Structured Latent Space in Variational Autoencoders (VAEs) - Task 2 Documentation

Akshay Shukla (2022A7PS0087P) Deepan Roy (2022A7PS0004P)

April 2025

0.1 Introduction

Through this project, we aim to build a custom variational autoencoder (VAE) architecture and successfully train it to create a well-structured latent space with disentangled dimensions for content and style. We will sample from this latent space to generate diverse and meaningful images. For this purpose, we build upon the foundational framework of Adversarial Autoencoders (AAEs), modifying and extending it to suit the specific requirements of Task 1 - designing a Variational Autoencoder (VAE) with a structured and semantically meaningful latent space.

0.2 Adversarial Autoencoders

The Adversarial Autoencoders (AAE) paper by Makhzani et al. (2015) introduces a novel approach to regularizing the latent space of autoencoders using adversarial training instead of the traditional Kullback-Leibler (KL) divergence employed in Variational Autoencoders (VAEs). This is in accordance with the problem statement given to us making this paper a very suitable fit.

In this framework, an autoencoder is trained to reconstruct input data while its encoder simultaneously learns to produce latent codes that match a target prior distribution (typically Gaussian). This is achieved by incorporating a discriminator that distinguishes between samples from the true prior and the encoder's output, while the encoder tries to fool the discriminator - similar to how a generator operates in GANs.

The method allows for more flexible and interpretable latent representations and can be extended to include label supervision for tasks such as semi-supervised learning or clustering. By blending the generative strengths of VAEs with the adversarial mechanisms of GANs, AAEs offer a powerful alternative for enforcing structured and meaningful latent spaces.

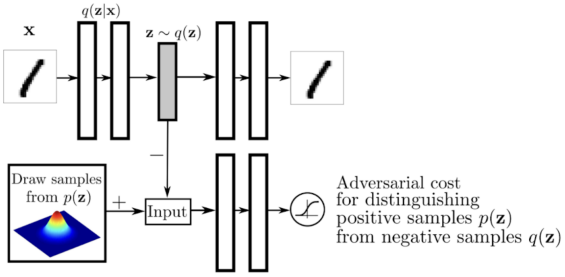

Figure 1: Architecture of an adversarial autoencoder.

0.3 Omniglot Dataset

The Omniglot dataset is a widely used benchmark for evaluating models in few-shot learning, generative modeling, and character recognition. It was introduced by Lake et al. in 2015. The dataset contains 1,623 handwritten characters from 50 different alphabets, including real-world and fictional scripts (such as Latin, Cyrillic, Tibetan, and others). Each character class contains 20 samples, drawn from different individuals to ensure stylistic diversity. The images are grayscale and typically resized to 28×28 pixels, making them compatible with many image-based neural networks.

A key feature of Omniglot is that it supports disentanglement of style and content - characters differ in identity (content) and in handwriting (style), which aligns perfectly with tasks like style transfer and latent space manipulation. Additionally, the dataset is often augmented through rotations (90°, 180°, 270°) to simulate new character classes and increase complexity. This makes Omniglot an ideal

choice for structured latent space modeling, as it enables the exploration of both semantic (character identity) and stylistic (writing style) variations.

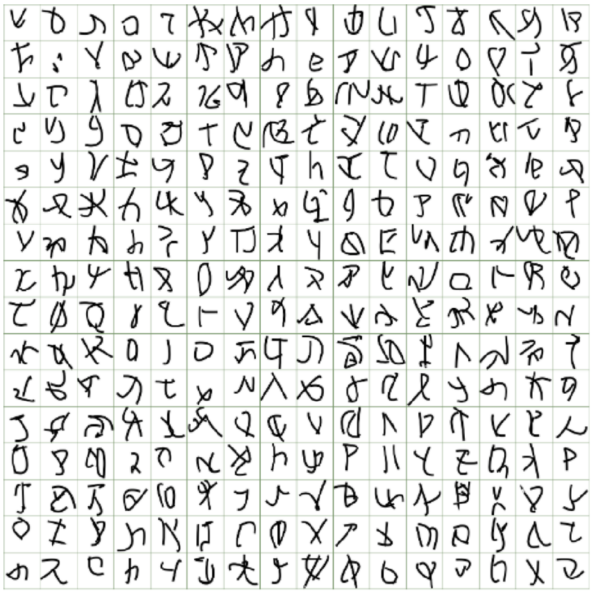

Figure 2: Samples from Omniglot dataset.

0.4 Implementation

Unlike a standard Adversarial Autoencoder which learns a single latent representation and enforces a global prior distribution through a discriminator, our model employs two disentangled latent spaces: z_content and z_style. This helps in separating high-level semantic information (e.g. character identity) from stylistic attributes (e.g. stroke or slant). Each latent space is handled independently throughout the architecture, allowing greater control over the generative process.

The encoder network is designed to process the input images from the Omniglot dataset and produce two separate latent vectors: one encoding content (z_content) and the other encoding style (z_style). These vectors are then passed to a decoder, which reconstructs the image using the concatenated latent representation. The decoder is built using transposed convolutional layers suitable for 28×28 grayscale images, with the objective of accurately reconstructing the original image while preserving content and style information encoded in the respective latent spaces. The decoder's training is guided by a pixel-wise reconstruction loss (Mean Squared Error), ensuring visual fidelity in the output.

To regularize the structure of the latent space and ensure that z_content and z_style align with standard Gaussian priors, our model incorporates two separate adversarial discriminators. These discriminators attempt to distinguish between true samples drawn from a Gaussian distribution and fake samples produced by the encoder. This GAN-like adversarial objective replaces the KL divergence term typically used in VAEs.

To further structure the latent space semantically, we integrate supervised classification heads for both latent vectors. The content classifier attempts to correctly predict character labels from z_content, encouraging it to capture identity-relevant features. This design implicitly encourages z_content to encode information related to class identity and z_style to capture complementary stylistic variations. We explored the possibility of applying a contrastive loss to z_content to enforce intra-class similarity and inter-class separability, but the results obtained were not very promising.

The total generator loss aggregates all these objectives: reconstruction loss, adversarial losses for both latent spaces and the classification losses. We have tuned the weights carefully to ensure no com-

ponent dominates the training process. During evaluation, our implementation includes interpolation between latent vectors, content-style recombination, and semantic traversal across individual latent dimensions - all of which visually demonstrate the model's ability to perform style transfer, continuous latent transitions, and interpretable manipulations.

0.5 Evaluation Results

We evaluated the latent space structure using KL divergence. We can compare our new evaluation results with that of part 1:

Latent Space	KL Divergence	Clustering Accuracy
Previous	63.1432	0.0017
New (content latent space)	**3.8717**	**0.0045**
New (style latent space)	**3.2622**	**0.0043**

Table 1: Comparison of evaluation results between previous and current implementations.

We also generated separate t-SNE plots for the content latent space and style latent space. The plots are as follows:

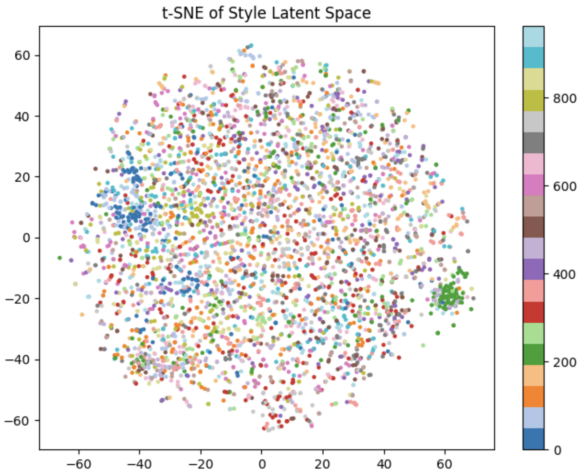

Figure 3: t-SNE of style latent space.

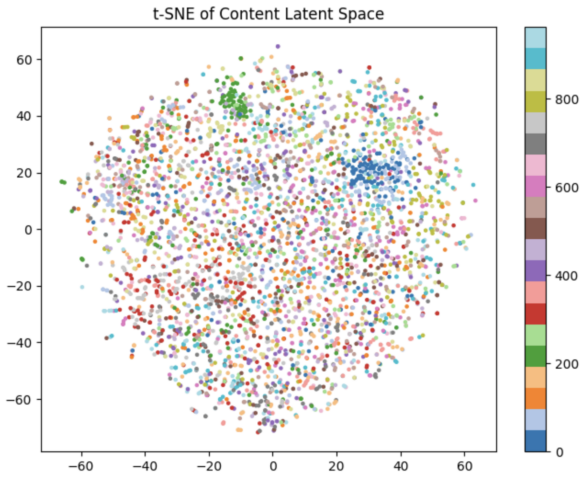

Figure 4: t-SNE of content latent space.

We can compare these results with those of part 1 of this task. The t-SNE plot for this model was as follows:

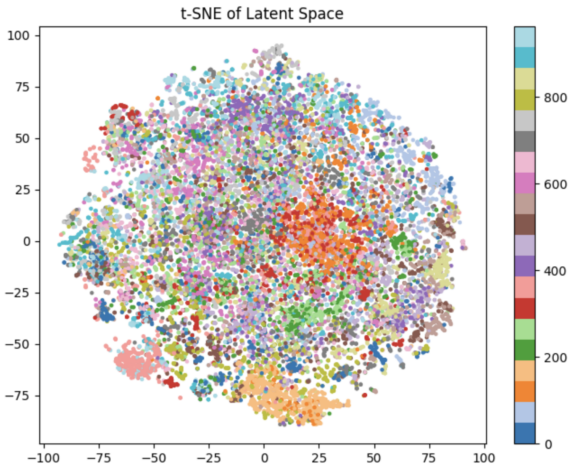

Figure 5: t-SNE of previous latent space.

We can see that the results obtained by our Adversarial Autoencoder model are much better than the results we had obtained with a straight-forward implementation with no discriminator or generator involved.

This can be clearly seen in the quality of the latent interpolations generated by both these models. The latent interpolations generated by the first model are as follows.

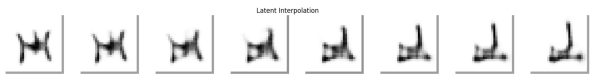

Figure 6: Previous latent interpolation results.

We can see that the interpolations generated are of poor quality and don't do justice to the content

and style features seen in the images of the Omniglot dataset. The latent interpolations generated by the second model are as follows:

Figure 7: New latent interpolation results.

The images generated here are of a much superior quality than those of the first model. They capture the various features of the dataset better and are able to generate new images using those learnings.

0.6 Scope for Future Improvement

While we have successfully managed to build a model with a much superior performance than our previous model, there is still a lot of scope for improvement. The KL Divergence values for both the content and style latent spaces have shown a drastic improvement. However, a similar pattern is not seen with the clustering accuracies of the two latent spaces with there being only a minor improvement in the two. A possible solution that can be explored for this purpose is to use an architecture similar to U-Net wherein we include skip connections and make our encoder and decoder deeper by adding more layers to them. We could also explore the use of contrastive loss to improve the training of the generator which could lead to better clustering.