Neural Networks

Project 4:

Understanding Variational Autoencoders

Mario Coronado Fernandez - 100496637
Jorge Chamorro Pedrosa - 100496527

1. 3D GMM Synthetical Data Generation

Following the statement, we first defined a 3-dimensional Gaussian Mixture Model (GMM) data generation function. This algorithm creates an array of number of dimensions by number of components.

Then, we defined for each component a mean that positions the centre of the Gaussian (G) in the 3-dimensional space, a covariance to have variance = 1 and covariance = 0 in all components, resulting in the mutual independence of the components and defining the shape and the spread of the G. Additionally, due to the fact that we use the identity matrix, G will be spherical, and a weight – probability to be selected for a data entry – to make them all sum 1. We used 10 as the number of feature components and 1000 as the number of samples and got a dataset with some separated and overlapping clusters.

This first plot shows some of the points' original clustering, and it may possibly represent the different components of the mixture.

Lastly, we divide the data using batches of 96 samples at once using DataLoader of pytorch, and we shuffle it in each epoch in order to avoid the Neural Network from learning any unwanted ordering for better convergence.

2. Replacing CNNs with Dense Layers

The Variational Autoencoder (VAE) using Dense Layers (DL) instead of Convolutional Neural Networks (CNN), is a choice based on the structure of the data. We are focused on capturing the internal distribution and correlations of the data at a global level, not assuming any spatial correlations, treating each input dimension independently, but with full connectivity to the subsequent layer.

Our VAE's forward method uses the encoder to obtain the mean, variance, and a sample of the latent space, then passes the sample through the decoder to generate the reconstructed output. We ran the VAE's extended function, fixing the channels = 3 (input dimension), dimz = 2, returning a compressed representation better for visualization and representation, with mu and var the learning rate for Adam's optimizer to lr = 1e-3, providing balance between convergence speed and stability, and finally, epochs = 100, allowing the model to adjust its weights iteratively to minimize the loss function.

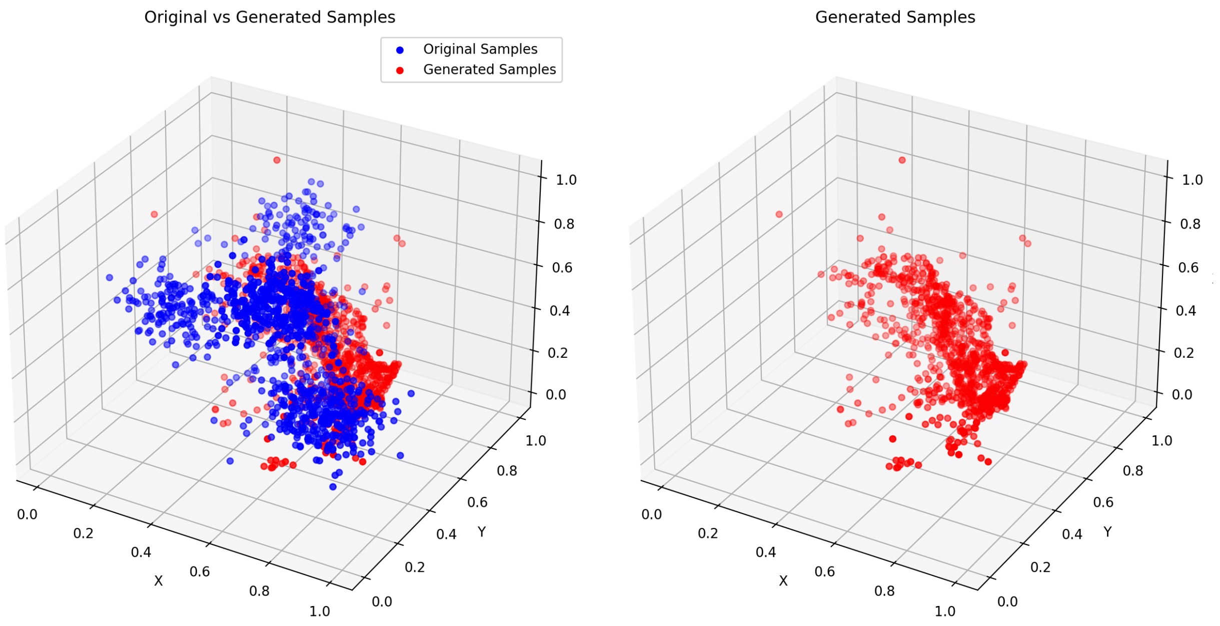
Training Epoch	1	25	50	75	100
Loss	*0.314280*	*0.052333*	*0.051693*	*0.051859*	*0.051494*

The fast drop of the loss values (in epoch #1, loss value already 0.314280) indicates a fast learning rate the model. The ending flattening of the learning curve suggests that the model is approaching a point of convergence where further training results will reduce improvements.

3. Comparison: VAE samples vs ground truth distribution

For comparison, we generate 1000 random from a standard Gaussian distribution vectors from the trained VAE, and then we apply scaling independently for each column.

This way we are going to be able to properly visualize how well our VAE model has learned to mimic the original data distribution. Ideally, the generated samples should cover the same space as the original data and exhibit similar clustering or dispersion characteristics.



As shown in the figures, we can observe that the generated samples approximate the distribution of the original samples. However, some differences are visible:

- The generated samples tend to concentrate more densely around certain regions, indicating that some modes of the original distribution are captured well, while others are missing.

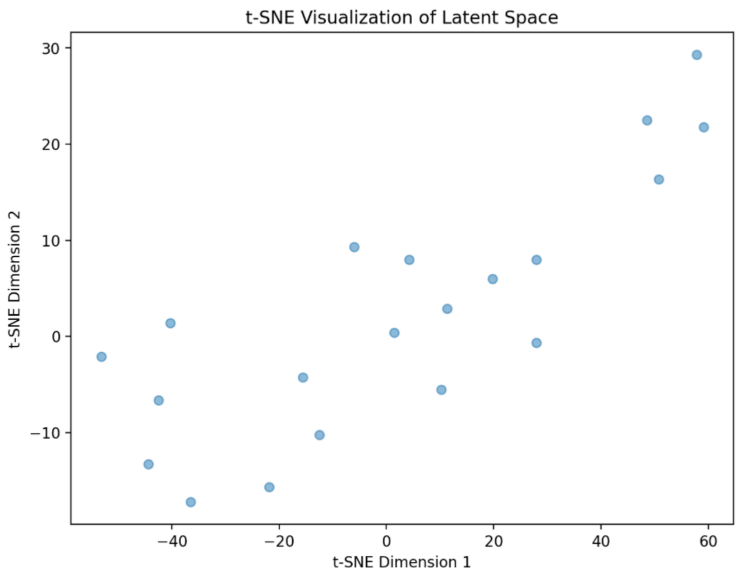
- The spread and variability of the generated samples are lower compared to the original ones, this suggests that the VAE has learned an approximation of the underlying distribution but with some degree of mode collapse.

- Overall, although the generated data does not fully replicate the original GMM, it manages to replicate its general structure and clustering behavior.

After applying clustering (with the elbow method), we observe that the VAE effectively captures around 3 to 5 components, depending on whether the data is scaled or not before visualization. Therefore, the VAE misses several modes, merging multiple original components into fewer regions in the latent space.

4. Using t-SNE for Visualization

We finally performed a t-SNE visualization of the latent space. This is a useful approach to inspect how the VAE represents and organizes data in a 2-dimension space.



The t-SNE plot shows several well-separated points in the latent space, we could even differentiate those 3 clusters. However, the points are isolated without forming dense, compact clusters typical of multiple samples per component.

Given that the original GMM had 10 components, the presence of around 20 points indicates some over-separation. This happens because t-SNE sometimes exaggerates distances between samples when the number of latent samples per mode is low.