DGM final project - VAE with latent clustering using GMM

Shilo Avital, Sarel Lieberman

April 27, 2025

Abstract

In this project, we attempted to cluster the MNIST digits in the latent space, and possibly generate a specific digit by sampling from a specific cluster. The aforementioned (in the project proposal) method to do so, to use a GMM prior instead of the standard gaussian, failed and will be mentioned in the methods section. We then tried various methods and even achieved very nice results using one of them. The full code is available at our GitHub repository.

1 Paper Summary

The Variational Autoencoder (VAE) is a generative model that learns a low-dimensional latent representation of data by combining principles from deep learning and variational inference. The core idea is to approximate the intractable posterior distribution over latent variables using a neural network (called the encoder), which maps inputs to a distribution over the latent space.

To allow backpropagation through stochastic nodes, the paper introduces the *reparameterization trick*: instead of sampling directly from the posterior $q_\phi(z \mid x)$, the model samples from a fixed distribution and applies a deterministic transformation.

The model is trained by maximizing a variational lower bound (ELBO) on the log-likelihood of the data (i.e. minimizing the loss function $\mathcal{L}$):

$$\mathcal{L}(\theta, \phi; x) = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)] + \mathrm{KL}(q_\phi(z \mid x) \parallel p(z))$$

The first term encourages accurate reconstruction of data, and the second term regularizes the latent distribution to stay close to the prior (typically a standard Gaussian).

VAEs offer a principled framework for probabilistic generative modeling with efficient training and inference using standard stochastic gradient descent.

2 Introduction

In a regular VAE, because of the normalization, the data is centered but very mixed. Different digits can appear very close to each other, even if they are very different logically and even pixel-wise. In our project we tried to separate the different clusters in the latent space, by using different models that will learn to project different digits to different areas in the latent space. The connecting thread between all the methods we tried is that we wanted to learn a μ and a σ for each cluster, so we will be able to sample a specific digit in the inference phase.

3 Methods

3.1 Attempt I: Use GMM as the New Prior

In our first approach, we attempted to replace the standard Gaussian prior in the VAE with a Gaussian Mixture Model (GMM) prior. The goal was to structure the latent space such that different clusters would correspond to different digits, enabling better separation and potentially controlled generation. After training the model, we sampled from the latent representations modeled as a GMM and attempted to sample from individual mixture components in order to generate specific digits. This method served as our initial attempt to impose cluster structure in the latent space.

However, this approach failed to produce meaningful clusters. Retroactively, we believe the reason is that the KL regularization term of GMM prior does not encforce good cluster seperation. In other words, the GMM regularization is not strong enough to prevent the gaussians from overlapping.

As a result, and after numerous trials and modifications to the model, we decided to try a different approach to the problem.

3.2 Attempt II: Cluster during training

In this approach, we modified the VAE architecture to explicitly encourage clustering in the latent space during training. We added a trainable matrix $C \in \mathbb{R}^{K \times d}$, representing K cluster centers in the latent space. For each latent embedding z_i produced by the encoder, we computed its soft assignment to each cluster based on Euclidean distance, using the formula:

$$w_{ik} = \frac{\exp(-\|z_i - C_k\|^2)}{\sum_{j=1}^{K} \exp(-\|z_i - C_j\|^2)}$$

We then added a soft K-Means clustering loss to the VAE objective:

$$\mathcal{L}_{\text{cluster}} = \sum_{i=1}^{n} \sum_{k=1}^{K} w_{ik} \|z_i - C_k\|^2$$

The final training loss became:

$$\mathcal{L} = \mathcal{L}_{\text{VAE}} + \lambda \cdot \mathcal{L}_{\text{cluster}}$$

where λ is a hyperparameter controlling the strength of the clustering term.

This method encourages the latent vectors to be softly pulled toward cluster centers throughout training, rather than regularizing with KL of GMM. We also tried to combine this method with the first one to check if the bad results we got was due to the lack of power in the clustering compared to the normalization.

Although promising in theory, this method did not yield clean cluster separations in practice, and not proper digits. We hypothesize that the clustering term might have dominated early in training, leading to poor reconstructions, or that the latent space lacked enough structure initially for meaningful clustering to emerge.

3.3 Attempt III: Mutual training of a regular VAE and a GMM

In this approach, we seperated the Variational Autoencoder (VAE) and the Gaussian Mixture Model (GMM), allowing each to specialize in its role. First, we trained a VAE using the standard Gaussian prior to obtain meaningful latent representations. Afterward, we trained a GMM on the VAE's latent embeddings to cluster the data.

This two-step pipeline proved remarkably effective even after a single iteration. Once the GMM was trained, we used its component means and variances as a new prior for the VAE in subsequent iterations. Specifically, for each data point, we assigned the corresponding cluster's mean and variance as the prior parameters in the KL divergence term. This created a form of mutual learning, where the VAE gradually aligned its latent space with the GMM clusters, and the GMM benefited from progressively improved representations.

The result was a well-structured latent space with clear digit-wise clusters. Sampling from specific Gaussian components allowed us to reliably generate digits of a desired class, demonstrating that the model successfully learned a semantically meaningful latent space. This method yielded the best clustering and generation performance among all the methods we explored.

4 Results

Although the first two methods were not checked thoroughly, because the third method yielded great results, we did not explore further to find out the reason they did not work.

Our final method, mutual training a standard VAE and a GMM, yielded compelling results. As shown in Figure 4, the latent representations formed well-separated clusters corresponding to digit classes. The GMM achieved high clustering accuracy (over 80%).

Moreover, sampling from individual Gaussian components led to coherent and class-consistent digit generation. For instance, Figure 1 shows samples generated from each single GMM component corresponding a digit with almost no variation across samples — confirming that each cluster captures a semantically meaningful region of latent space.

We also observed a notable improvement in the ELBO over training (Figure 2), demonstrating that our mutual learning setup remains effective even when integrating external cluster structure into the latent prior. The final t-SNE projections in Figure 4 highlight how the latent space evolved into a well-clustered layout.

(Figure 3) shows how each inference sample is made. Around every mean we sample from a lower variance gaussian so that the samples will not likely come from another cluster. With that approach we get different enough samples but samples that are close enough to the digit they are supposed to represent.

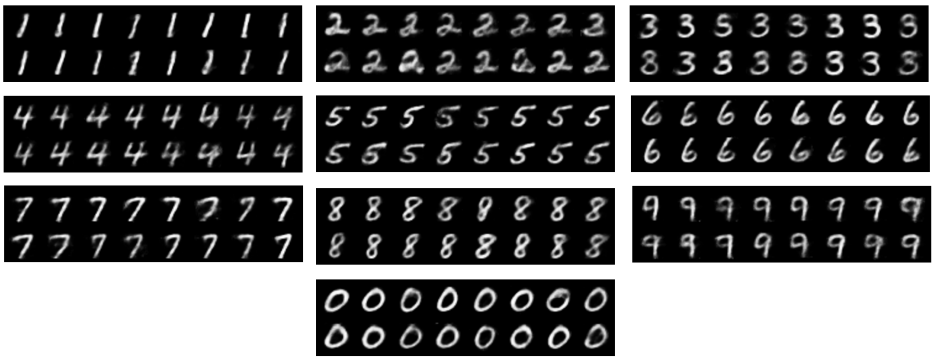

Figure 1: Generated samples from a all GMM clusters. All samples resemble one another in each cluster and represent the same digit.

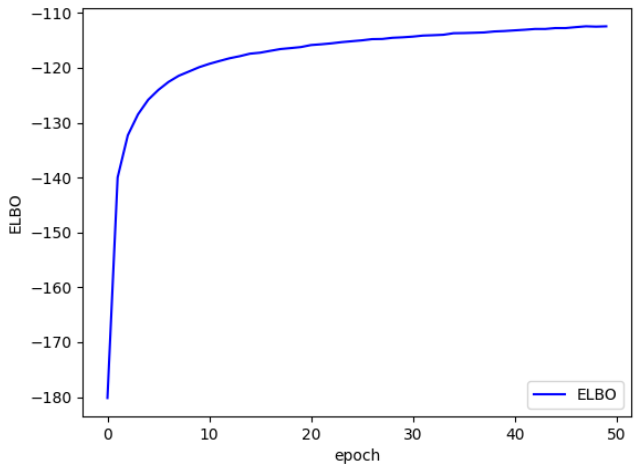

Figure 2: ELBO during training. The plot shows a monotonous increase in the ELBO with each epoch. the values were taken from the last iteration over the VAE.

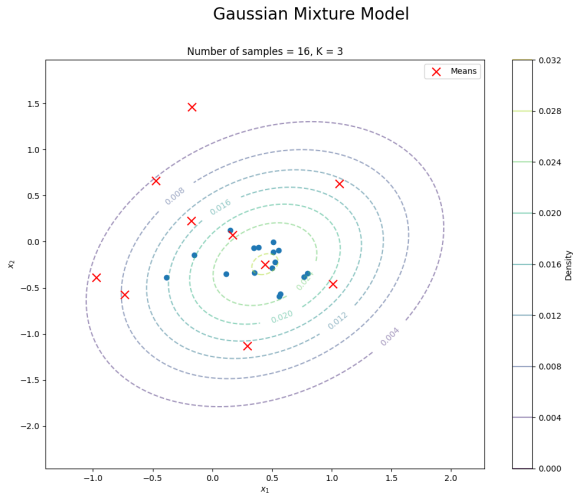

Figure 3: The cluster means scattered around the latent space. Each sample is taken from a reduced variance gaussian around the means we found after the last GMM iteration.

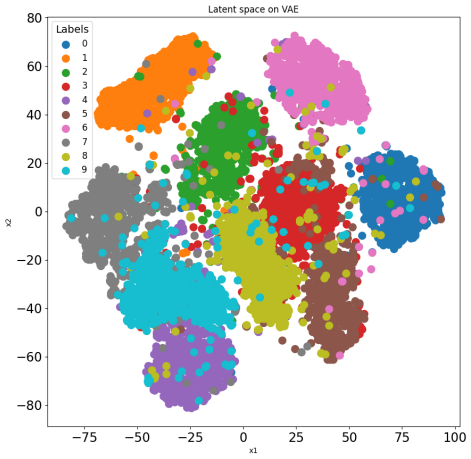

Figure 4: t-SNE visualization of latent space. Clear separation emerges between different classes.