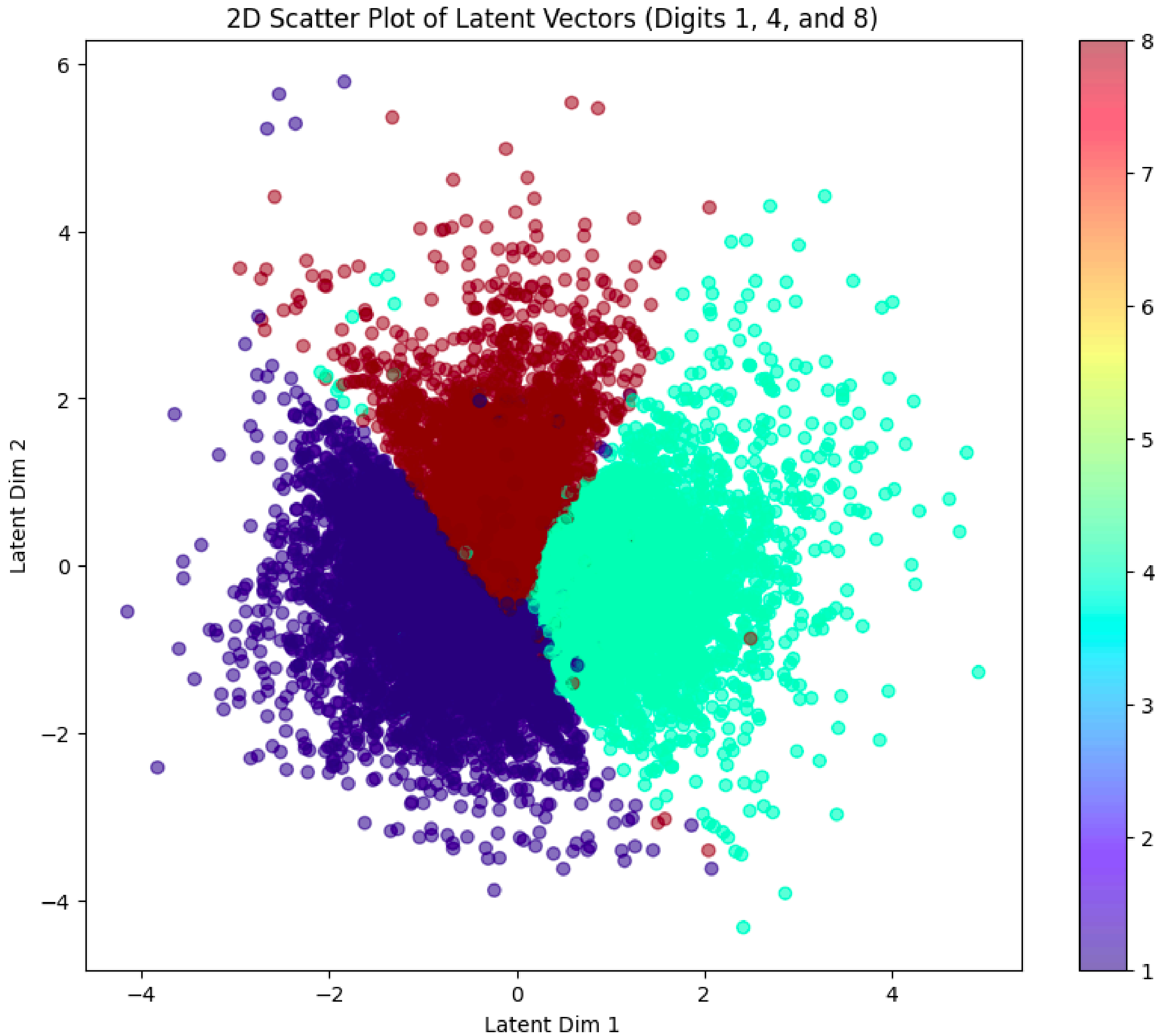
In forward pass, the input is encoded to produce mu and logvar, sampled to generate z, and decoded to produce the reconstructed image. The model returns the reconstructed image, along with mu and logvar for calculating the VAE loss.

Reconstructed Images:

Generating new images:



Visualizing Latent Space:



Component 2: Gaussian Mixture Models:

The Gaussian Mixture Model (GMM) is a probabilistic model that assumes data is generated from a mixture of several Gaussian distributions, each representing a different cluster or component in the data. Each Gaussian distribution has its own mean and covariance, which help define the location, spread, and orientation of each cluster. GMM is widely used for clustering and density estimation. The model employs the Expectation-Maximization (EM) algorithm to iteratively estimate the parameters (means, covariances, and weights) that maximize the likelihood of the data.

Visualizing GMM model