Exploring the Latent Space of Autoencoders with a Metric of Artistic Style

School of Computer Science & Applied Mathematics
University of the Witwatersrand

William Rich
2151490

Supervised by Dr Richard Klein

October 12, 2021

A research report submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Master of Science in Computer Science by Course Work and Research Report.

Abstract

This research report uses a metric called style loss to quantitatively explore the latent space of autoencoders. Style loss measures the difference in artistic style between two images [Gatys *et al.* 2015]. The goal of the research is to use style loss to compare the latent space of five autoencoder models, in order to determine whether or not style loss is useful in better understanding the latent space in terms of its organisation and regularisation. The five autoencoder models used are an Adversarial Autoencoder (AAE), Triplet Variational Autoencoder (TVAE), β-Variational Autoencoder (β-VAE), Variational Autoencoder (VAE), and Autoencoder (AE). In the experiments a data set of famous artworks is used, and the artistic style (Impressionism, Expressionism, Realism, Romanticism) is used as the class labels.

For each model, all of the artworks are encoded using the encoder networks of the autoencoders. Pairs of points in the latent space are chosen according to various criteria. Then for each pair, a linear trajectory is determined, and 10 equally spaced points are sampled from the trajectory. Each point in the trajectory is then decoded using the decoder network of the respective autoencoder model, to yield a series of images. For each consecutive pair of images in this series the style loss is then evaluated.

Initially experiments are conducted to get a basic understanding of how the latent space of the five different models compare. For this Principal Component Analysis (PCA) is used along with various clustering algorithms. From these experiments it is found that the AAE has spherical covariance in the latent space. The distribution in the latent space for the TVAE is oblong, and the clustering experiments strongly suggest that the data is grouped according to the class labels. For the β-VAE, VAE, and AE the data manifold is also oblong, but there is not compelling evidence to suggest that the data has been clustered according to the class labels.

A major finding in the style loss experiments is that there are two distinct patterns. The first pattern, mainly seen in the AAE and VAE, is that the style loss takes a distinctive horseshoe shape, and the second pattern is that the style loss takes a wave-like shape. By investigating the image reconstructions, it appears that the horseshoe pattern occurs when the image is transformed along the interpolation, and the wave pattern occurs when there is a cross-dissolve between the images. This indicates that style loss can be used to better understand the latent space, as there is a strong correlation between the amount of cross-dissolve and the pattern of style loss along interpolation trajectories.

Declaration

I, William Beyers Alistair Rich, hereby declare the contents of this research proposal to be my own work. This proposal is submitted for the degree of Master of Science in Computer Science at the University of the Witwatersrand. This work has not been submitted to any other university, or for any other degree.

Acknowledgements

I would like to thank my wife and family for their kind love and encouragement throughout my academic career. Without them I would not have had the support to further my research. Dr. Richard Klein has inspired me to become fascinated with computer vision and his guidance and patience will benefit me for many years to come. Dr. Bevan Smith has been an excellent friend and mentor to me. I thank God for his provision and compassion to me as a postgraduate student. This work used computational resources funded in part by the National Research Foundation of South Africa (Grant Number 118075).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

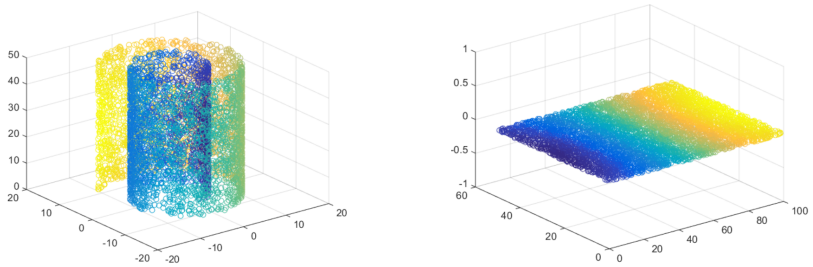
1.1 Autoencoders for Representation Learning

In machine learning, obtaining an appropriate representation of training data is imperative to learn effective models. Typically, training data is first transformed so that the statistical properties of the data are better represented to a specific machine learning algorithm. A major component of this process is feature engineering, where additional features are modelled to better represent discriminative features that may exist. For example, some machine learning algorithms train more efficiently with discrete values, rather than continuous values. A continuous value can be made discrete by binning values into different categories by using a predefined set of categories or intervals [Jin *et al.* 2009]. However, choosing the correct number of intervals may be labour-intensive and may require deep domain knowledge and experience [Shen *et al.* 2020a]. With representation learning, the labour-intensiveness of this can be reduced by automating feature engineering processes.

Representation learning concerns using machine learning techniques to automate the generation of useful features [Bengio *et al.* 2013]. The term *latent space* is key in representation learning. The latent space is a representation of data where the original data structure has been abstracted into a compact representation, such that the dimensionality of the latent space is lower than the dimensionality of the original data. Typically, latent spaces do not perfectly represent the original data distribution as some information is lost when the dimensionality of the data is reduced. However, if the latent space can be manipulated to better represent latent discriminative features that may exist in the original data, the loss of information is a sensible trade-off. Consequently, by training models on such abstracted representations, the performance of various machine learning algorithms can be improved [Vinay *et al.* 2005].

When it comes to understanding the latent space, two important concepts are the data manifold and regularisation in the latent space [Brahma *et al.* 2015]. A data manifold is a data distribution with dimensionality that is embedded in a higher dimensional space. Figure 1.1a shows a Swiss roll data manifold. In this Figure the data is embedded in a 3-dimensional space, but intuitively one can see that the representation of the data can be simplified. The Swiss roll can effectively be unravelled to occupy a region that only exists in two dimensions, which is shown in Figure 1.1b. Brahma *et al.* [2015] argue that in many deep learning models, the models implicitly learn to

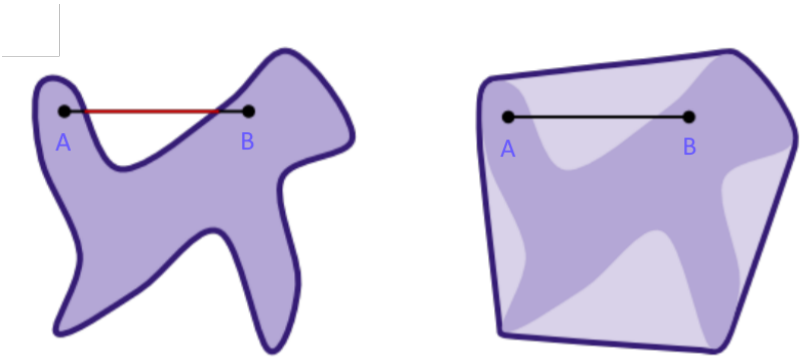
unroll the manifold structure so that the data distribution can more easily be used for analytical tasks.



(a) Swiss Roll Manifold (b) Unravelled Manifold

Figure 1.1: Swiss Roll Manifold Example [Brahma *et al.* 2015].

While the data manifold is an important concept in representation learning, regularisation in the latent space is equally important. Regularisation in the latent space ensures that the data manifold does not overfit the training data, and that the data manifold has favorable properties throughout, such that features in the latent space are generalised. By illustration, suppose that there is a data manifold represented in Figure 1.2a [Sainburg *et al.* 2018]. The enclosed area are points that are on the data manifold. Note that this data manifold is not convex in shape. Points A and B are on the data manifold, but points along the red segment between A and B are not on the data manifold, and so it would not be sensible to sample those points, because their meaning is potentially ambiguous. Ideally the manifold structure should be convex, as in Figure 1.2b. This ensures that points sampled along an interpolated trajectory between A and B are still on the data manifold.



(a) Non-convex Manifold (b) Convex Manifold

Figure 1.2: Non-convex and Convex Data Manifold Example [Sainburg *et al.* 2018].

Because many machine learning models depend on learning appropriate data representations, studying the latent space has received increased attention in recent literature [Patrini *et al.* 2020]. This begs the question as to how the latent space should be evaluated, to ensure optimal feature learning. Autoencoders provide an effective means of studying the latent space. An autoencoder is a neural network that learns an abstract representation of data [Rumelhart *et al.* 1985]. The characteristic trait of these

networks is the way in which the network learns this representation. Autoencoders typically consist of two components, namely an encoder network and a decoder network. A simple representation of an autoencoder is shown in Figure 1.3. With image data, an input image is passed through the encoder of the network. The encoder outputs a vector which is a learned representation of the input image in the latent space. The decoder takes this representation and decodes the latent vector to produce an image, called the reconstructed image. The effectiveness of the autoencoder is primarily measured by the error between the input image and reconstructed image, which is called the reconstruction loss. By training the autoencoder on a data set of images, the data is abstracted into the domain of the latent space, where the data manifold of the images can now be explored.

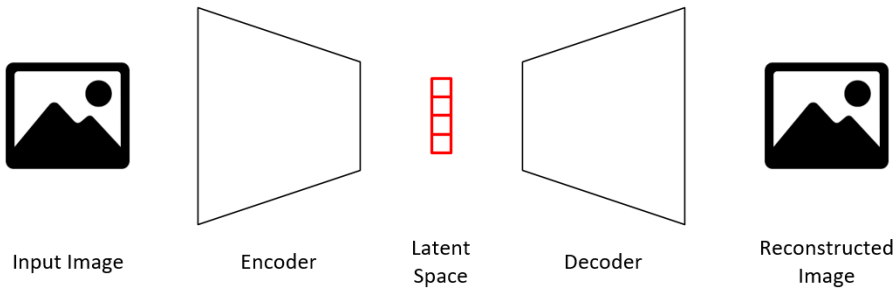


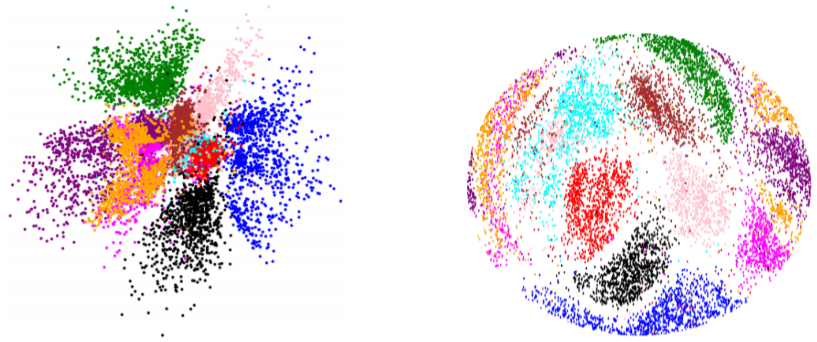
Figure 1.3: Autoencoder Intuition.

1.2 Latent Space Evaluation

Latent space evaluation is the process of investigating how the latent space is organised, in order to ascertain how generalised and useful the learned latent features are for a particular task [Bengio *et al.* 2013]. This is especially important because in many applications of machine learning it is necessary to understand the latent factors that cause the model to have a particular response [Li *et al.* 2021]. Typically, latent space evaluation involves a component of manual inspection and interpretation. An important related work in literature, is that of Liu *et al.* [2019]. Their work proposes a framework of tasks which can be used to explore and understand the latent space. Some of the tasks in their framework include: reconstructing samples from the latent space, examining nearest neighbours in the latent space, visualisation, and interpolation in the latent space. One very popular method of understanding the latent space is by using the t-SNE algorithm [Maaten and Hinton 2008]. The algorithm is used to reduce the dimensionality of a high dimensional space, so that the high dimensional space can be shown on a two or three dimensional plot.

Figure 1.4a and 1.4b show two latent spaces for the MNIST dataset using two different autoencoders, where each colour represents a different digit $(1 - 10)$ [Davidson *et al.* 2018]. This plot compares the latent space of two autoencoders, namely a hyperspherical autoencoder (N-VAE), and a variational autoencoder with a Gaussian

variational posterior (S-VAE). Even though these plots show that the organisation of the data in the latent space for the two models is significantly different, there needs to be a means by which one can make further quantitative measurements and qualitative observations about the latent space to better understand its organisation. From plots such as these one might make observations concerning how close together the points are for each digit, or how many outliers there are for each digit. However, there are a number of questions that cannot be answered from such plots alone. These plots are 2-dimensional representations of latent spaces that are often 32, 64, 128 or more dimensions, and they may give a false impression of how close together points really are.



(a) Latent Space of S-VAE. (b) Latent Space of N-VAE.

Figure 1.4: Comparison of Two Latent Spaces of Autoenoders [Davidson *et al.* 2018]

For generative modelling in computer vision, interpolation is perhaps the most common method of evaluating the latent space [Shen *et al.* 2020b]. Interpolation in the latent space is when a starting point, ending point, and step size are chosen, and points are sampled along a trajectory between the starting and ending points in the latent space, with each sampled point being one step size apart. Collectively the points along the interpolation are referred to as an interpolation trajectory. By using the decoder network in autoencoders, each point in the interpolated trajectory can be decoded to create an image. These reconstructed images can then be compared to each other to assess the latent space and effectiveness of the training procedure.

Figure 1.5: Interpolation Results [Radford *et al.* 2015]

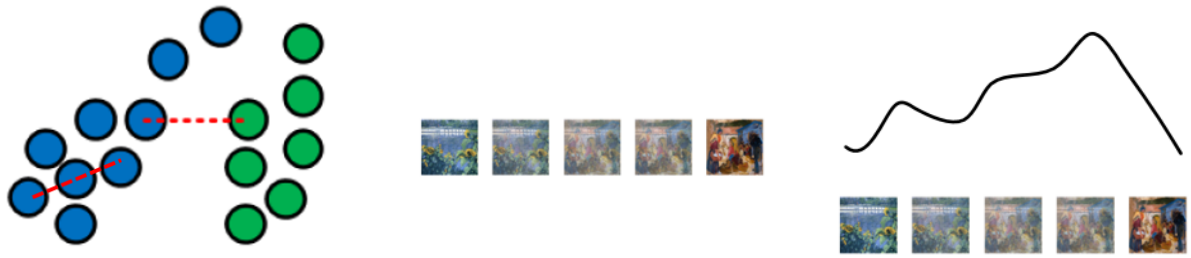
A good example of the use of latent space interpolation is in the work of Radford *et al.* [2015]. Their work uses a data set of images of bedrooms. Figure 1.5 is an example of interpolations using their DCGAN model. This Figure consists of 10 images, where each image is generated from a point that is decoded from an interpolation trajectory in the latent space. From this interpolation we know that the latent space has favourable properties, since each step in the interpolation yielded an image that approximately represents a bedroom, even though the interpolated points may not necessarily

be points in the training data. If the latent space was not regularised, the decoding network would arbitrarily return images that do not represent images that are similar to the training data. Therefore, because the images all look like bedrooms, the data manifold is said to have favourable properties. This illustrates how interpolation can be used to evaluate the latent space.

1.3 Problem Formulation

In light of the importance of representation learning in machine learning, understanding the distribution of data manifolds in the latent space is becoming increasingly important. However, based on the review of contemporary literature, this is often treated as a qualitative task. Viewing reconstructed samples from the latent space, and visualising the latent space (eg. t-SNE) are among the most common qualitative means of doing this [Liu *et al.* 2019]. The purpose of this research report is to investigate the use of a quantitative metric to understand the structure of the latent space. The metric used to achieve this is called *style loss* [Gatys *et al.* 2015].

A data set of famous artworks is chosen to facilitate the chosen metric. Style loss quantitatively measures the difference in the stylistic characteristics between two images. A high style loss indicates that the two images are dissimilar in their stylistic content, and a low style loss indicates that the images are similar in their stylistic content. The exact formulation of style loss is given in Section 2.9.



(a) Interpolation Trajectory. (b) Decoded Samples. (c) Style Loss Plot.

Figure 1.6: Style Loss Experiment Example.

The intuition behind how style loss is used to evaluate the latent space is shown in Figure 1.6. Suppose that there are two clusters in the latent space, where one cluster represents a group of Impressionism artworks, and the other cluster represents a group of Expressionism artworks (Figure 1.6a). One point from each cluster is chosen to be the starting and ending points of the interpolation trajectory (the trajectory is shown in red). Using an autoencoder model, the decoder network is used to reconstruct images from each point in the trajectory (Figure 1.6b). The style loss is measured between consecutive pairs of images, and the style loss is plotted (Figure 1.6c). With plots such as these the research questions can be formalised:

1. **Research Question 1:** Is there evidence that samples have been grouped according to their class label in the latent space?

2. **Research Question 2:** How does the style loss change when interpolating within one class in the latent space (intra-class style loss)?

3. **Research Question 3:** How does the style loss change when interpolating between two classes in the latent space (inter-class style loss)?

1.4 Research Contribution

With the proposed research questions, suitable autoencoder architectures must be chosen. This report answers the research questions by considering five different autoencoder models. The experiments therefore also provide insight on how the latent spaces of the five models compare to each other. The five models are:

1. **Autoencoder (AE).** This is the most basic autoencoder model, similar to that in Figure 1.3 [Wang *et al.* 2014].

2. **Variational Autoencoder (VAE).** This follows a similar architecture to the AE, except that an element of stochasticity is brought in to the latent space of the network, which aids in regularising the latent space [Kingma and Welling 2014]. A stochastic term is introduced in the network loss function, and is measured with the Kullback–Leibler divergence.

3. **β-Variational Autoencoder (β-VAE).** This is the same architecture as the Variational Autoencoder, except the Kullback-Liebler divergence term in the loss function is scaled by a factor of β; $0 < \beta < 1$ [Higgins *et al.* 2016].

4. **Triplet Variational Autoencoder (TVAE).** A third term is introduced to the loss function of the β-VAE. This term, called triplet loss, encourages that samples of different classes are at least a distance m from each other, while samples from the same class are at least a distance m close together [Hoffer and Ailon 2015].

5. **Adversarial Autoencoder (AAE).** This model is significantly different to the others. The basic autoencoder architecture is maintained, but a discriminator network is introduced, to discern between samples in the latent space, and samples from a Gaussian distribution. This introduces an adversarial loss term into the autoencoder loss function, that encourages the data manifold to follow a Gaussian shape [Makhzani *et al.* 2015].

The contribution of this research report is outlined in the following three points:

1. Each of the five models are trained on the data set of artworks.

2. The latent space is explored using Principal Component Analysis and clustering methods to compare the latent space of the different models.

3. Style loss is used in intra-class and inter-class interpolation experiments to compare the latent space of the different models.

1.5 Document Overview

The remainder of the report is structured as follows. The background of important technical subjects pertaining to the research is provided in Chapter 2. Following this, related research in contemporary literature is reviewed in Chapter 3. In Chapter 4,the research methodology provides detail on the data set, models, and design of experiments. Thereafter Chapter 5 presents how the models are implemented, with some examples from the models. The results from the research are presented in two Chapters. Chapter 6 investigates the organisation and structure of the latent space, by means of PCA and clustering in the latent space. In Chapter 7 a series of style loss experiments are conducted by interpolating in the latent space. The report ends with a conclusion in Chapter 8, where a summary of the research methodology is provided, the key insights are listed, as well as the assumptions, limitations, and suggestions for further work.

Chapter 2

Background

2.1 Introduction

In this Chapter the background of technical subjects related to the research topic is presented. In Sections 2.2, 2.3, 2.4, 2.5, 2.6 an overview of the major learning architectures relevant to the research is presented. Thereafter, in Sections 2.7 the discussion focuses on the latent space and its interpretability. Section 2.8 presents triplet loss, which is a learning mechanism that encourages better organisation of the latent space. Finally, a major topic of interest in this proposal concerns Neural Style Transfer, which is discussed in Section 2.9.

2.2 Autoencoders

An autoencoder is a neural network that is used in unsupervised learning. The network is characterised by the output layer of the network which reconstructs the data that is passed to the input layer [Wang *et al.* 2014]. Rumelhart *et al.* [1985] proposed this architecture in 1985, and since then there have been various developments such as convolutional autoencoders (Section 2.3) and variational autoencoders (Section 2.4).

Autoencoder networks can be considered in two components, namely an *encoder network*, and a *decoder network*. Autoencoders typically have a conceptual architecture similar to that shown in Figure 1.3. The input data is a vector with m-dimensions, and similarly the output from the last layer is a vector with m-dimensions. The encoder network takes the m-dimensional input and learns a latent representation of the data such that the dimensionality of the latent space (l) is lower than the input dimensionality (m). Hence the encoder network has encoded the data to a domain with fewer dimensions than the original data domain. It can be said that the input data has been *encoded*. Conversely, the decoder network reverses this process. The decoder takes the latent representation of the data (the encoded data) with l-dimensions and decodes this to a domain with the same number of dimensions as the input data.

The process of training autoencoders consists of the following steps [Chen *et al.* 2017]:

1. Encode the input vector with m-dimensionality with a forward pass with the encoder portion of the autoencoder, creating an l-dimensional representation.

2. Decode the l-dimensional representation with a forward pass of the decoder network resulting in a m-dimensional output.

3. Calculate the loss, called the reconstruction loss, as the mean-squared error between the input vector and the output vector.

4. Perform a back-propagation step to update the network weights.

5. Training ends when the mean-squared error is sufficiently small.

One significant use of encoding data to a lower dimensional space is that it allows the extraction of latent feature vectors that are learnt by the autoencoder. The latent space can be compared to the principal components that are obtained from principal component analysis (PCA) [Almotiri *et al.* 2017]. A key difference between PCA and the latent space of autoencoders is that PCA is limited to linear transformations of the input features, whereas autoencoders introduce non-linear feature learning because of the non-linear activation function used in the neurons of the neural network. The significance of this latent representation is that most of the information and data structure contained in the input data (m-dimensions) is maintained in the l-dimensional representation. The benefit of this is that the latent representation of the data can be used in other tasks, with the advantage of having the same information represented as the original data, but in a lower dimensional space.

2.3 Convolutional Autoencoders

The same architecture used in autoencoders can be extended to cater for spatial data such as image data. With autoencoders the network weights between the neurons in subsequent neural layers are all fully-connected; that is each neuron in each layer has a connection to every neuron in the succeeding layer (except for the output layer). For convolutional autoencoders the dense layers are replaced with convolutional layers of various sizes such that meaningful spatial patterns can be recognised in the input data [Knyaz *et al.* 2017]. In this case the convolutional layers respond to spatial features that exist in the input image; hence the learning in the network is not only by each network weight separately, but by the feature maps in the convolutional layers collectively. The ability to learn these feature maps has given rise to the rapid development of computer vision applications in recent years.

Convolutional autoencoders also have both encoder and decoder networks. The feature maps that convolutional autoencoders learn are also a latent representation of the input data. Just as the autoencoders in Section 2.2 learn latent representations of the input data, similarly convolutional autoencoders learn spatial feature maps that represent latent spatial features of input images. This is different to the latent representation of linear autoencoders, as the latent representation of the data is abstract and in many cases not interpretable. With convolutional autoencoders the latent feature maps can be explored and understood by visualising each feature map as a 2-dimensional plot. By plotting these features it is possible to visually understand what spatial features the network has deemed to be important in encoding and decoding the input data.

2.4 Variational Autoencoders (VAEs)

A particular characteristic of autoencoders is that they cannot be used to generate new data. Autoencoders merely reconstruct the data that is passed to them as an input. If the model could be modified to include an element of stochasticity then new data could be generated using the latent space and the decoder network. This is what variational autoencoders achieve [Chen *et al.* 2016].

Rather than the model learning a single linear vector in the latent space, a Gaussian distribution parameterised by a mean (μ) and standard deviation (σ) is learnt from which new data can be sampled. This allows the decoder to sample from this distribution so that new images can be created. As such, variational autoencoders are generative models, because they allow new data to be generated, by sampling from the learned distribution.

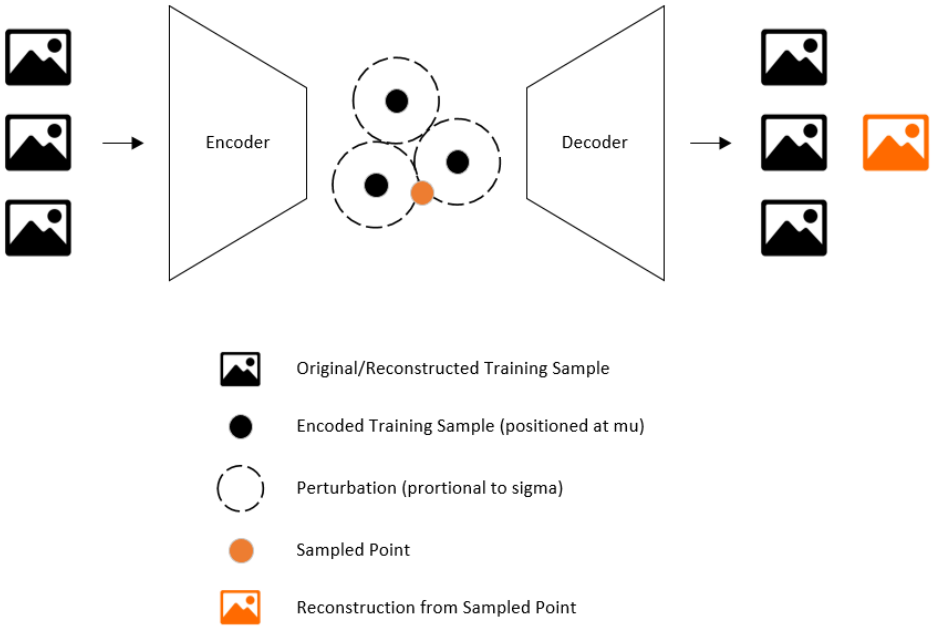


Figure 2.1: VAEs and Generative Modelling.

Figure 2.1 shows how VAEs can be used as generative models. The encoder encodes the training samples to exist on the data manifold in the latent space, and the decoder decoder decodes these points to reconstruct the original training samples. However, each training point is perturbed by a small amount each time a forward pass is done on the network. The position of the point is determined by the learned mean vector (μ), and the magnitude and direction of the perturbation are proportional to the learned standard deviation. Because the perturbation is different on each forward pass, the area around the training sample in the latent space becomes regularised, and consequently each training sample is mapped to a small area in the latent space, which is approximately Gaussian distributed. New samples can be drawn from the latent space because of this regularisation, because areas between training samples on the data manifold are now more likely to look like the original data when passed through the decoder network.

With convolutional autoencoders, the loss term, called reconstruction loss is the mean-squared error between the input data and the network reconstruction. With variational autoencoders, a second loss term is added.

$$\mathcal{L} = \mathbb{E}_{z\sim q_\phi(z|x)}[log(p_\theta(x|z)] - D_{KL}(q_{\phi(z|x)}||p_\theta(z)) \tag{2.1}$$

In equation 2.1 the first term is the reconstruction loss, which is the same as for convolutional autoencoders. This term is the log likelihood of the decoder reconstruction, where p is the decoder network, θ is the decoder network's weights and biases, for a sample x, that is encoded as z in the latent space, by encoding function q with weights and biases ϕ The second term is the Kullback–Leibler divergence of the prior p and posterior q, which is the relative entropy between the two functions [Chen *et al.* 2016].

2.5 β-VAE

Higgins *et al.* [2016] extend the variational autoencoder architecture by introducing a certain parameter β into the model. The β term is a factor that balances the trade-off between reconstruction loss and regularisation in the latent space. The loss function is formalised as follows:

$$\mathcal{L} = \mathbb{E}_{z\sim q_\phi(z|x)}[log(p_\theta(x|z)] - \beta(D_{KL}(q_{\phi(z|x)}||p_\theta(z))) \tag{2.2}$$

The first term in equation 2.2 relates to the reconstruction loss of the autoencoder, and the second term is the Kullback–Leibler divergence that is introduced from sampling in the latent space (see Section 2.4). The authors argue that the Kullback–Leibler divergence may have an overbearing effect, resulting in reconstructions that are not accurate. Therefore β is effectively used to scale the Kullback–Leibler divergence, resulting in a better reconstructions at the cost of a *less*-regularised latent space. β typically takes on a value between 0.5 and 10, but can also be as large as 250, where a larger β tends to better disentangle latent factors, but a smaller β tends to improve the quality of the reconstructions [Higgins *et al.* 2016].

2.6 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are another class of generative model. GANs have some similarities with variational autoencoders but they are fundamentally different because GANs learn by a process of *adversarial training* [Creswell *et al.* 2018].

The intuition behind adversarial training can be illustrated with a simple analogy. Suppose some counterfeiters are trying to create counterfeit bank notes. There are some police who try to distinguish the counterfeit notes from real ones. The counterfeiters try to get better and better at making counterfeit notes but the police try to get better and better at detecting the counterfeits. Perhaps initially the counterfeiters are not so good at fooling the police. But as they are caught, they learn from their mistakes and so they become better at making counterfeits. The police realise that they do not detect the counterfeit money every time, and so they improve the way in

which they discern the counterfeits from real notes. The result is that the counterfeiters become very good at creating counterfeits, the police become very good at detecting counterfeits, and to the naked eye it would be impossible to tell the difference between a counterfeit and a real note. Although this explanation is naive, it intuitively captures the operation of GANs [Goodfellow 2016].

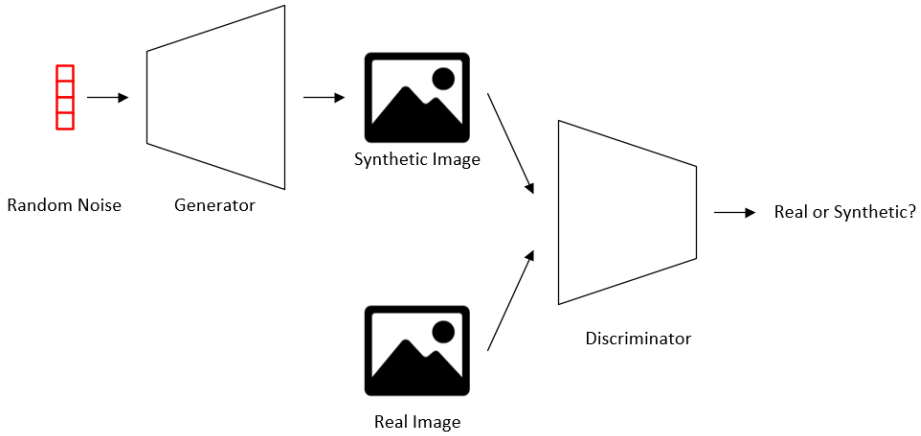


Figure 2.2: A Representation of the GAN Training Process.

The process in which GANs learn to generate "counterfeit" data is formalised in the below steps, in conjunction with Figure 2.2 [Goodfellow *et al.* 2020]:

1. First an image x is sampled from the training data.
2. This image is passed to the discriminator function ($D(x, \theta)$), which is a differentiable function. This function is typically a neural network, with parameters θ that are updated during the training process. $D(x, \theta)$ outputs a binary value (1 or 0).
3. $D(x, \theta) = 1$ means the discriminator evaluates the input image to be a real image, $D(x, \theta) = 0$ means the discriminator evaluates the input image to be synthesised (a counterfeit). Hence $D(x, \theta)$ tries to learn network parameters so that $D(x, \theta)$ is close to 1 since at this stage all the training data consists of real images only.
4. In the second half of the process some random noise is sampled from a distribution z.
5. The sample of random noise is passed to the generator ($G(z, \phi)$). $G(z, \phi)$ is also typically a neural network, with parameters ϕ that are updated during the training procedure. $G(z, \phi)$ is evaluated, and is a new image. The goal of the generator is to generate $G(z, \phi)$ such that it seems just like an image from the training data.
6. $G(z, \phi)$ is passed to the discriminator (D, θ) so that $D(G(z, \phi), \theta)$ is evaluated.
   (a) Here the discriminator network is updated to learn that $D(G(z, \phi), \theta)$ is a synthetic image; that is that $D(G(z, \phi), \theta) = 0$.
   (b) Similarly the generator network is updated to fool the discriminator such that the discriminator evaluates $D(G(z, \phi), \theta) = 1$.

The loss function of the GAN is calculated as a combination of the generator loss and discriminator loss. Formally, the model tries to minimizes a two player mini-max game with value function $V(G, D)$ [Goodfellow *et al.* 2014]:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z)))] \tag{2.3}$$

In equation 2.3 D and G are the discriminator and generator networks with parameters θ and ϕ respectively, and z is some random noise sampled from a prior distribution. This can be considered as a mini-max game as the two networks are effectively *playing* against each other to maximise their own objectives. The discriminator tries to distinguish synthesised samples from real samples, by minimising the number of data samples that it labels incorrectly; and the generator tries to maximise the same thing.

The importance of reviewing GANs at this point is that the work of Makhzani *et al.* [2015] proposes an Adversarial Autoencoder (AAE), which takes an autoencoder architecture, and combines it with a discriminator that is characteristic of GANs. The AAE, is both a GAN and an Autoencoder, and could also be used for generative modelling.

2.7 Understanding the Latent Space

A number of algorithms are often employed to learn a representation of the latent space so that it can be visualised on a 2-dimensional Cartesian plane. Some examples of these algorithms include Principal Component Analysis (PCA) [Baek *et al.* 2002], Independent Component Analysis (ICA), Sammon mapping [De Ridder and Duin 1997], and t-SNE [Maaten and Hinton 2008]. PCA depends on eigen-decomposition to extract orthogonal features that represent variation in the data [Mishra *et al.* 2017]. The first two eigenvectors (ordered by their eigenvalue) encapsulate the most variation of the original data. Hence by visualising these eigenvectors of the decomposition, the variation that exists in the data can be shown on a 2-dimensional plot. ICA searches for a linear transformation W of the data X, such that the dependence between the rows of X is minimised [Baek *et al.* 2002].

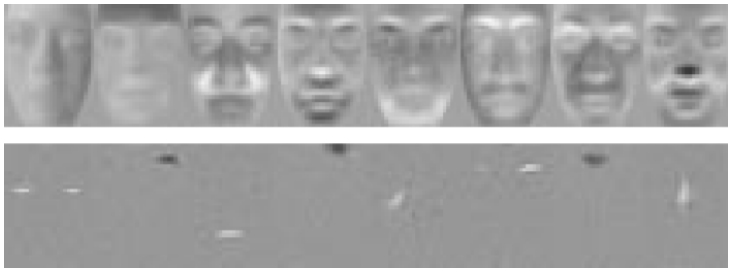


Figure 2.3: PCA Compared to ICA on the FERET Data Set [Baek *et al.* 2002].

Figure 2.3 shows the difference between PCA and ICA on a dataset of faces [Baek *et al.* 2002]. The top row shows the principal components of the faces, the second row shows the independent components. Each image shows one principal component or independent component of the data respectively. This clearly demonstrates how each

independent component corresponds to a particular facial feature, whereas the principal components try to capture the variation in the data. The first principal component (the first image) is an eigenvector that encapsulates the most variation (compared to the other principal components), hence it is appears as a generalisation of all the faces in the data set. On the other hand the first image of the independent components is not necessarily more important than the others. Each component (each image) captures a particular aspect of the face; that is, independent components of the face.

Even though PCA and ICA are widely used to visualise and understand data, the downfall of these techniques is that they are only able to decompose the data using linear transformations, and they do not take class information into account (for classification tasks). Sammon mapping (or Sammon Projection) [Maaten and Hinton 2008] is a method of mapping data from a high dimensional space to a lower dimensional space, while still allowing for non-linear relationships between features. Sammon mapping learns a representation of the data in a lower dimensional space by minimising the cost function:

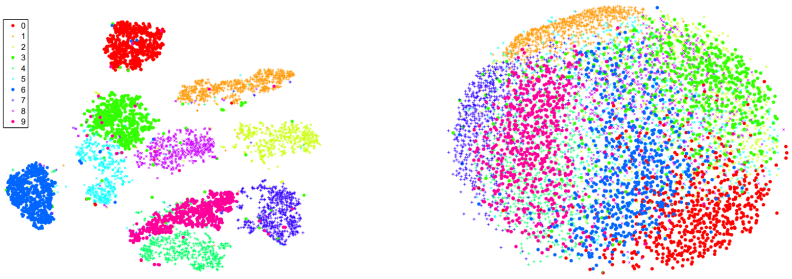
$$E = \frac{1}{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} d_{ij}} \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \frac{(d_{ij} - d'_{ij})^2}{d_{ij}}, \tag{2.4}$$

where d_{ij} is the Euclidian distance between the points in the original data space, d'_{ij} is the Euclidian distance between the points in the lower dimensional space, and n is the number of data points [De Ridder and Duin 1997]. The main advantage of this method over a method such as PCA or ICA is that by visualising the data in a 2-dimensional space, some class separation can be observed. Nevertheless, the cost function does not directly include a term to minimise the distance between samples of the same class in the lower dimensional space. t-Distributed Stochastic Neighbour Embedding (t-SNE) is a method of transforming data into a lower dimensional space, where the optimisation specifically tries to minimise the distance between data points of the same class [Maaten and Hinton 2008]. Formally, t-SNE converst Euclidean distances between points to probabilities that represent how similar any two points are. This conditional probability is given in equation 2.5.

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)} \tag{2.5}$$

In equation 2.5 x are the data points, p is the conditional probability that point x_j is similar to point x_i in proportional to a Gaussian probability density function with standard deviation σ.

A comparison between Sammon mapping and t-SNE is shown in Figure 2.4 [Maaten and Hinton 2008]. The data used is the MNIST data set - a collection of hand written digits. The left image shows how the data is grouped using t-SNE, and the right image shows how the same data is grouped using Sammon Mapping. It can be observed that the t-SNE algorithm has more effectively grouped the data into separate classes. Each class is more *tightly* grouped together with the t-SNE algorithm, which has led to its popularity for this task.



(a) t-SNE Plot. (b) Sammon Mapping Plot.

Figure 2.4: Comparison of t-SNE to Sammon Mapping on the MNIST data set [Maaten and Hinton 2008].

2.8 Triplet Loss

When it comes to encoding image data, one effective way to learn the network parameters to improve the performance of a classification task (for example) is to use triplet loss [Hoffer and Ailon 2015]. Triplet loss uses pairs of images to learn an encoded representation of the input images such that images that belong to the same class are closer together in the latent space, and images that are of different classes are further apart [Weinberger and Saul 2009]. Particularly, two pairs of images are considered at a time when learning the network parameters. A group of three images (called a *triplet*) are termed the anchor image, the positive image and the negative image, where the anchor and positive image are of the same class, and the negative image is of a different class. Together the triplet is considered in two pairs, namely the anchor and positive image form one pair, and the anchor and negative image form another pair. By using these two pairs of images the loss function tries to learn the network parameters such that the difference between the anchor and positive image is small, and the difference between the anchor and negative image is large. Particularly triplet loss tries to learn a representation such that positive and negative pairs are at least a certain distance apart, called the *margin*. Figure 2.5 illustrates this.

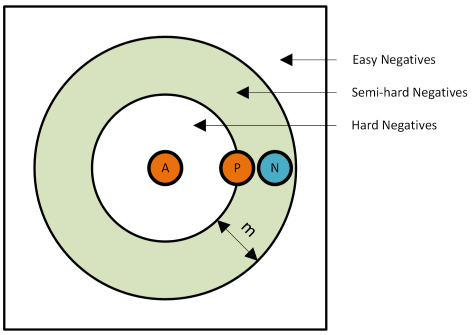


Figure 2.5: A Conceptual Diagram of Triplets.

One important consideration in triplet loss is the way in which triplets are selected from the training data. Some triplets are more important for the network to learn from than others. Triplets can be considered in three categories. Easy negatives are triplets where the negative image is sufficiently far from the anchor and positive image

such that there is at least a distance m (margin) between the positive and negative images. Such samples are not so useful for training since the anchor and positive image are already sufficiently close together, and far away from the negative image. Hard negatives are where the negative image is closer to the anchor image than the positive image. In some cases however, using hard negatives can lead to converging at local minima early on in training [Schroff *et al.* 2015]. It is therefore desirable to define a third category called semi-hard negatives. In this category triplets are in an area between hard negatives and easy negatives, specifically the negative anchor is within some distance m away from the positive anchor. Formally the loss function for triplet loss is shown in equation 2.6.

$$\mathcal{L} = max(d(a,p) - d(a,n) + m, 0) \tag{2.6}$$

Equation 2.6 says that the difference between the distance between the anchor sample (a) and positive sample (p), and the distance between the anchor (a) and negative sample (n), must be at least m. This means that the distance between the anchor and negative image must be at least m bigger than the distance between the anchor and positive image.

2.9 Neural Style Transfer

In this research report a metric used to quantify artistic style in images is used to draw conclusions concerning the latent space. This metric relies heavily on the techniques used in Neural Style Transfer.

Neural Style Transfer is when the artistic style from one image, called the *style image*, is applied to another image, called the *content image*, by using deep learning. This was first introduced by the work of Gatys *et al.* [2015]. In their work the VGG 19 architecture [Simonyan and Zisserman 2014] is used to extract content and style information from a particular input image, which allows style transfer between images. Figure 2.6 shows a representation of the VGG19 architecture. In this Figure, the white blocks are convolutional layers, and the blue blocks are max pooling layers. Each convolutional layer (referred to as a *block* in Figure 2.6) is followed by a max pooling layer, where the maximum value of local areas in the convolutional block's feature map are filtered on. In this Figure, *kernel* refers to the size of the filter that is used in the convolutional layer, and *size* refers to the number of filters that are used to make up the convolutional block.

Formally, neural style transfer can be formulated as an optimisation problem. Consider two input images, namely the content image and style image. The goal of the algorithm is to create a new image, called the artwork, that represents the content of the content image (for example, buildings, people, shapes etc.) but in the artistic style of the style image. The loss function (total loss) in the optimisation problem is therefore composed of two parts, where each part pertains to the content loss and style loss respectively:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{content} + \beta\mathcal{L}_{style} \tag{2.7}$$

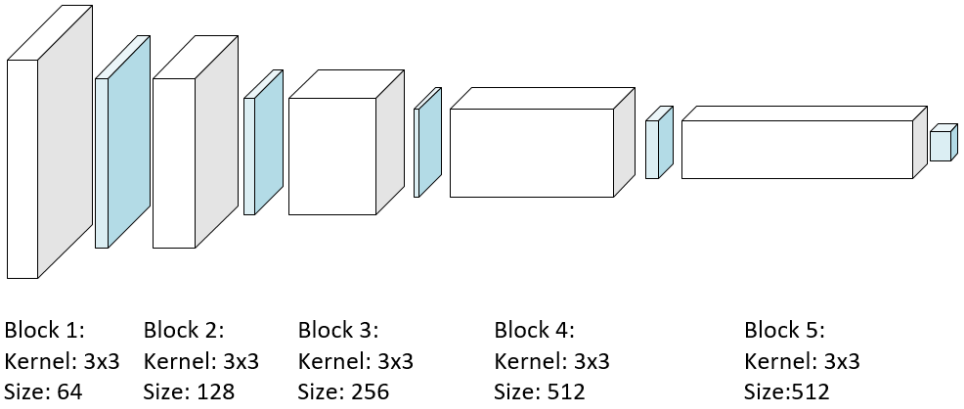


Figure 2.6: VGG19 Architecture.

In equation 2.7 α and β refer to weights given to the content and style loss. The authors do not publish what values for α and β are used in their method, however they specify a ratio of α to β. Various ratios are used in their work; typically the ratio of α to β is $1:100$, or $1:1000$, which indicates that the loss function is weighted towards style loss. The detail of each of the terms in equation 2.7 is now discussed. The content loss can be formulated as:

$$\mathcal{L}_{content} = \frac{1}{2}\sum_{i,j}[F_{i,j} - P_{i,j}]^2 \tag{2.8}$$

In equation 2.8, F is the network response of the content image at the content layer, and P is the network response of the artwork at the content layer, where i and j are the dimensions of the network responses (they have the same dimensionality). Gatys *et al.* [2015] experiment with different combinations of style and content layers, and evidence their choice of style and content layers based on empirical results. The most common layer used for content loss is the second convolutional layer in the forth block of convolutions(see Figure 2.6). The intuition behind this is that the artwork should represent the content image in its content, so the mean square error between the response of the two images at this layer is minimised.

With reference to Figure 2.6, with each block, the first convolutional layer is used in calculating style loss. For each of these layers, the Gram matrix is evaluated. The Gram matrix is the dot product of the flattened feature response of the style image and artwork at each of the style layers. By calculating the Gram matrix of each of the convolutional layers, correlations between the different feature responses are identified, and these correlations quantify how similar the artwork is to the style image in terms stylistic elements. The intuition of this can be understood with a simple example. Suppose that a specific channel might represent whether there are black and white stripes in the image, another channel might represent the outline of a car. If the input image generates a large response in both these channels, our interpretation would be that the artistic style of the image is that the artist has painted cars with black and white stripes. This also shows how the semantic meaning of objects in the input image are combined with stylistic properties. Formally, the style loss is given in equation 2.9.

$$\mathcal{L}_{style} = \sum_{l} w_l \frac{1}{4N_l^2 M_l^2} \sum_{i,j} [G_{i,j}^l - A_{i,j}^l]^2 \tag{2.9}$$

Here G is the Gram matrix of the original image at layer l, A is the Gram matrix of the artwork at layer l, and M and N are the height and width dimensions of the layer. This means that the optimisation problem minimises the difference in *style* between the two images. In this formulation w is an important parameter which assigns an importance weight to each of the convolutional layers. Each weight (w) is given a default value of 0.2, although the authors experiment with different values to see how this affects the performance of their method.

Figure 2.7: An Example of Neural Style Transfer.

Figure 2.7 shows an example of Neural Style Transfer. [Gatys *et al.* 2015]. Here the content image (left) is a photo of Tubingen in Germany. The style image is The Starry Night by Vincent van Gogh (middle). The third image shows how the artistic style from van Gogh's artwork has been transferred to the photo of Tubingen (Right).

2.10 Conclusion

In this Chapter a number of topics were covered. The purpose of this is to introduce these topics in an intuitive way so that they can be referenced throughout the research report. The important concepts covered are the different types of autoencoders, and how they are different; PCA, ICA, t-SNE, and Sammon Mapping as a means of understanding the latent space; and finally Neural Style Transfer was presented, which demonstrated the use of style loss in deep learning. The formulation of style loss is crucial to this research report. The significance of style loss is that it abstracts the artistic style of the artworks into a quantifiable metric. This report uses this metric to evaluate the latent space of various models.

Chapter 3

Related Work

3.1 Introduction

In this Section related work to the research is reviewed. The purpose of this Section is to evidence that the report builds on what others have done, and that the methodology is well grounded. Firstly, related work concerning triplet networks (Section 3.2) and adversarial autoencoders (Section 3.3) is reviewed. The reason why these are discussed is because their respective model training procedures directly manipulate the latent space in a way that is different to the autoencoders of the previous Chapter. Thereafter Section 3.4 presents latent space cartography, which concerns a framework for exploring the latent space, is presented. This is also a crucial area of literature in contemporary research, since using style loss to explore the latent space can be seen as an extension of their work. Thereafter the work of Oring *et al.* [2020] is reviewed in Section 3.5. Their work is significant, in that the ideas of latent space cartography (specifically latent space interpolation) are included in the training procedure of their model; as opposed to this research report where interpolation is done afterwards as a separate task. Finally in Section 3.6 a discussion on neural style transfer is included, where Hicsonmez *et al.* [2020] provide an alternative formulation of style loss. This evidences that style loss is still an area of active research in recent literature.

3.2 Triplet Networks

Schroff *et al.* [2015] use triplet loss to train a deep convolutional network for face verification. The accuracy of their method was unprecedented at the time (in 2015). The method followed uses one of two deep neural networks, with a loss function formulated according to triplet loss as discussed in Section **??**. In their experiments two deep neural networks are compared, the first architecture is inspired by Zeiler and Fergus [2014], and the second architecture is based on work by Szegedy *et al.* [2015]. The significance of their work is that by using triplet loss as a metric to regularise the latent space, faces of the same (or similar) people will occupy a particular region in the latent space. By traversing the latent space in a particular area, and reconstructing images from positions along the traversal, one expects that all the images in a small area would approximately reconstruct faces with similar attributes.

Hoffer and Ailon [2015] propose a generalized triplet model to learn useful representations of labelled data. The authors use triplet loss to train a model similar to a Siamese network [Bromley *et al.* 1994]. Siamese networks are parallel networks with shared network weights. The architecture proposed by Hoffer and Ailon [2015] is shown in Figure 3.1. The network is trained by feeding the anchor, positive, and negative images to a copy of the network in parallel. Then the distances between the anchor and positive image, and the anchor and negative image are used to calculate the triplet loss. The network parameters are then updated by using both the triplet loss and reconstruction loss.

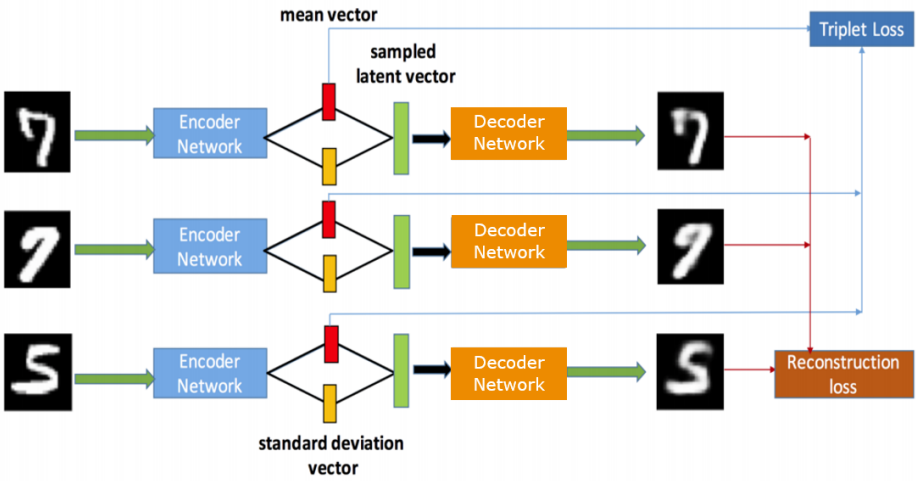


Figure 3.1: A Representation of the Triplet Network [Ishfaq *et al.* 2018].

By using this Siamese-like architecture the authors show that the learned representation outperforms its predecessors, and can learn representations on various labelled data sets. Their work leverages the advantages of a deep network by Schroff *et al.* [2015] but with the advantage that the learning procedure can be generalised to work for other tasks. The significance of the contribution of this work is that by using triplet loss with a deep learning architecture, a latent space where classes have been separated is consistently learned. Therefore this architecture is chosen as the first architecture that will be compared in this research report. To be concise with naming conventions, this architecture is referred to as TVAE (Triplet Variational Autoencoder) in this report.

3.3 Adversarial Autoencoders

The work by Makhzani *et al.* [2015] is highly relevant to this research report, and the architecture of this work is one of the architectures that will be used in the proposed experiments. The authors use a GAN architecture [Goodfellow *et al.* 2014] and combine it with a convolutional autoencoder, to create an adversarial autoencoder (AAE). The schematic diagram of the architecture is shown in Figure 3.2. In this architecture the encoder of the autoencoder is the generative network, and the discriminator tries to distinguish samples as originating from the generator or from the prior distribution $p(z)$, where $p(z)$ is a distribution chosen beforehand (typically a Gaussian distribution).

The result is that the encoder learns to map input samples to a latent space where the data is indistinguishable from samples from the prior distribution. Therefore the autoencoder acts as a generative network because the decoder portion of the network generates new samples from the latent space, where samples from the latent space resemble samples from the prior distribution. To train the adversarial autoencoder the adversarial and autoencoder components of the network are trained concurrently by adding an adversarial loss term to the cost function.

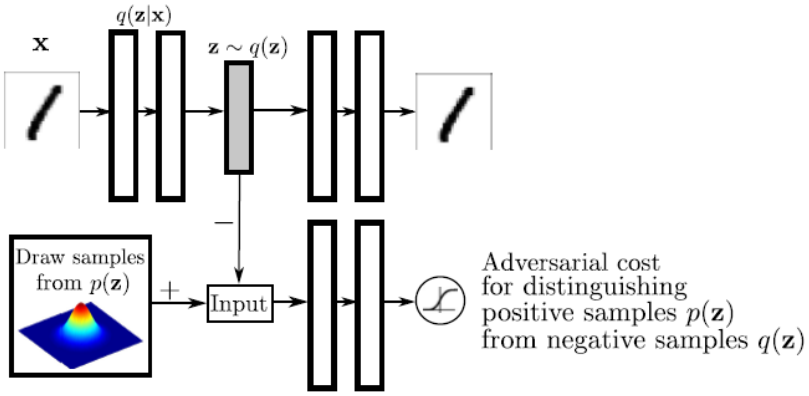


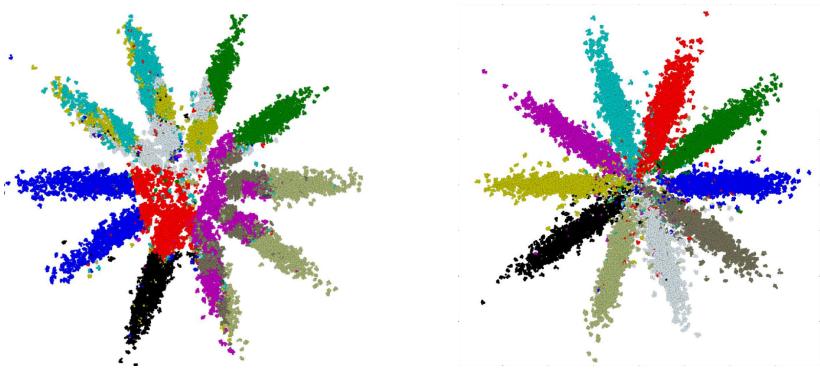
Figure 3.2: Adversarial Autoencoder Architecture [Goodfellow *et al.* 2014].

Figure 3.3a shows the latent space of an AAE trained on the MNIST data set, where the prior is chosen as a mixture of 10 2-dimensional Gaussians. This is why the latent space in this Figure has 10 *arms*, since each *arm* represents one of the Guassians in the Gaussian mixture. Figure 3.3b presents the latent space of a modified version of the AAE. The discriminator is passed a one-hot encoded vector that indicates to which class each sample belongs. In Figure 3.3b the AAE successfully matches each mode of the Gaussian (there are 10 modes) to a class in the data set (there are 10 classes). This should be contrasted with what triplet loss achieves. With triplet loss, samples with the same class label are closer together in the latent space by using class information in the triplet cost function. With the adversarial autoencoder the latent space is conditioned by feeding class information to the discriminator network.

The significance of this work is that the encoded data in the latent space is encouraged to match a chosen prior distribution. The AAE architecture can further be modified to take the class information in the training procedure, in an attempt to match each class to a different mode of the prior distribution.

3.4 Latent Space Cartography

Liu *et al.* [2019] have contributed research concerning latent space cartography, which they describe as mapping and comparing meaningful semantic dimensions within latent spaces. This work reviews various tasks that can be conducted to understand the latent space, common practices in exploring the latent space, and the demonstration of how a nine task framework which the authors propose, can be used to explore the latent



(a) Without Conditioning. (b) With Conditioning.

Figure 3.3: AAE: Latent Space Visualisation [Goodfellow *et al.* 2014].

space of a training set of emoji images. The framework focuses on exploring latent spaces that are used for generative modelling and word embedding applications. Even though their work focuses on word embeddings, the principles used in the way they explore the latent space are also appropriate in computer vision. They further support their workflow with a visual analytics system to support these tasks. The main tasks in their framework are reviewed in this Section. The purpose of this is to provide context on the work that has been done on methods of exploring the latent space. This provides a concrete motivation for the means by which the latent space is explored in this research report. The main tasks in their work are:

1. **Viewing reconstruction samples.** This is the most intuitive way in which the latent space can be explored. In the case of autoencoders, by reconstructing samples using the decoder network, it can be quite easily determined how well the latent space is regularised. If reconstructed samples (that are not in the training data) resemble samples from the training data, that would be a strong indication that the latent space has been regularised. Particularly in computer vision this is extremely useful. By forming images from decoded points, the images can be visually inspected to decide if the model can produce plausible outputs from the latent space. Although in computer vision this is often only a qualitative task, for spatial data this could also include a quantitative aspect, since the decoded points could be tested if their statistical properties are similar to the original data distribution.

2. **Interpolating between samples.** With linear interpolation, two points are chosen in the latent space, and a linear interpolation is performed. At each step in the linear interpolation an image is generated using a generative model. By doing this, a region in the latent space can be evaluated, rather than mere individual samples. If the transition between the reconstructed images is smooth, it could imply that: firstly, the latent space is adequately regularised, and secondly that the interpolation is on the data manifold. If the transition between interpolated images is not smooth, it would indicate that the interpolation has ventured off the data manifold.

3. **Examining of nearest neighbours.** With this task, a particular sample in the latent space, and its nearest neighbours are identified, and a generative model is used to decode the samples. Since the samples in the latent space are close to each other, it is expected that the decoded samples would have some observable similarity. The degree to which samples are similar could indicate the shape of the data manifold distribution, or the degree to which the latent space is regularised. If nearby samples are dissimilar, that would indicate a lack of regularisation in the latent space.

4. **Visualising distributions.** This task involves visualising the latent space with techniques such as t-SNE (as discussed in Section 2.7). Although this is a very popular method of understanding the latent space, a major concern of this is that 2-dimensional or 3-dimensional plots may not adequately represent latent spaces that often have hundreds of dimensions.

5. **Inspecting attribute vectors.** By combining the tasks of viewing reconstruction samples and interpolation, Liu *et al.* [2019] come up with another method of inspecting the latent space. In this task multiple interpolation trajectories are compared and plotted on a 2-dimensional plot of the latent space.

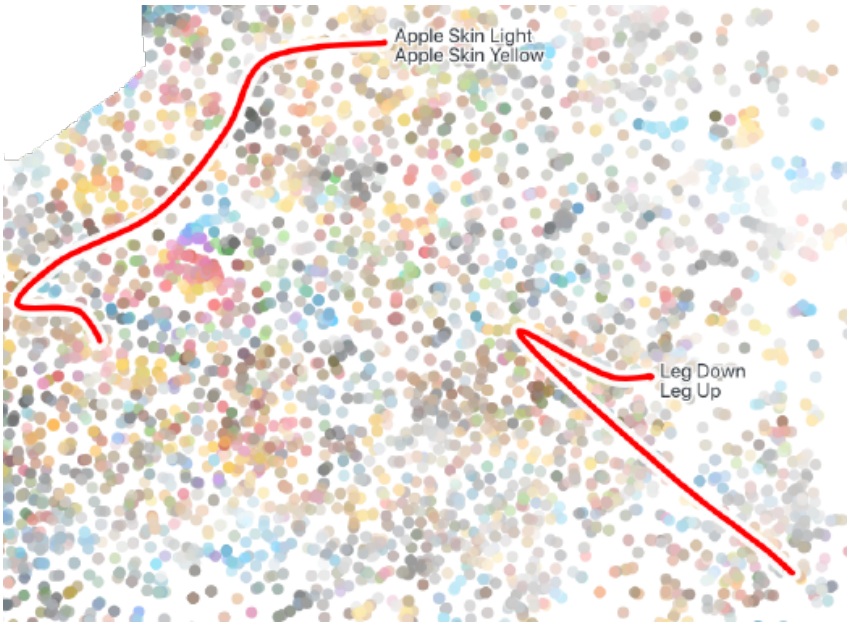


Figure 3.4: Attribute Vector Map [Liu *et al.* 2019].

Figure 3.4 shows an example of such a plot. The small dots each represent a data point. In this experiment a data set of emojis is used. The red lines are called attribute vectors. An attribute vector is an interpolation, that acts as a hypotheses that a relationship exists along the trajectory. For example the left red line links an area in the latent space of emojis with light apple skin, to yellow apple skin. By selecting this line, the image reconstructions between these samples can then be viewed. If the model sensibly interpolates in this trajectory (i.e. a smooth transition), then we may accept this attribute vector since the latent space does

seem to be organised according to skin colour (at least for these colours of skin). Conversely if it is found that foreign images are introduced in the interpolation, the attributed may be rejected. The value in this method, is that the 2-dimensional plot of the latent space can be annotated to show where sensible relationships exist in the latent space.

The compilation of tasks into a single framework by Liu *et al.* [2019] provides an extensive workflow as to how one can go about understanding and exploring the latent space. Even though this framework provides a fairly comprehensive set of tools to visualise and interact with the latent space, it does not provide a means where by the latent space can be measured by using a metric. If a metric could be used to take measurements from the latent space, this could allow a quantitative measure of the degree of regularisation in the latent space. This is a strong motivation for the use of style loss as a metric in exploring the latent space in this report.

3.5 Shaping the Latent Space for Interpolation

In the work of Oring *et al.* [2020] a regularisation technique is proposed to promote convexity of the data manifold. By improving the structure in this way, images sampled from a trajectory along an interpolation are more likely to represent images from the data distribution. The authors argue that without shaping the latent space, a mere linear interpolation between two points on the data manifold would result in deviating from the data manifold. They propose shaping the latent space by enhancing the training of an autoencoder by using a particular loss function.

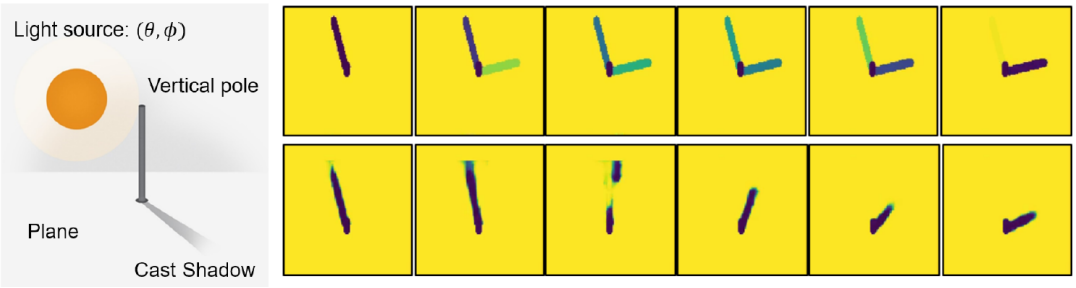


Figure 3.5: Image Interpolation without Shaping [Oring *et al.* 2020].

A data set of images of a pole casting a shadow is used. Figure 3.5 (left) shows how the images are generated. A pole casts a shadow because of a light source shining on it. As the light source is moved the shadow of the pole is cast differently. At each position of the light source, the scene is captured from above in order to show the length and orientation of the shadow cast. Movement of the light source is parameterised by elevation (θ) and azimuth (ϕ).

The two rows of images shown in Figure 3.5 are the result of two interpolations. In this colour scheme the colour of the shadow indicates the intensity of the shadow. The top row is an interpolation in the domain of the input images. Interpolating in the domain of the images can merely achieve a *fading* effect. As one traces the images from

left to right it can be seen that the shadow in the left most image fades away, while the shadow of the right most image gradually appears. This is because in the domain of the image data, when one only considers these two images, there is nothing to suggest that the shadow should move in a clockwise direction. In the second row the interpolation is done in the latent space. Here the interpolation does not merely fade-in or fade-out the original images (far left and far right of each row), but instead the movement of the shadow in a clockwise direction is evident. This is because the latent space has learnt (to an extent) that there are latent factors that directly affect the position and length of the shadow. We know that the latent factors controlling the orientation and length of the shadow are the elevation and azimuth of the light source, and so ideally the latent space should be a representation of these factors. If one could traverse the data manifold in the latent space such that a traversal correlates to an interpolation in the domain of elevation and azimuth, then the latent space could be said to perfectly capture the latent parameters of the data distribution.

The authors improve the structure of the latent space so that the structure of the latent space better represents the latent parameters (elevation and azimuth) from which the image data is created. A loss function of four terms is used to achieve this (Equation 3.1), where each term is weighted by a factor of λ.

$$\mathcal{L}^{i \to j} = \mathcal{L}_R^{i \to j} + \lambda_1 \mathcal{L}_A^{i \to j} + \lambda_2 \mathcal{L}_C^{i \to j} + \lambda_3 \mathcal{L}_S^{i \to j} \tag{3.1}$$

In this equation the loss is the loss for a particular traversal from point i to j, and so in each forward pass of the network a pair of images is used. The terms of the loss function are:

- **Reconstruction Loss ($\mathcal{L}_R^{i \to j}$)**: the mean squared error between the input image and reconstructed image, which is typical of autoencoder networks.
- **Adversarial Loss ($\lambda_1 \mathcal{L}_A^{i \to j}$)**: for a each pair of images a new sample is interpolated between the two images in the latent space. A discriminator is then used to distinguish between the **real** points and the interpolated points. This results in sampled points from interpolation to yield images that look as though they come from the same data distribution as the input data.
- **Cycle-consistency Loss ($\lambda_2 \mathcal{L}_C^{i \to j}$)**: points that are decoded from the latent space are mapped back to the latent space. This further ensures that decoded latent samples are mapped back to the image manifold.
- **Smoothness Loss ($\lambda_3 \mathcal{L}_S^{i \to j}$)**: Ensures that there is smoothness between interpolated images from latent vectors.

The authors compare their method to other existing autoencoder architecture. They train their model and perform a set of interpolations on the COIL-100 dataset, and compare their method against an Adversarial Autoencoder (AAE) [Makhzani *et al.* 2015], Adversarially Constrained Autoencoder Interpolation (ACAI) [Berthelot *et al.* 2018] and β-Variational Autoencoder (β-VAE) [Higgins *et al.* 2016].

Figure 3.6 shows the comparison of the results. Interpolation is done from the left most image to the right most image in each row. It is clear that by shaping the

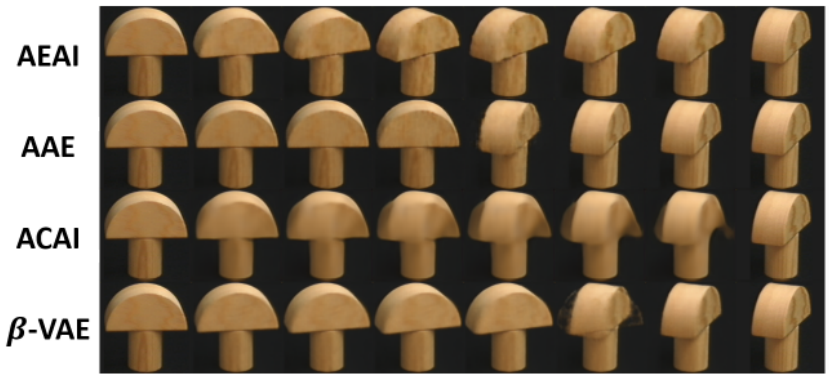


Figure 3.6: Image Interpolation with Shaping [Oring *et al.* 2020].

latent space as seen with Autoencoder Adversarial Interpolation (AEAI) the interpolated images evidence that the latent space is able to infer how the object should rotate. Notice that the Adversarially Constrained Autoencoder Interpolation (ACAI) [Berthelot *et al.* 2018] and β-Variational Autoencoder (β-VAE) evidence a cross-dissolve effect, and have *merged* the start and end points in the trajectory. This is possibly because the β-VAE has left the data manifold, causing the poor reconstructions. This evidences that by introducing an adversarial loss component in the loss function, convexity in the data manifold is encouraged. Therefore it appears that the models with an adversarial loss term have better reconstructions.

The importance of this work in terms of this research report is that the authors address a similar problem to this report, except they approach it in a different way. In this research report, as well as in the work on latent space cartography (Liu *et al.* [2019]), the approach is to use a suitable method or tool to explore and understand the latent space. From the work of Oring *et al.* [2020] one could argue that it is not so much the evaluation method that assists in understanding the latent space, but the training procedure itself that should assist in making the latent space easy to interpret. The work of Oring *et al.* [2020] was only published after the results from this research report were completed. Therefore even though their method is not used in this research report, it could be considered as further work.

3.6 Quantifying Artistic Style as a Metric

In Section 2.9 the concept of neural style transfer was reviewed, and in Section 2.6 an introduction to GANs was provided. Hicsonmez *et al.* [2020] combine GAN architecture with neural style transfer concepts to form Ganilla. Ganilla is a GAN that transfers style from illustrated children's books to new images. The authors show that the quality of the style transfer that can be achieved outperforms previous models. In addition to their model which is capable of state of the art style transfer, the authors propose a framework for quantitative evaluation of content transfer and style transfer. This is done by training two separate CNNs that evaluate images in terms of how well the content and style have been transferred to generated samples. The style CNN evaluates whether a sample belongs to a particular style, and the content CNN evaluates whether

the content of the generated image has similar content to the content image used to generate the sample. The relevance of the style CNN and content CNN that the authors use to evaluate images, is that this is where the motivation for evaluating style loss in this research report comes from. Their work is a recent work that evidences that evaluating artistic style retrospectively in generated images is a sensible thing to do. Hicsonmez *et al.* [2020] use style loss to transfer style, but they do not use it to evaluate and explore the latent space, as is done in this research. Therefore it can be argued that the use of style loss to evaluate images retrospectively can be backed up by recent literature.

3.7 Conclusion

This Section reviewed some of the major works in literature that are relevant to this research report. First the works of Hoffer and Ailon [2015] and Makhzani *et al.* [2015] are reviewed to introduce the TVAE and AAE, which are two of the main architectures that are used in this research report. Thereafter the work of Liu *et al.* [2019] is discussed. The importance of their work is that they provide a framework for exploring the latent space, which is what this research report is focused on. Particularly it is noted that most of the tasks in their framework are qualitative, whereas in this research report style loss is used as a quantitative method to explore the latent space. The work of Oring *et al.* [2020] is relevant because it showed that interpolation can be done during the training of autoencoders, to better regularise the latent space better. This is different to many other works where interpolation is only done retrospectively; that is after the models have finished training. Finally the work of Hicsonmez *et al.* [2020] evidences that style loss is still an active area of research.

Chapter 4

Research Methodology

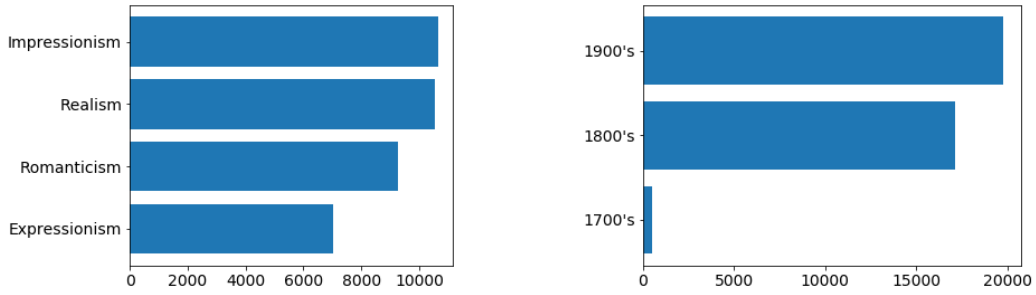
4.1 Introduction

In the Introduction of this research report, it was discussed that there are only limited methods that are used to quantitatively explore the latent space. Most of the techniques are qualitative (visualisation the latent space, reconstructing samples etc.). This Chapter provides the research methodology as to how style loss can be used to quantitatively evaluate the latent space.

4.2 Data

All experiments are conducted with a data set of artworks available on Kaggle [Nichol 2016]. The data consists of 103250 artworks of various artists. Information is available for each artwork including the year, artist, and artistic style. In this research the artistic style (also known as *genre*) is chosen as the label for each of the artworks; hence each artwork is said to belong to a certain class. It is decided that not all of the classes would yield meaningful insight in the research because of the sparsity of some of the classes. Instead only the four most populous classes are used in the research, namely, Expressionism, Impressionism, Realism, and Romanticism. This brings the total number of samples in the data set to 37464, with each class having between 6000 - 11000 samples. The data set includes artworks made with various media, including sketches (ink, charcoal, pastel) and paintings (acrylic, oil, vinyl, and others). Figures 4.1a and 4.1b shows some basic statistics concerning the data.

The complexity of the data distribution is non-trivial. For someone who is not educated in the visual arts, it may prove difficult to correctly classify artworks into their respective class (Expressionism, Impressionism, Realism, and Romanticism). Figure 4.2 shows an example of four images from the Impressionism and Expressionism classes respectively. The first two images on the left are from the Impressionism Class, and the two artworks on the right are from the Expressionism class. The challenge presented by these artworks is that they have similar visual features. The use of colour in the artworks is very similar, and the use of *texture* (style of brush stroke) is also similar. There are also no objects or shapes that could be used to distinguish the Impressionism artworks from the Expressionism artworks. Only someone with reasonable education



(a) Number of Samples per Class (b) Number of Samples per Century

Figure 4.1: Training Data - High Level Statistics.

in visual arts would easily be able to distinguish which artworks are from the respective classes. It can be reasonably be assumed that a neural network would also struggle to distinguish artworks such as these in terms of their artistic style. Just because two samples are from the same class it does not necessarily mean that they will be close together in the latent space. It may well be that there are artworks from different classes that will be quite close together in the latent space.

Figure 4.2: Artworks of Different Classes with Similar Visual Features.

Just as the artworks from different classes in Figure 4.2 have similar visual features, the visual features of the artworks in a single class are also disparate. Figure 4.3 shows an example of artworks from the Impressionism class.

Figure 4.3: Impressionism Artworks with Disparate Visual Features.

Although the images are all from the Impressionism class, there is a large variety in the content of each of these images. Consequently, it is anticipated that the clusters

of data in the latent space may not be clearly be identifiable to one class, but rather that clusters will be composed out of a mixture of classes. See Section 4.6 on a brief discussion on understanding the stylistic characteristics of the different classes.

4.3 Model Selection

The goal of this research report is to use style loss to compare the latent space of different models by training them on the data set of artworks. These five models are:

1. **Autoencoder (AE).** This model is chosen because of its simple architecture. This model serves as a baseline to which the other models can be compared. See Section 2.3.

2. **Variational Autoencoder (VAE).** This follows a similar basic architecture to the AE, except that an element of stochasticity is brought in to the latent space of the network, which aids in regularising the latent space [Kingma and Welling 2014]. A stochastic term is introduced in the network loss function, and is measured with Kullback–Leibler divergence. See Section 2.4.

3. **β-Variational Autoencoder (β-VAE.)** This is the same architecture as the VAE, except the Kullback-Liebler divergence term in the loss function is scaled by a factor of β; $0 < \beta < 1$ [Higgins *et al.* 2016]. See Section 2.5.

4. **Triplet Variational Autoencoder (TVAE).** A third term is introduced to the loss function of the β-VAE. This term, called triplet loss, enforces that the distance between samples from the same class is less than the distance between samples of different classes [Hoffer and Ailon 2015]. See Section 3.2.

5. **Adversarial Autoencoder (AAE).** This model is significantly different to the others. The basic autoencoder architecture is maintained, but a discriminator network is introduced. This introduces an adversarial loss term into the autoencoder loss function [Makhzani *et al.* 2015]. See Section 3.3.

4.4 Research Questions

With the five models chosen in section 4.3, the following research questions are formalised:

- **Research Question 1:** Is there evidence that samples have been grouped according to their class label in the latent space?

  Motivation. Before doing style loss experiments it would be beneficial to get a basic understanding of the data manifold. This research question examines the data manifold of the different models using two methods; namely, PCA and clustering methods. The goal of this is to determine to what extent the data has been grouped according to the class labels in the latent space.

The principal components give an indication of how the data is distributed in the data manifold. Each component is considered in terms of the percentage variance that it contributes. If the variance contribution of the first principal component is very high compared to the others, it means that the data is distributed along an axis corresponding to that principal component. If the variance contribution of the principal components is spread more equally between the components, it means that there is not an axis along which the data is pre-dominantly distributed.

Clustering the data in the latent space gives an indication on how clearly the classes are separated from each other. Ideally there would be four clusters that are clearly separated from each other, since there are four classes in the data set. However it is more likely that the clusters will have a mixture of classes. By clustering the data in the latent space, the degree to which the classes have been clustered together can be ascertained.

- **Research Question 2:** How does the style loss change when interpolating within one class in the latent space (intra-class interpolation)?

  Motivation. In research question 1, whether or not there are clusters according to the class labels in the latent space can be established. In this research question, the style loss within each of the classes will be evaluated. This is done by calculating the style loss across a number of pairs of images over an interpolation trajectory. The manner in which the style loss changes over the trajectory would give rise to discussion on the convexity of the data manifold in the latent space. By interpolating within one class, the results will indicate how the latent space has been regularised within that class. Sudden changes in style loss or sudden changes in the reconstructed images in the trajectories would indicate that there is a lack of regularisation in the latent space for a particular class. See section 4.5 for detail on how interpolation is used in conjunction with interpolation to design the experiments.

- **Research Question 3:** How does the style loss change when interpolating between two artistic styles in the latent space (inter-class interpolation)?

  Motivation. The motivation for this research question is similar to that of research question 2. Here the interpolations are between two classes at a time. This will give rise to discussion on how the latent space has been regularised between classes in the latent space.

4.5 Using Style Loss as a Metric

The calculation of style loss is implemented according to the work of Gatys *et al.* [2015]. This research report uses style loss to evaluate the latent space of the different models. The procedure in which this is done is summarised as follows:

1. Two images are chosen according to experiment specific criteria. Various experiments are designed, and image pairs are chosen accordingly.

2. The images are passed through the encoder portion of the trained models. The latent representation is recorded for each of the images. The two images now exist in the domain of the latent space, and the two positions of the images in the latent space are the starting point and ending point of the interpolation trajectory.

3. For the interpolations, it is decided that 10 interpolated steps is sufficient. Choosing a higher number of steps would make the images cumbersome to interpret, while using fewer steps would not adequately represent the interpolation trajectory. This is also in accordance with the work of Radford *et al.* [2015]. Choosing 10 interpolation steps does not mean that there are 10 points along the trajectory. There are 11 points along each interpolation trajectory, and two points are separated by one step. Therefore there must be 11 points in the trajectory to have 10 steps.

4. The positions of the interpolated points are then determined by linearly interpolating between the starting point and the ending point of the trajectory.

5. Each interpolated point along the trajectory is then passed through the decoders of the respective models to produce an image. An example of the generated images is shown in Figure 4.4.

6. Style loss is calculated between each consecutive pair of images along the trajectory. For example in Figure 4.5, the style loss at point 1, is the style loss between images 1 and 2 in the trajectory.

7. The style loss for all of the image pairs is plotted on a line graph to view how the style loss changes along the trajectory in the latent space. With the example shown in Figure 4.5, the style loss is shown along the interpolated trajectory. The type of observation that arises from inspecting this plot is that the VAE has noticeably lower style loss. By inspecting the images from the interpolation in Figure 4.4 we can observe that this is correlated to the images of the VAE being consistently blurry. This illustrates how these plots may be used to make observations on how style loss can be used to make comparisons between the different models.

The intuition behind the plot in Figure 4.5 is that a high style loss indicates that the two images that are being compared are different in terms of their artistic style, whereas a low style loss indicates that the two images are similar in their artistic style.

4.6 Style Loss Interpretation

What is artistic style? As this research deals with style loss, it is worth describing what style actually means. Fernie *et al.* [1995] describe style as distinctive elements that allow works to be grouped into distinct categories. On the one hand such elements include, medium, use of colour, themes, and brush stroke, which describe the artwork itself. But it is not only the elements that contribute to a particular style. In the visual arts, artistic styles also often refer to a particular period in time. For example, Impressionism refers also to a particular art movement that took place in the 19^{th} century,

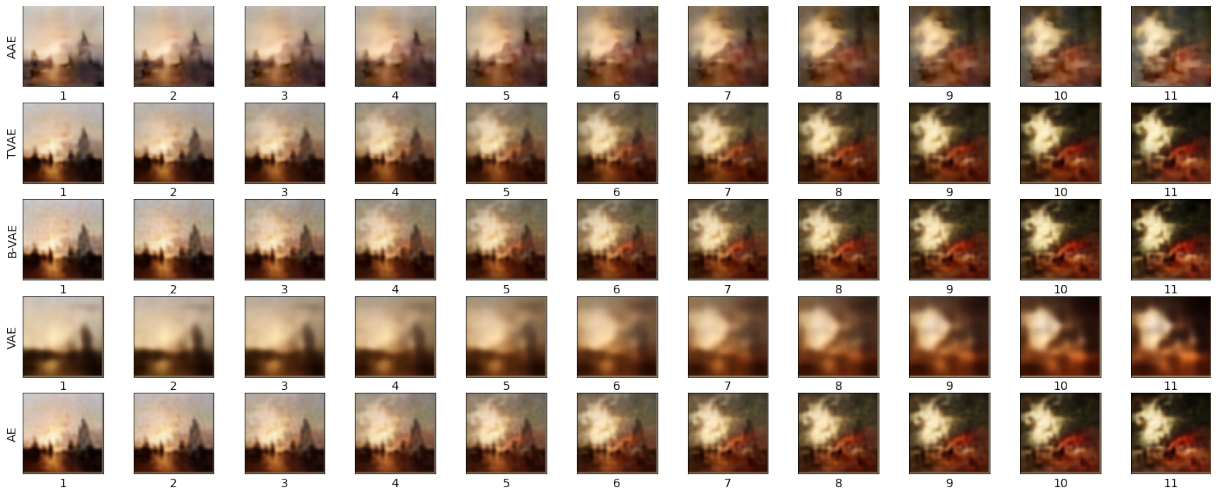


Figure 4.4: Example Image Interpolation.

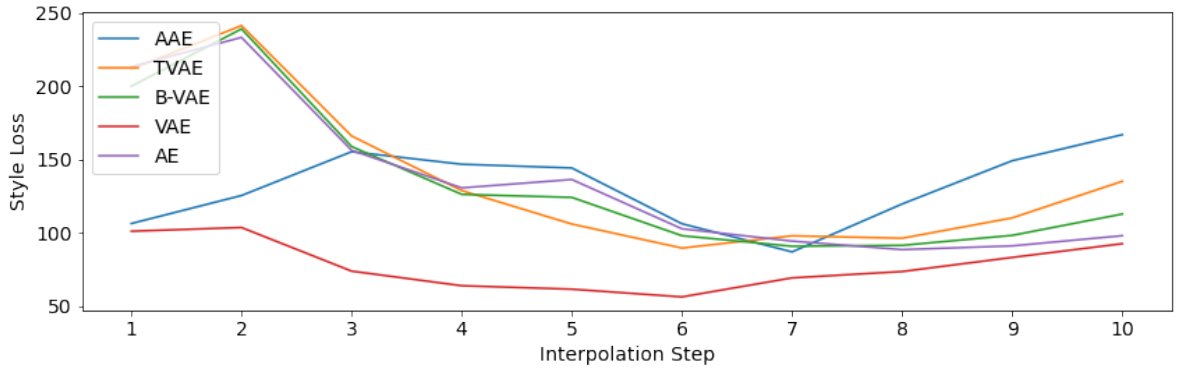


Figure 4.5: Example Style Loss Plot.

where the artists often emphasise temporal qualities in their artwork [Bomford and Britain) 1990]. So, it is not only the elements of the artwork that contribute to the style, but also the history of the period that the artists lived in that affects the style in artworks.

In the presentation of results, various conclusions concerning the style loss along the different trajectories are drawn. Ideally, an Art expert would verify whether or not the perceived changes in the artistic style of the generated images agree with what is seen in the style loss; however, that is beyond the scope of this research report.

4.7 Conclusion

This Chapter presented the research methodology. The data set is introduced, along with some examples of the artworks. The main purpose of this Chapter is to describe what the research questions of this report are. The first question deals with understanding the latent space using PCA and clustering methods. This is to provide a basic understanding of how the data in the latent space is organised. The second and third question investigate how latent space can be explored and understood by using style

loss. Based on this methodology, the following Chapter will discuss how the models are trained so that the research questions can be answered.

Chapter 5

Model Training & Performance

5.1 Introduction

In this section the implementation of the models is presented. First, an overview of the model architectures is given in section 5.2, and then information on how they are implemented is presented. Finally an example of the images reconstructions are shown for each of the models in section 5.3.

5.2 Model Architecture Implementation

Each of the models are trained using a similar neural network architecture for the autoencoder portion of the models. The choice of design of the autoencoder architecture follows the work of Subramanian [2020]. Their work provides source code for several different types of autoencoders, of which only one, namely VAE is also used in this research report. The autoencoder architecture used in this research report is similar to the VAE architecture used by these authors. This architecture is presented in Figure 5.1.

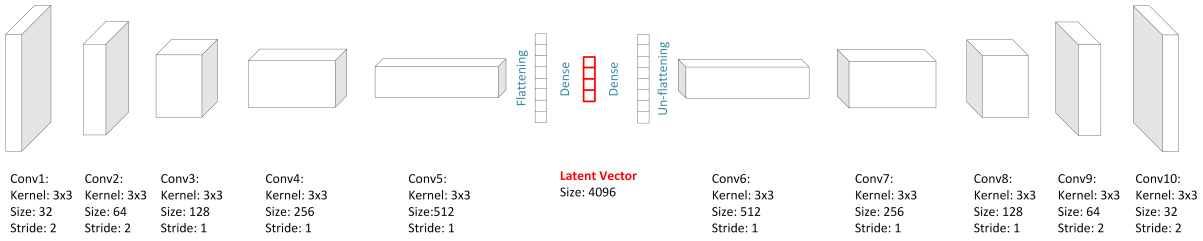


Figure 5.1: Autoencoder Architecture.

The architecture consists of four convolutional layers to encode the image data. Thereafter two linear (dense) layers are used to encode and decode the data. After the data is passed through the first dense layer, the data has the lowest dimensionality. It is at this layer that the data is considered to be encoded in the latent space, and it is in this domain where the interpolation is done in the experiments (see Latent Vector in Figure 5.1).

A key parameter in the experiments is choosing an adequately large size for the latent vector. Setting the latent vector to be too small has a significant impact on the reconstruction loss. In other works the latent vector is set to have a dimensionality as small as 128, however because of the complexity of the data distribution, larger sizes are chosen for this work. Figure 5.2 shows the range of latent vector sizes that were experimented with in tuning this parameter. It is found that a size of 4096 is suitable to ensure that the reconstruction loss is sufficiently low to produce reconstructions that adequately represent the training data. The *Relu* activation function is used at all layers.

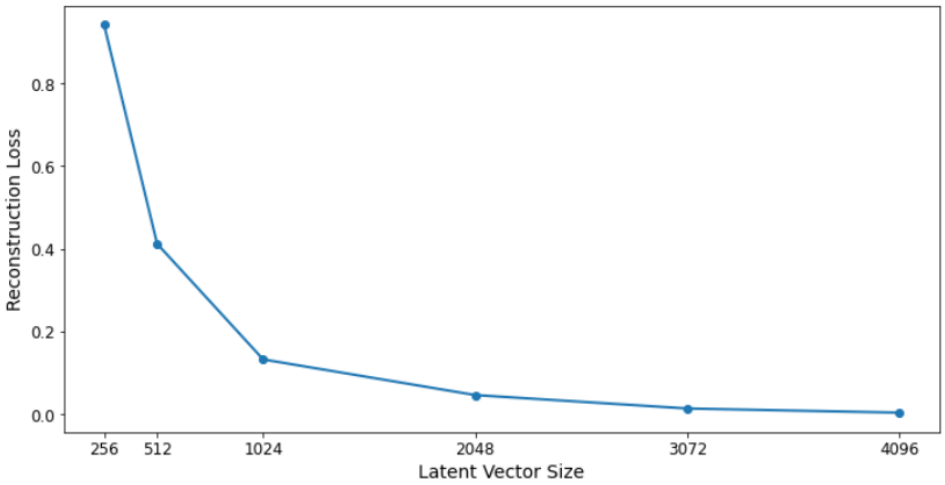


Figure 5.2: Reconstruction Loss and the Latent Vector Size.

By using the architecture in Figure 5.1 as a base, the rest of the model architectures are developed accordingly. The VAE and β-VAE architectures (Figure 5.3) are the same as the autoencoder architecture, except that additional dense layers are used to reparameterise the data for sampling (see section 2.4). The TVAE architecture is the same as the VAE and β-VAE architecture, except that triplet loss is included in the loss function. Lastly, the AAE is extended by adding a discriminator network to the Autoencoder architecture. The prior distribution is chosen as a Gaussian distribution with a mean of zero and a standard deviation of one. It may be argued that the design of these architectures is sub-optimal. Ideally, each of the parameters in the architecture (number of convolutional layers, hidden layer size, choice of activation function etc.) should be parameters that could be tuned in a parameter sweep. The manner in which these values are chosen in this research are not chosen by a parameter sweep, but chosen to use as much of the possible memory on the graphics card that is available (approximately 12GB Cuda memory).

5.3 Model Loss Comparisons

Some statistics on the models are provided on the different losses of the models. The data is split into a training and validation set, where 80% of the data is used as the training set, and 20% of the data is used as the validation set. Tables 5.1 and 5.2 show the loss terms for the training set and test set respectively.

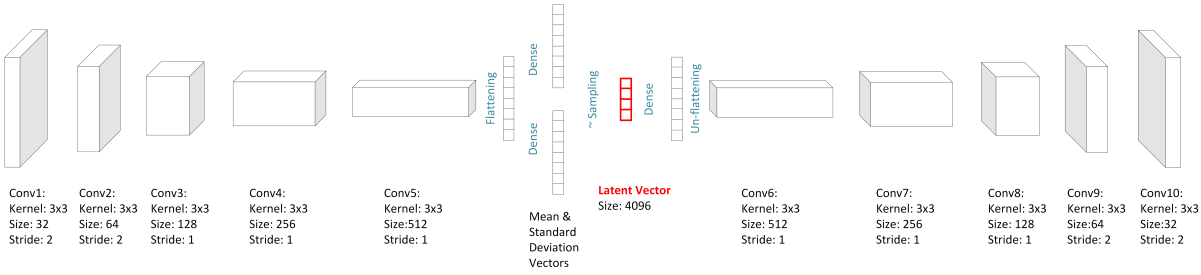


Figure 5.3: Variational Autoencoder and β-VAE Architecture.

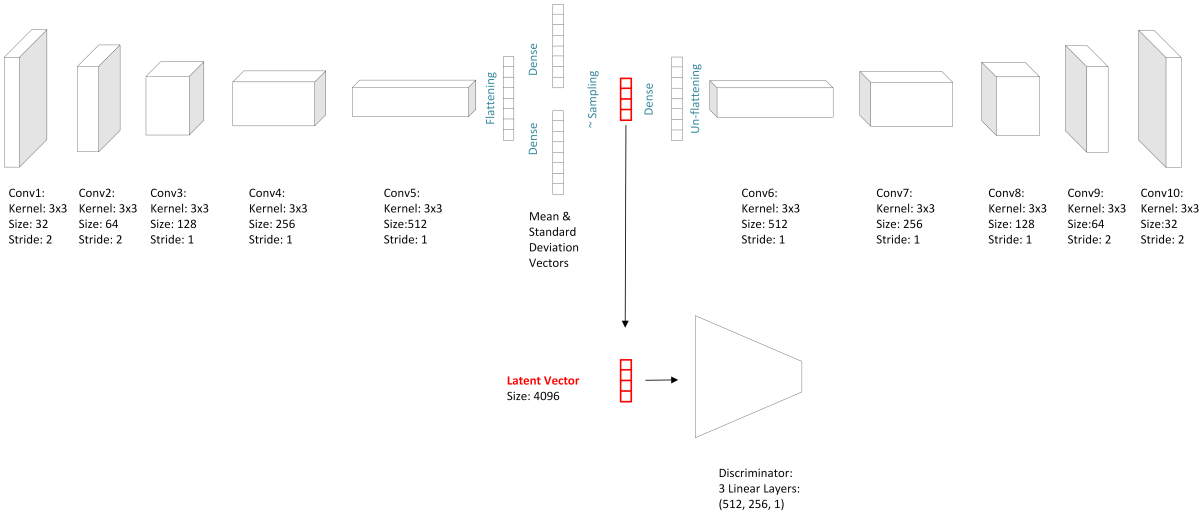


Figure 5.4: Adversarial Autoencoder Architecture.

In table 5.2 a significant insight is that the AE has the lowest reconstruction loss. This is because the other models have some additional loss terms that serve as a trade-off for reconstruction loss. The VAE has the highest reconstruction loss. The effect of β, ($\beta = 0.4$) can be seen in the β-VAE. The reconstruction loss for the β-VAE is significantly better than that of the VAE. Even though the Kullback–Leibler divergence is only 0.01 smaller for the β-VAE compared to the VAE, the effect of this is clear in Figure 5.5. This Figure shows a comparison of the reconstructions with the artwork *Bouquet of Sunflowers*, by Claude Monet (Impressionism).

A concerning result is that of the VAE. A number of different aspects of the training procedure and architecture were varied in order to improve the reconstruction quality. This included, changing the size of the latent space, the design of the convolutional layers, learning rate schedulers, activation functions, and hyper-parameters (learning rate, batch size, image size). It seems as though the Kullback–Leibler divergence term in the loss function is too overbearing for the model to learn better reconstructions. In general, increasing the size of the latent space assisted the models in learning better representations. Because of the poor reconstruction quality of the VAE, the VAE is given somewhat less attention in the findings of the report.

Loss Term	**AE**	**VAE**	**B-VAE**	**TVAE**	**AAE**
Reconstruction Loss	0.0031	0.0070	0.0041	0.0043	0.0061
KLD	-	0.0143	0.0032	0.0064	-
Triplet Loss	-	-	-	0.0077	-
Discriminator Loss	-	-	-	-	0.0131

Table 5.1: Comparison of Training Loss Terms.

Loss Term	**AE**	**VAE**	**B-VAE**	**TVAE**	**AAE**
Reconstruction Loss	0.0042	0.0081	0.0047	0.0051	0.0076
KLD	-	0.0162	0.0069	0.0071	-
Triplet Loss	-	-	-	0.0083	-
Discriminator Loss	-	-	-	-	0.0142

Table 5.2: Comparison of Validation Loss Terms.

5.4 Conclusion

In this section the implementation of the model architecture is presented along with some simple statistics of training the models. These trained models can now be used to explore the latent space. The next Chapter will present the findings of comparing the latent space of the models using statistical methods.

(a) Original

(b) AE

(c) VAE

(d) β-VAE

(e) TVAE

(f) AAE

Figure 5.5: Reconstruction Examples.

Chapter 6

Examining the Latent Space

6.1 Introduction

In this Chapter the latent space is explored for each of the models. Ultimately, in the next Chapter, the goal is to interpolate between regions in the latent space, and to evaluate the latent space by calculating the style loss between pairs of images. However, first there needs to be some interrogation of the latent space, in order to understand the latent space of the different models at a high level. First, Principal Component Analysis (PCA) is performed on the data in the latent space for each of the models in section 6.2. Ideally by using PCA, the structure (if any) of the latent space can be understood. Secondly, possible groupings of the classes (Impressionism, Romanticism, Realism, Expressionism) are investigated in the latent space by using clustering algorithms in section 6.3. The purpose of this is to understand whether particular regions in the latent space correspond to a particular class or classes.

6.2 Principal Component Analysis Experiments

Principal Component Analysis (PCA) is used to understand how the data is distributed in the latent space. PCA is chosen over other techniques such as t-SNE, since the principal components represent the inherent structure of the data, but with t-SNE we cannot determine what latent factors are more important, or if the data is predominantly distributed along one of the principal components [Maaten and Hinton 2008]. Figure 6.1 shows the percentage variance contribution of each of the first 10 principal components in descending order for all the models. This Figure is referred to in the successive sections.

6.2.1 Probing the Latent Space of the AAE

For the AAE, the variance accounted for by each of the principal components is significantly lower compared to the other models (see the orange line in Figure 6.1). The percentage variance contribution of the first principal component is only 2.5%, and the variance contribution of the successive components slowly decreases until about 1% (component 10). This is different to the other models, where the first principal

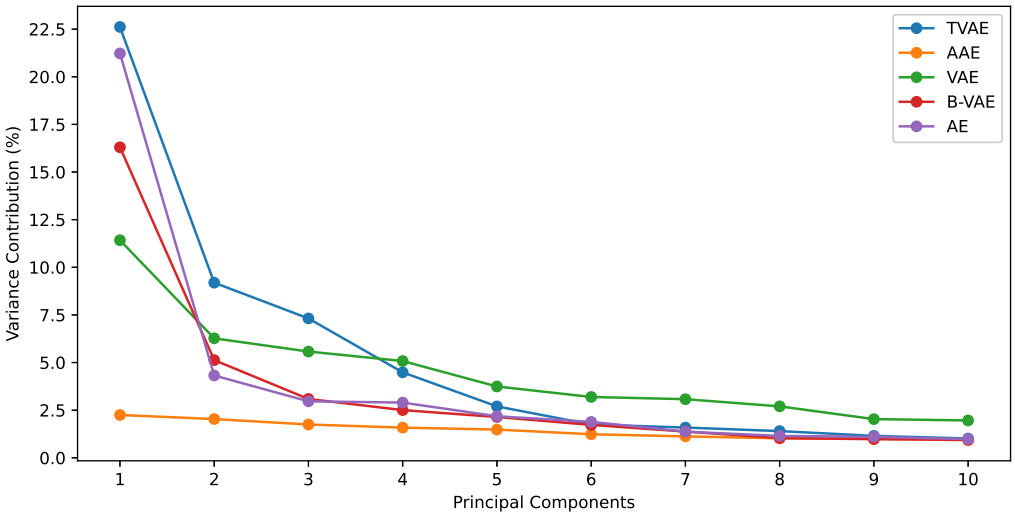


Figure 6.1: All Models - Principal Component Variance Contribution.

component always accounts for at least 10% of the variance. Figure 6.1 evidences an occurrence of a recurring finding in this report. Recall that in the training procedure of the AAE, points in the latent space are encouraged to be approximately normally distributed, because the discriminator tries to distinguish real samples from synthetic samples, where the synthetic samples are drawn from a normal distribution. Ultimately this leads to real samples that are encoded by the encoder of the AAE to be approximately normally distributed. Because of this constraint, the covariance matrix of the data is approximately spherical in the latent space, which implies that the principal components change independently of each other. Because of this spherical covariance, it can be confirmed that the data manifold is approximately spherical in the latent space. This is why the percentage variance contribution of the first principal components of the AAE are so similar, and this also confirms that the AAE is able to successfully mimic the chosen prior Gaussian distribution.

Another PCA experiment is done on the data in the latent space of the AAE, except this time the data is split into four subsets, where each subset relates to the data for a specific class. The results of this are shown in Figure 6.2. The line from Figure 6.1 corresponding to the AAE is included in Figure 6.2 (purple, labelled "Combined") to show how the principal components from the entire population, differ to the principal components from each class. The variance contained in the first 10 principal components of the individual classes is higher than that of the combined classes. In the previous paragraphs it was discussed that when PCA is done on all of the classes together, the data is approximately normally distributed. When the principal components are determined for each class separately, the variance contribution for the principal components is higher, which indicates that each class is not distributed in the same way, and there is no reason to believe that these are normally distributed as the loss function only enforces this structure on the entire population of samples.

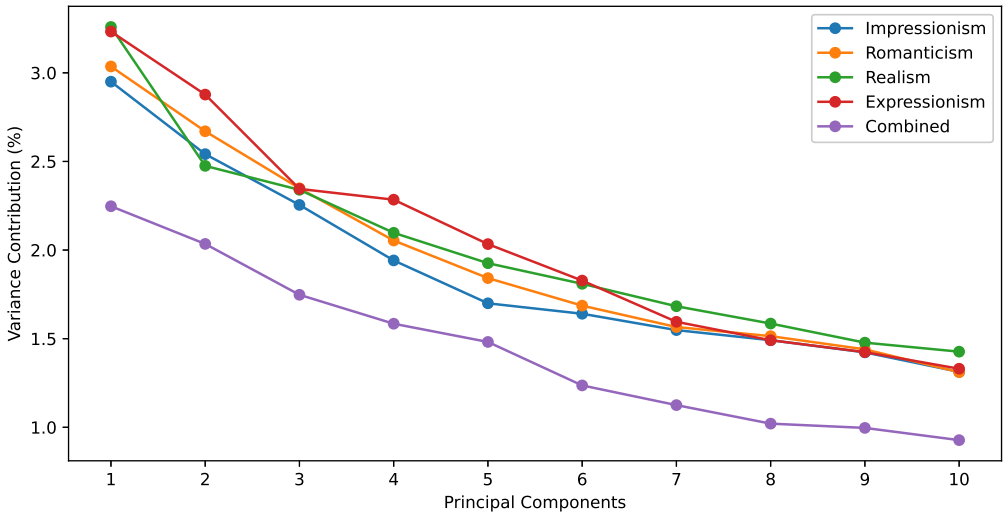


Figure 6.2: AAE - Principal Component Variance Contribution per Class.

6.2.2 Probing the Latent Space of the TVAE

The same procedure followed for the AAE, is followed for the other models. Refer to Figure 6.1 again. This Figure shows the variance contribution for the principal components of all the models. For the TVAE, β-VAE, VAE, and AE, a similar pattern is observed in that Figure, and this pattern is different to that of the AAE. For the TVAE, it is striking that the first principal component accounts for 23% of the variance, where after the variance contribution of each of the components gradually decreases. Because the first component contains 23% of the variance of the data, the data is predominantly distributed along this component in the latent space. This is completely different to what is found with the AAE. With the TVAE, the data has been organised such that there exists a component along which most of the data is distributed, but for the AAE there is not such a component. This indicates that the data distribution for the TVAE in the latent space is approximately oblong in shape.

The variance contribution of each of the principal components for each class for the TVAE is shown in Figure 6.3. The principal components for each class are marginally different to the entire data set (labelled "Combined"). The reason for this difference is possibly because of the triplet loss term in the TVAE. The triplet loss term has grouped samples of the same class together, and pushed samples of differing class apart. Therefore the variance contribution of the principal components for each class is quite similar compared to the entire population. This is investigated further, by determining by how much the centroid of each class is offset from the centroid of the entire data set.

Ideally, a table that shows the coordinates of the centroid (middle point) of each of the four classes in the latent space would be included in this report. Since it is not sensible to present a table with 4096 dimensions, the absolute relative position of these centroids are shown instead. Table 6.1 indicates by how much the centroid of each class is offset from the centroid of the entire data set in the latent space. For each class, the position of the centroid is calculated in the latent space, and this is subtracted from the

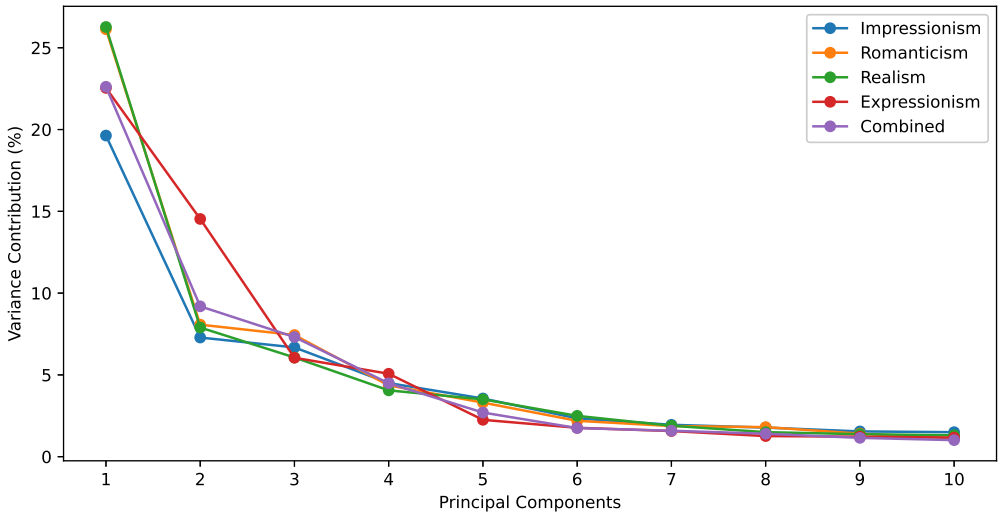


Figure 6.3: TVAE - Principal Component Variance Contribution per Class.

Class	AAE	TVAE	B-VAE	VAE	AE
Expressionism	136	410	335	40	133
Impressionism	111	258	294	26	140
Realism	81	251	187	23	107
Romanticism	92	219	200	23	103

Table 6.1: Relative Position of Classes in the Latent Space.

position of centroid of the entire data set. The absolute value of these distances is then aggregated over the features in the latent space. For the TVAE, it is clear that each class is offset. Therefore even though Figure 6.3 suggests that the principal components are similar, each class has been grouped by the triplet loss function, since the centre of each class is offset from the centre of the entire data distribution. This is further validated in the clustering experiments.

6.2.3 Probing the Latent Space of β-VAE

The other models (β-VAE, VAE, AE) follow a similar pattern to the TVAE, but the variance contribution of the first principal component is not as high as that of the TVAE. For these models it can also be concluded that the data distribution is approximately oblong in shape. A theme that emerges is that there is a certain *order* that is found in the experiments. That is, the β-VAE consistently ranks in between the TVAE and the VAE in various experiments. Recall that, the VAE does not constrain the organisation of the latent space like the TVAE does. The TVAE explicitly enforces that points be organised according to their class. Also recall that with the VAE, there is no weighting of the terms in the VAE's loss function, but with the β-VAE the Kullback–Leibler divergence is weighted by a factor of β.

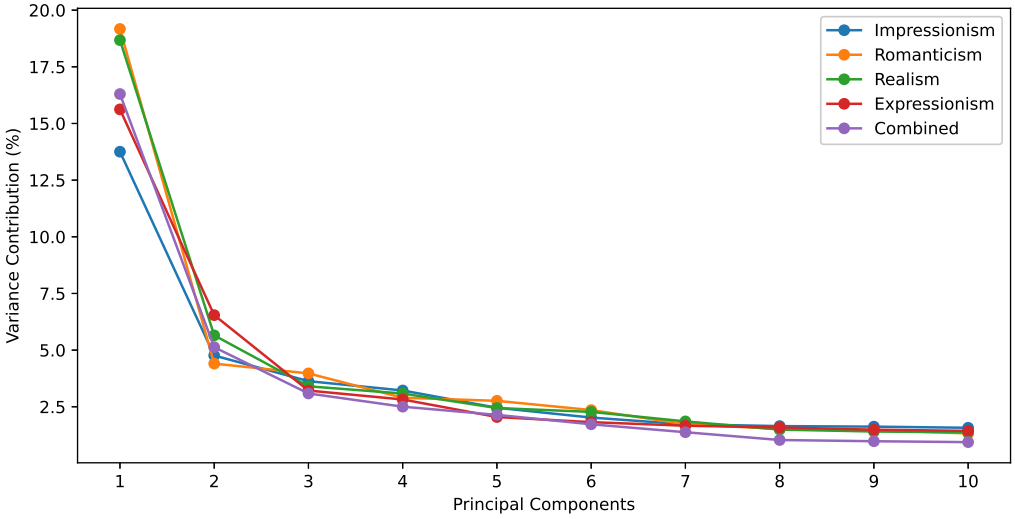


Figure 6.4: β-VAE - Principal Component Variance Contribution per Class.

For the β-VAE, in Figure 6.4 the variance of each of the components is very similar for each of the classes. This indicates that the distribution of each of the classes could be similar. Additionally, the principal components from Figure 6.1 are also shown (purple, labelled "Combined"). Since the principal component variance contribution for each class is so similar to the variance contribution of the principal components for the entire data set, it appears that each class individually is distributed similarly to the entire data set.

The offset of each class for each model is shown in Table 6.1. For the β-VAE, each class is significantly offset in the latent space, compared to the other models. This suggests that there are certain regions in the latent space of the β-VAE that correspond to a particular class, and this is investigated in the clustering experiments. What is surprising is that the offset for the β-VAE is similar to the TVAE. This is investigated further by understanding how the direction of the principal components for each class differ from those of the entire data set.

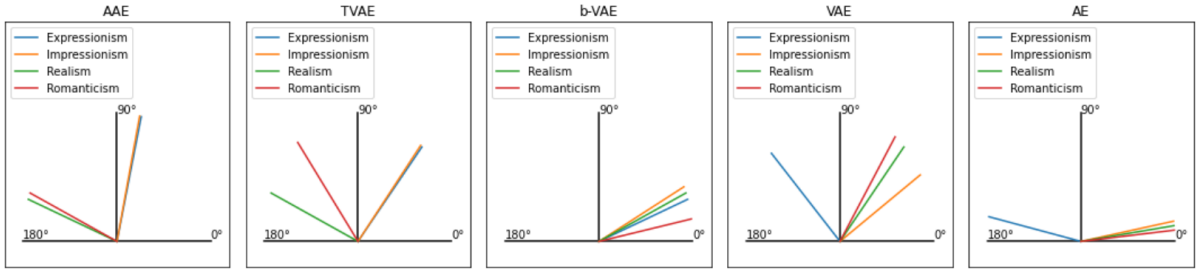


Figure 6.5: Comparison of Basis Vector Directions.

The direction of each of the principal components of the entire data set are compared against the direction of the principal components of each class. Figure 6.5 and Table 6.2 show the angle (in degrees) between the direction of the first principal com-

Class	AAE	TVAE	B-VAE	VAE	AE
Expressionism	75	47	19	137	169
Impressionism	76	48	25	31	9
Realism	161	158	22	47	7
Romanticism	158	130	10	54	5

Table 6.2: Comparison of Basis Vector Directions - Data.

ponent of each class and the first principal component of the entire data set. This indicates by how much the orientation of each class differs to the entire data set, in terms of their first principal components. The motivation for only showing the calculations for the first principal components in Table 6.2, is that a large portion of the variance is attributed to the first principal component, as in Figure 6.1. For the β-VAE, the difference in direction of the first principal component is up to 25 degrees, for each class. This is a relatively small compared to the TVAE, for example. This means that the first principal component of each class is approximately oriented in the same direction as the first principal component of the entire data set. Therefore for the β-VAE, the principal components evidence three things. Firstly, the distribution of the data for each class is similar to the distribution of the entire data set (Figure 6.4); secondly, each class is offset in the latent space (Table 6.1); and thirdly, the first principal component of each class is oriented in a fairly similar direction to the first principal component of the entire data set. This is different to the TVAE, since Table 6.2 shows that the orientation of the principal components of the TVAE are not in the same direction as the principal components of the entire data set.

6.2.4 Probing the Latent Space of the VAE and AE

The principal component variance contribution of the VAE is shown in Figure 6.6. This Figure is comparable to that of the AAE (Figure 6.2), in the way that the slope of the percentage variance contribution gradually decreases. This agrees with what one would expect, since with the VAE, the Kullback–Leibler divergence is not weighted as with the β-VAE, and there is also not a triplet loss term that would cause samples of the same class to be grouped together. The regularisation brought about by perturbing samples in the training procedure of the VAE causes the principal component's variance contribution to be distributed more evenly between the components. There is also no evidence to suggest that groups according to the class label exist in the latent space. Table 6.1 indicates that the centre point of each class is very close to the centre of the entire data set.

An insight concerning Figure 6.1 is that for the AE, the percentage decrease in the contained variance between principal component 1 and 2 is the largest of all the models. This indicates that a significant proportion of the variance of the data is accounted for in the first principal component, and that the other components are not as important. This is expected because for the other models, there is a regularisation term in the loss function, but for the AE there is not. For example, for the VAE and β-VAE, in each training step the points are perturbed by a small amount to regularise the area around each of the data points. Furthermore when the principal components of the AE are

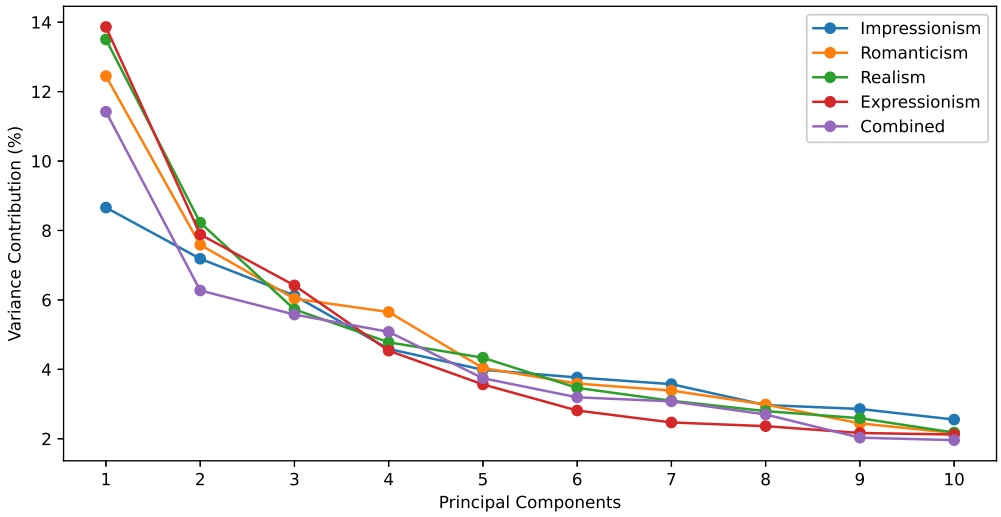


Figure 6.6: VAE - Principal Component Variance Contribution per Class.

examined (Figure 6.7), there is nothing that suggests that there is a grouping of the classes in the latent space. The distribution of each class is very similar in terms of their variance contribution.

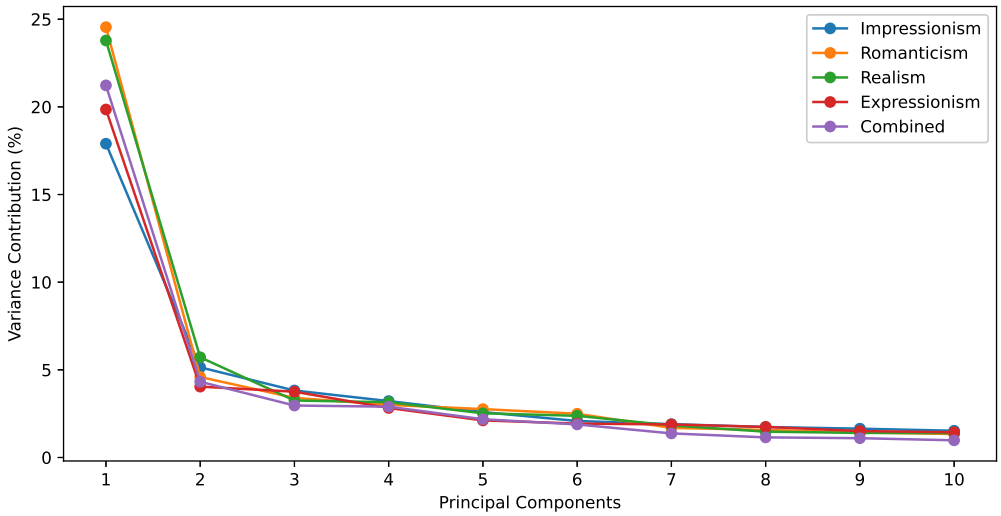


Figure 6.7: AE - Principal Component Variance Contribution per Class.

6.2.5 Conclusion - PCA

In this section the latent space of each of the models was examined by using PCA. The results can be classified mainly in three groups. Firstly, the AAE showed that the

variance is more evenly distributed among the principal components than the other models. However, when PCA is conducted on each class individually, the first few principal components have a higher percentage variance contribution, compared to the principal components of when all the data is together.

Secondly, for the TVAE, the first principal component contains a significantly large portion of the variance of the data, compared to the other principal components. The position of the centre of each of the classes, is significantly offset from the centre of the entire data set. This is evidence to suggest that there are groups according to the class labels in the latent space.

Thirdly, the β-VAE, VAE, AE showed that the principal components for the entire data set are very similar to that of each class individually. Overall these three models could be considered in their own category because it is difficult to detect any class related grouping based on PCA alone.

6.3 Clustering Experiments

Possible groups in the latent space are further investigated by clustering the data in the latent space. In these experiments either the k-means algorithm or spectral clustering algorithm is used. The choice of k in the experiments is chosen between 4 and 8. The choice of k is then verified by using Silhouette scores [Lovmar *et al.* 2005]. For brevity the results of experimenting with different values of k are not shown. However, for the TVAE, AAE, and β-VAE it is found that the choice of $k = 4$ is satisfactory using the Silhouette scores. For the VAE and AE the Silhouette scores were ambiguous, and consequently the clustering results for these models are shown in the Appendix.

6.3.1 Clustering in the Latent Space of the AAE

Figure 6.8 shows the clusters that are identified for the AAE. This Figure shows us the constitution of each cluster in terms of the class proportion, and the size of each cluster. Each colour refers to a different cluster, and each bar refers to a different class in each cluster. The vertical axis (for example, "Impressionism-3") refers to the class names and the cluster number (clusters are labelled 0-3). The bar length (horizontal axis) indicates the proportion of samples that belong to a particular class for each of the clusters. The number indicated alongside the bar indicates the number of samples that belong to each class for each cluster, which quantifies the size of the cluster. The clustering experiments were run multiple times to ensure that the same clusters are consistently found.

A number of observations are made about Figure 6.8. None of the clusters are significantly dominant in one particular class. This agrees with what is stated in section 6.2.1 - because the model tries to encode samples to be approximately normally distributed, the density of points in the latent space is approximately normally distributed. Only cluster 0 has a relatively dominant class. It seems that the Impressionism class is somewhat dominant (in terms of proportion). An interesting insight is that a significant proportion of the data from the Romanticism and Realism classes sits in cluster 1. This is arguably because the stylistic similarities between the classes have grouped

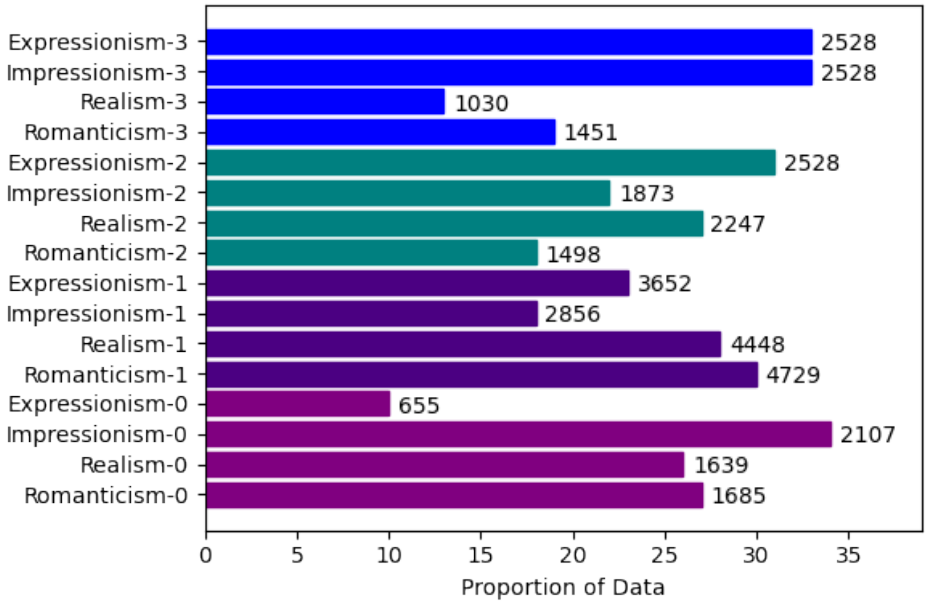


Figure 6.8: AAE - Clustering in the Latent Space.

samples together in the latent space. Therefore, for the AAE, it is difficult to delineate a particular region of the latent space to a particular class in both the PCA and clustering experiments.

6.3.2 Clustering in the Latent Space of the TVAE

The same clustering experiment is repeated for the TVAE. The results of the clustering experiment for the TVAE are shown in Figure 6.9. The results of this experiment are different to that of the AAE, in that some of the clusters are more distinguished in which class they represent. The Expressionism class is dominant in both cluster 0 and 2. Just as with the AAE, cluster 1 shows how a subset of the Realism and Romanticism classes have been paired together. The Impressionism class dominates cluster 3, with approximately half of the Impressionism samples of the entire data set being in this cluster.

In the ideal case the triplet loss function would perfectly separate classes into distinct groups in the latent space. The clustering algorithms also do not perfectly capture the way the triplet loss function has organised the latent space; perhaps there is a clustering algorithm that would yield a clustering result where each cluster more distinctly represents one of the classes. There is also a trade-off between the reconstruction loss and triplet loss terms in the TVAE. By weighting the triplet loss term higher, reconstruction quality would be lower, but the clusters would be more specific to one class. However imperfect the clustering may be, it is still insightful to compare the clustering results between the models. The result for the TVAE in Figure 6.9 is significantly different to that of the AAE. It is clear that the latent space of the TVAE is organised according to the class of the samples to a degree, which is seen in the semi-dominance of some classes in the clusters.

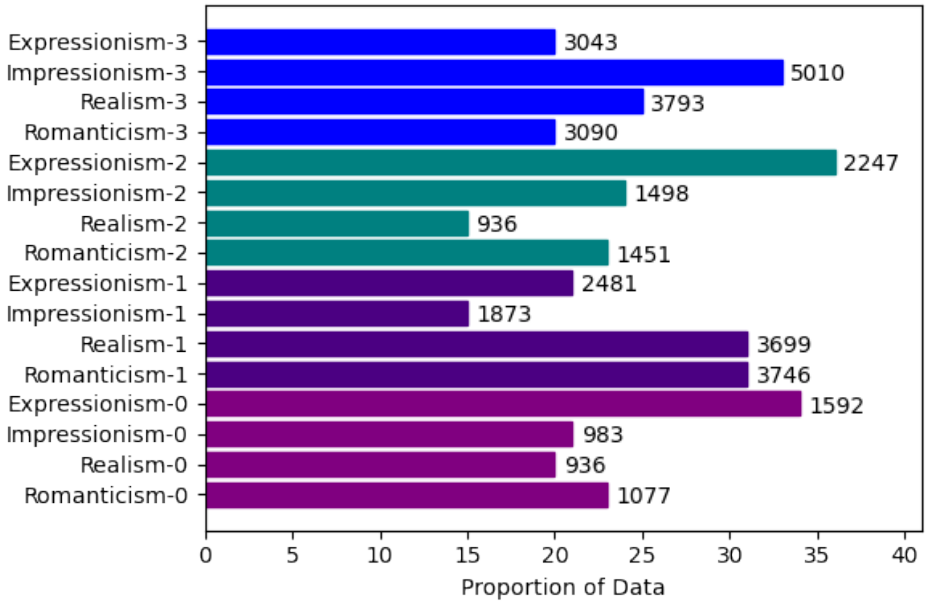


Figure 6.9: TVAE - Clustering in the Latent Space.

6.3.3 Clustering in the Latent Space of the β-VAE, VAE, AE

Figure 6.10 shows the clustering results for the β-VAE. Here cluster 0 and 2 are a mixture of the different classes, and none of the classes are particularly dominant. Cluster 1 mainly consists of the Impressionism class, with almost 50% of the Impressionism samples belonging to this class. Similarly, cluster 3 exhibits a similar behaviour as the clustering for the TVAE, that is, that the Realism and Romanticism class are equally dominant.

This is not a surprising result for the β-VAE, since in the model training procedure, there is no limitation to particularly group samples according to their class, except that there may be similarities in the images themselves, which may cause such samples to be closer together in the latent space. This is different to that of the TVAE since with the TVAE, the triplet loss term explicitly trains the model according to the class labels of the samples. For the VAE and AE, it was not possible to cluster the data in a way that showed that there are at least some areas where a particular class is dominant. The clusters are even more vague than that of the β-VAE. The clustering plots for these models can be found in the Appendix.

6.3.4 Conclusion - Clustering

By doing the clustering experiments, the findings of the principal component analyses could further be investigated. For the AAE, the clustering algorithms could not confirm that there are clear groups in the latent space. Some of the clusters are dominant in one class, but not by a large margin.

For the TVAE, the clusters in the latent space of the TVAE more clearly show that specific regions of the data manifold correspond to a particular class. These regions

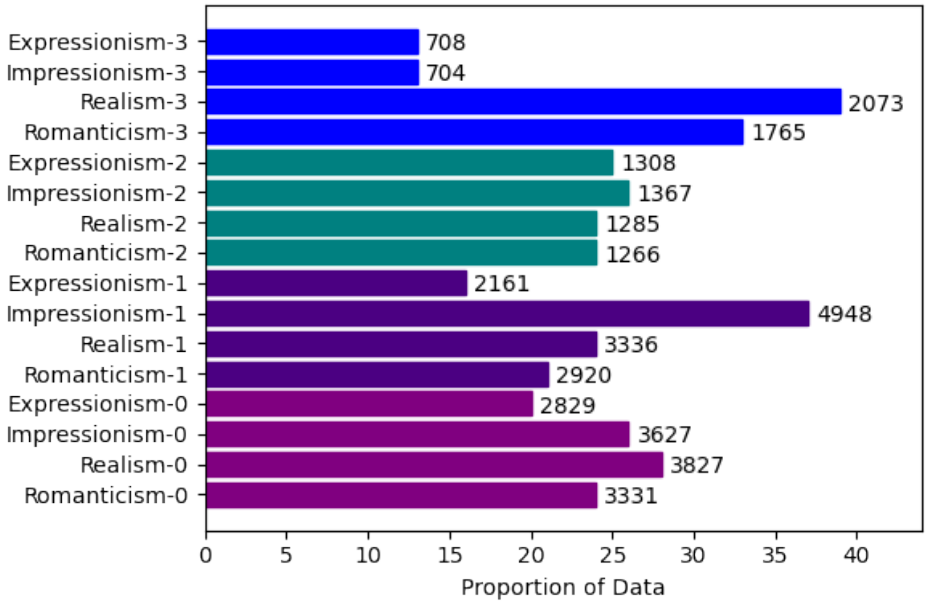


Figure 6.10: β-VAE - Clustering in the Latent Space.

include: a region that has a mixture of the Realism and Romanticism samples, a region where the Impressionism class is dominant, and at least one region where the Expressionism class is dominant.

For the β-VAE, VAE and AE, the clustering experiments provide less compelling evidence that clusters formed around the class labels exist in the latent space. It was anticipated that the clusters would not be comparable to the clusters in the TVAE, since with the TVAE, the model explicitly trains the model using the class labels. There are various reasons as to why this is the case. It could be that an appropriate clustering procedure was not chosen, or it could be that the artistic similarities in the artworks themselves caused the clusters to be formed around those similarities and not the class labels themselves. Another reason could be that some samples may not fully depict the characteristic artistic style of a particular class label; that is, a Realism painting may stylistically look more like a painting from the Romanticism style.

6.4 Conclusion - Exploring the Latent Space

For the AAE, the PCA results confirm that the latent space is approximately Gaussian distributed, which is expected. The clustering experiments could only vaguely identify clusters in the latent space of the AAE. The PCA and clustering experiments for the TVAE show evidence that there are clusters in the latent space that reasonably correlate to a particular artistic style. The β-VAE, VAE, and AE, showed less compelling evidence that clusters exist in the latent space. This does not mean that these models should be discarded from the interpolation experiments, but it helps in understanding why the interpolation results for these models may be significantly different to those of the TVAE and AAE.

Chapter 7

Interpolation & Style Loss Experiments

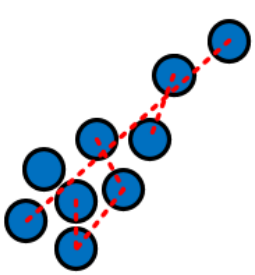
7.1 Introduction

In this chapter the results from the interpolation and style loss experiments are presented and discussed. For brevity, plots for each class and model are not shown. An extensive set of plots can be found in the Appendix. The chapter is organised as follows:

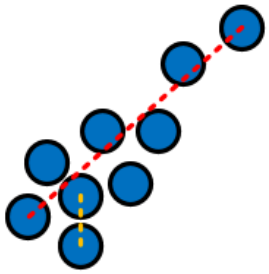
1. **Intra-class interpolation.** These are interpolations between images of the same class. Three experiments are conducted, and an intuitive representation of the experiments are shown in Figure 7.1.
   (a) **Mean style loss trajectories**. 100 random pairs within each class are sampled to understand how style loss generally changes each class. See Figure 7.1a.
   (b) **Shortest trajectories**: This experiment uses samples that are closest together in the latent space. This is to investigate how the style loss changes in short distances in the latent space. See Figure 7.1b.
   (c) **Longest trajectories**. This experiment uses samples that are furthest in the latent space. This is to investigate how the style loss changes over long distances in the latent space. See Figure 7.1b.
2. **Inter-class interpolation.** These are interpolations between images of different classes. The same experiments as for the intra-class trajectories are conducted. However, the longest trajectory experiments are excluded, because in these trajectories it is possible to arbitrarily pass through multiple classes in the latent space, making it difficult to compare the trajectories. See Figure 7.1c.

7.2 Intra-class Interpolation

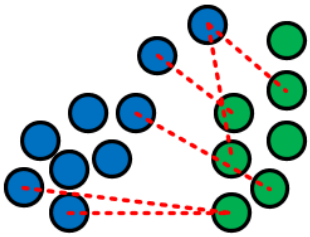
With intra-class interpolation three experiments are conducted. These three experiments are to determine the mean style loss within each class, to interpolate along the shortest trajectory in each class, and to interpolate along the longest trajectory in each class. Distances are calculated using the Euclidean distance in the latent space.



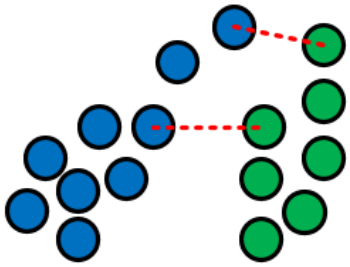
(a) Intra-class: Mean Trajectories



(b) Intra-class: Longest and Shortest Trajectories



(c) Inter-class: Mean Trajectories



(d) Inter-class: Shortest Trajectories

Figure 7.1: Intuition of Style Loss Experiments.

7.2.1 Intra-Class Mean Style Loss

For each of the four classes, 100 random image pairs are chosen. Then for each pair, an interpolation of 10 steps is done in the latent space. Each point in the interpolation is then decoded using the decoder of each respective network. The result is that 11 images are reconstructed using the network, for each image pair. Between each consecutive step along the interpolation, the style loss is then computed. This tells us by how much the stylistic characteristics of the image changes as the interpolation is done in the latent space. The style loss for the 100 pairs is then averaged for each class. This gives a sense of how the style loss generally changes within each class.

Figure 7.2 shows the mean style loss for the 100 random pairs per class. The below points discuss these results:

1. A notable distinction between the models is the shape of the style loss plots. The style loss for the AAE has a *horseshoe* shape. In the first few steps of the trajectory the style loss increases, which implies that the stylistic characteristics of the images are typically lost, and in the second half of the trajectory the stylistic characteristics of the ending image are slowly manifested, causing style loss to decrease. Possible reasons for this are provided in subsequent paragraphs where

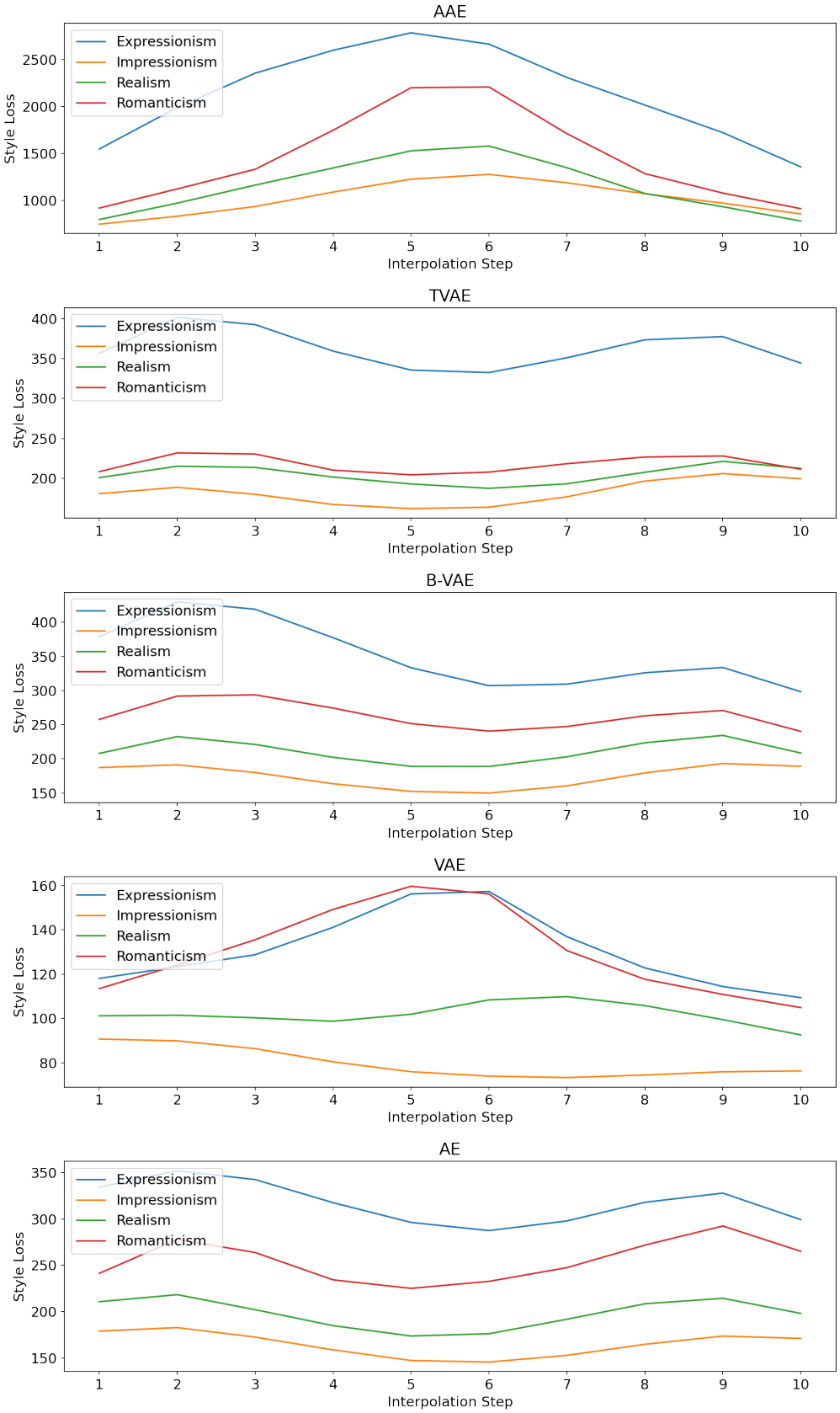


Figure 7.2: Mean Style Loss of 100 pairs per Class.

individual samples are provided (Section 7.2.2).

2. For the TVAE, β-VAE, and AE the shape of the style loss plot is different. For these models the style loss increases and decreases in an *wave* pattern. In steps 1 - 3 the style loss reaches a local maximum. This is because in the first few steps the image becomes blurred, and so some style characteristics (such as texture, for example) are lost. In the middle of the interpolation (steps 4 - 6), the style loss reaches a local minimum. The way this can be interpreted is that the images at this point in the trajectory cannot be identified to a particular style. In most cases, the images show the cross-dissolve effect; that is, that they appear to show elements of both the starting image and ending image at the same time. From steps 7 - 10 a local maximum in the style loss is reached again. The reason for this is the same as for the maximum in steps 1 - 3; that is, the image is becoming less blurred, and more like the ending image in the trajectory. An example of the observed cross-dissolve is shown in Figure 7.3.

Figure 7.3: Cross-dissolve Effect in the Impressionism Class.

3. In all of the plots in Figure 7.2, the order of the classes is always the same. The order is Impressionism, Realism, Romanticism, Expressionism. The mean distance between each of the samples is calculated per class in the latent space. The mean distances are shown in table 7.1. It is interesting to see that the order of the classes in Figure 7.2, is highly correlated to the distances in Table 7.1; that is, it appears as though when samples are more closely situated to one another in the latent space, style loss is lower, and when the samples are further apart, style loss is higher. This is sensible because, if in the latent space samples are close together, that means that the features of the original images are similar; and for images that are similar, it may be that their stylistic characteristics are similar too, which would lead to a lower style loss between the images. Therefore by comparing Figure 7.2 and Table 7.1, style loss is indicative of how close together samples are in the latent space.

Class	**AAE**	**TVAE**	**β-VAE**	**VAE**	**AE**
Expressionism	4238	2941	4580	1378	1926
Impressionism	4091	2438	3994	1349	1707
Realism	3947	2512	4070	1366	1765
Romanticism	3982	2585	4106	1370	1781

Table 7.1: Mean Distances between Samples.

4. The experiment related to Figure 7.2 is also considered in terms of the style loss distribution. Figure 7.2 showed the mean style loss for the 100 image pairs, and Figure 7.4 shows the distribution of the style loss 100 pairs used for the Romanticism class. The plots for the other classes are included in the Appendix. The

observations made about the Romanticism class hold true for the other classes as well. In Figure 7.4 one standard deviation from the mean is plotted with the mean. The standard deviation of the AAE is distinctly different to that of the other models. The scale of the style loss in this Figure is important. The standard deviation is much higher for the AAE than the other models. This indicates that even though there is high variance in the scale of the style loss (for the AAE), this does not mean that the interpolated images are more likely to have a cross-dissolve. Conversely, the standard deviation of the other models is quite small compared to the AAE. β-VAE and AE stand out as the standard deviation in the intermediate steps is smaller than that of the beginning and ending steps. A likely reason for this is discussed with Figure 7.5.

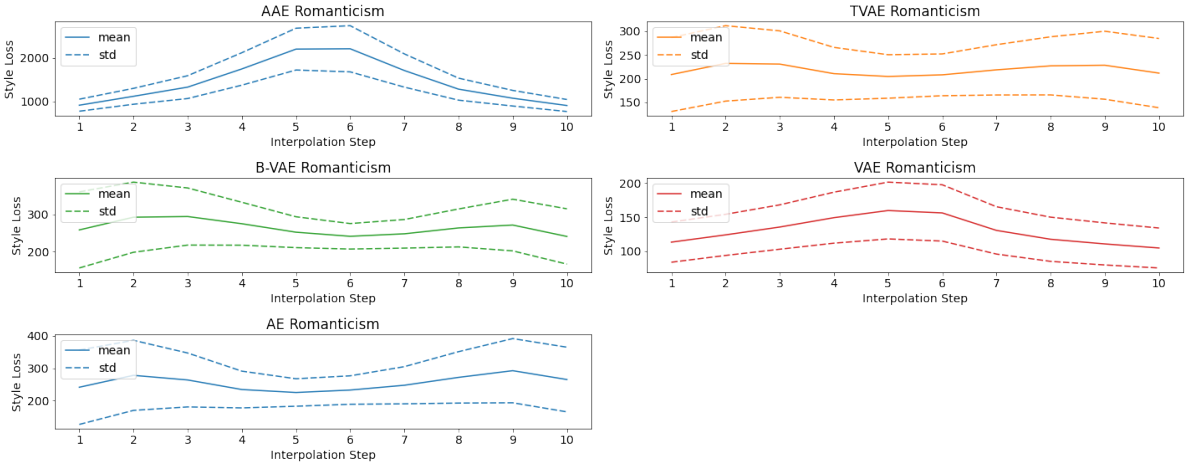


Figure 7.4: Style Loss Distribution of Romanticism.

To better understand the results from Figure 7.2 individual examples are examined with their image reconstructions. Two experiments are done to achieve this. First the image pairs, where the two images in the pair are closest together for each class, are examined. The trajectories for these pairs are termed *shortest trajectories*; since these are the shortest trajectories in each class. These trajectories give insight on how style loss changes as very small steps are taken along an interpolation in the latent space. Similarly experiments for the longest trajectories in each class are conducted, to investigate how style loss changes over larger distances.

7.2.2 Shortest Trajectories

For each of the models, and for each class, the images are identified that are closest together in the latent space. Accordingly there are 20 pairs (5 models, 4 classes) of images where interpolation is done. Figure 7.5 shows the style loss in the Romanticism class along these trajectories. Figure 7.6 shows the interpolated images from which the style loss is calculated.

The following observations are made about the style loss over these trajectories:

1. It is not surprising that often the same pairs of images are used in each of the trajectories for the different models. This is because the features of the artworks are

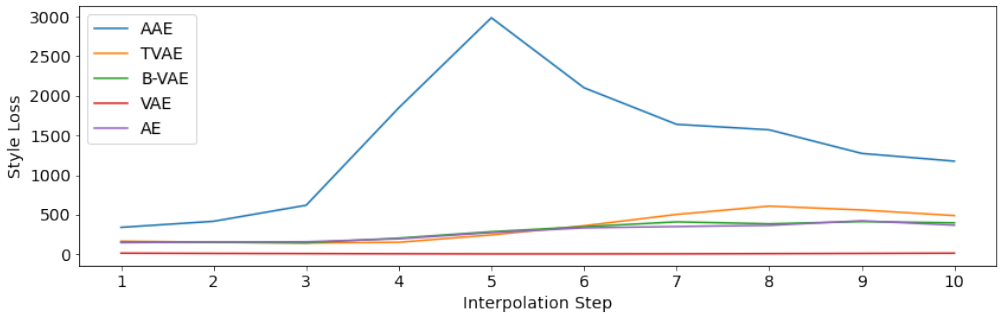


Figure 7.5: Style Loss along the Shortest Trajectories in Romanticism.

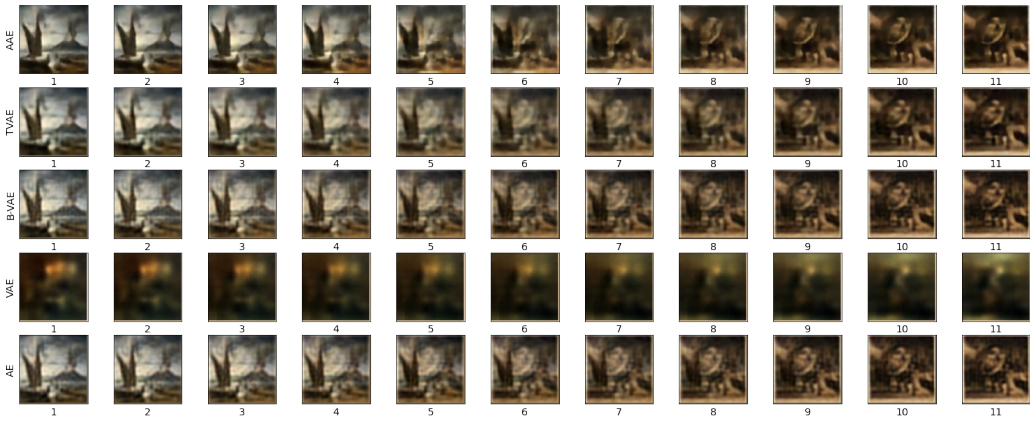


Figure 7.6: Interpolation along the Shortest Trajectory in Romanticism.

similar, and would be closer together in the latent space of each of the respective models.

2. It is clear that the AAE is an outlier. The large spike in the style loss indicates that there is a large change in the stylistic elements of the image along the trajectory. It can be difficult to confirm this by visually inspecting the images in the trajectory. However, if one closely looks at the trajectory reconstructions in Figure 7.6, there are at least two items that distinguish the AAE reconstructions from the others:

   (a) A major finding in the interpolation experiments is that in some interpolations, certain foreign objects are introduced along the interpolation trajectory. Figure 7.7 shows a close up of steps 4-9 of the AAE and β-VAE. For the AAE there is a black shape (see red box) that is introduced into the image. This black shape is neither in the starting or ending image in the interpolation trajectory. As one follows the interpolation the black shape is slowly transformed to take the shape of the face in the last image of the trajectory. This is different to the other models. For example, observe that for the β-VAE the face is slowly faded into the image along the trajectory. It is likely that this is a major contributor as to why the style loss of the AAE is so much higher than that of the other models. In this rest of this Chapter this finding is referred to as a transformational effect in the interpolations.

Therefore it is concluded that the high style loss in Figure 7.2 is because of this transformational effect rather than because of blurring.

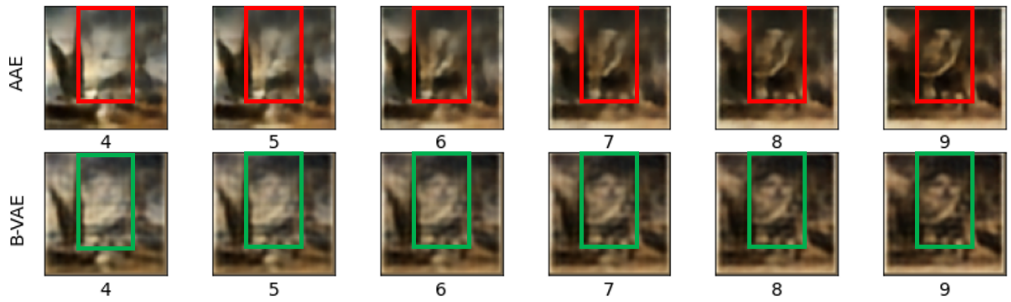


Figure 7.7: AAE - Introduction of Foreign Objects.

(b) Recall that one of the constraints in the AAE architecture is that samples in the latent space are forced to be approximately normally distributed. It is possible that because of this regularisation in the latent space, the model has caused the style loss of the decoded images to be higher.

3. The style loss for the other models is very similar in this experiment. However, a significant difference between these models is that the quality of the image reconstructions for the TVAE, β-VAE, and AE is significantly better than that of the VAE. This is because the Kullback–Leibler divergence term in the VAE is too overbearing for the model to have reconstructions comparable to that of the other models.

A second example of the shortest trajectories is shown in Figures 7.8 and 7.9. This example evidences a recurring finding in the research; that is, that style loss for the TVAE and β-VAE often follow a similar pattern, and the style loss for the VAE and AE also follow a similar pattern. Recall that in section 2.5 and 3.2 the main difference between the loss functions of the TVAE and β-VAE is that the triplet loss function forces samples of the positive class to be closer together while ensuring that the positive class is at least a distance m away from samples of the negative class. That the TVAE and β-VAE have such similar style loss profiles indicates one of two things. Either the triplet loss constraint did not have such a drastic impact on organisation of the latent space, or using style loss is perhaps not the best metric to discern between the latent spaces of the β-VAE and TVAE respectively.

It is interesting that the VAE and AE style loss are similar. One would expect the VAE to have a significantly different style loss to the AE. This is because with the AE the reconstruction loss is found to be significantly lower than that of the VAE. In examples such as Figure 7.8, it is evident from the reconstructions of the VAE that blurred reconstructions do not necessarily imply that the style loss would be high. Instead style loss is inversely proportional to how blurry the images are. This is because, with blurry images the artistic style is not well defined, and evidence of stylistic attributes in the images is vague.

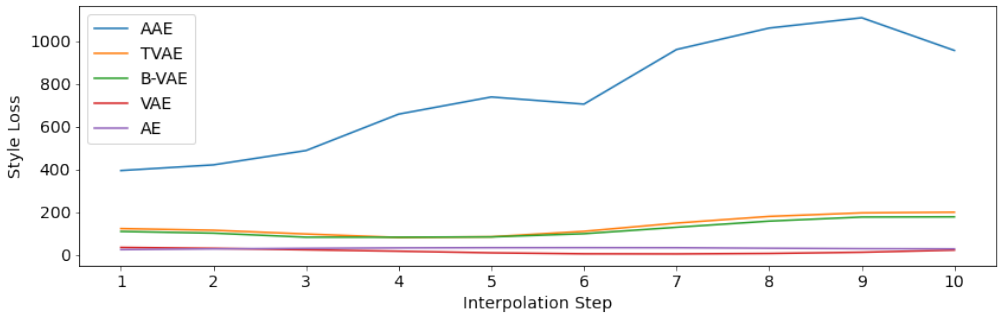


Figure 7.8: Style loss along the Shortest Trajectory in Impressionism.

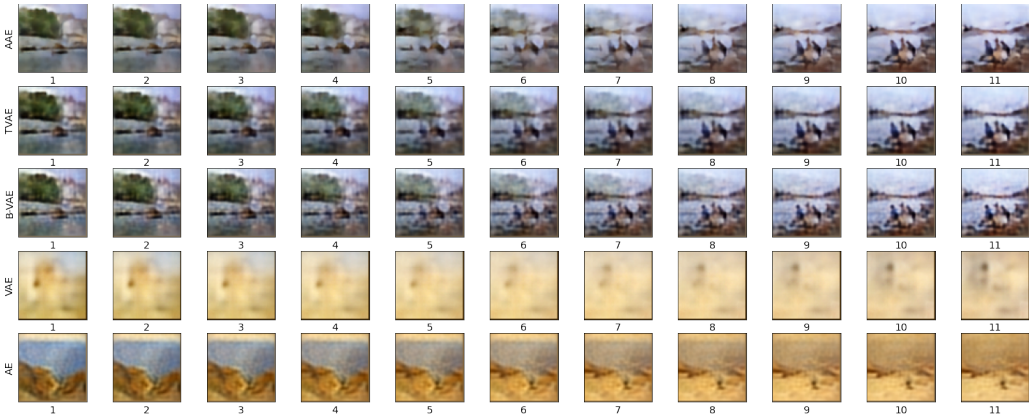


Figure 7.9: Interpolation along the Shortest Trajectory in Impressionism.

7.2.3 Longest Trajectories

This experiment uses samples that are furthest apart in the latent space. This is to investigate how the style loss changes over long distances in the latent space. Figure 7.10 and 7.11 show the style loss and interpolated images respectively for the longest trajectories in Romanticism. The following observations are made concerning the longest trajectories:

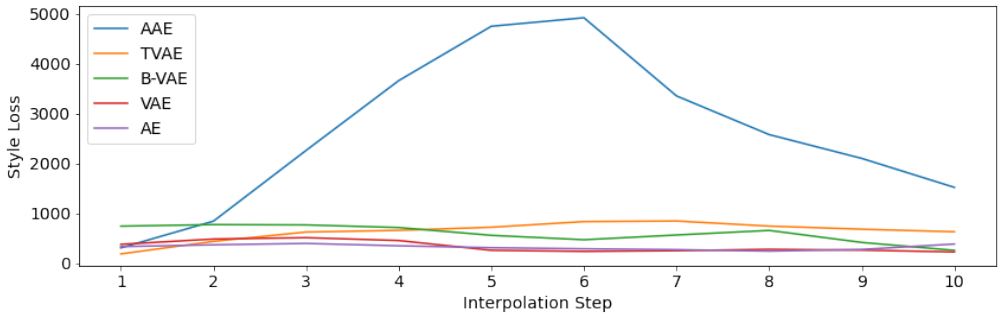


Figure 7.10: Style Loss for the Longest Trajectory in Romanticism.

1. It is common that the images that are used in the longest trajectories have some distinctive characteristics. For example, in Figure 7.11, several of the starting and

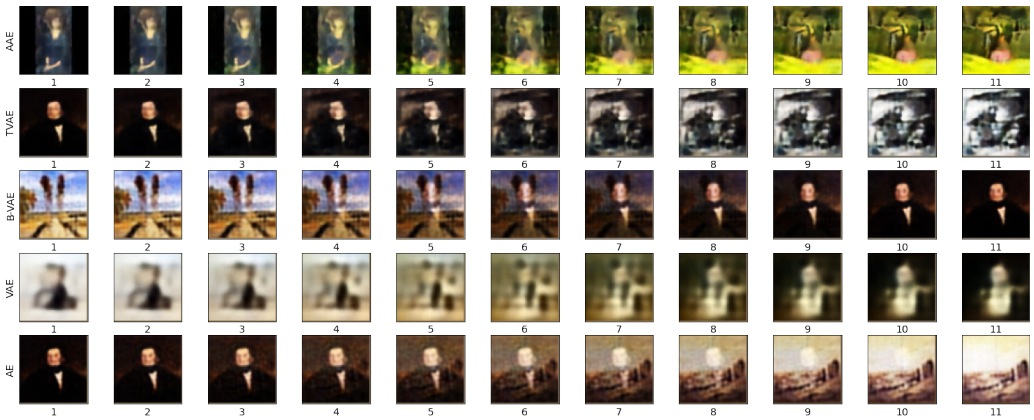


Figure 7.11: Interpolation along the Longest Trajectory in Romanticism.

ending images have a striking use of colour (particularly dark or light). This is because the images are being sampled from the outer regions of the data manifold in the latent space.

2. As with the trajectories in the previous sections, the AAE consistently has a relatively higher style loss compared to the other models. The same reasons as provided in section 7.2.2 are applicable in these experiments.

3. For the other models (AE, VAE, β-VAE, TVAE) the shape of the style loss line graph is relatively flat compared to the AAE. It was expected, since the distance between the starting images and ending images in each trajectory is large, that there would be more evidence that the interpolation has ventured off the data manifold. It would not have been surprising if all of the models showed *spikes* in the style loss plot because of the length of the interpolation trajectory. This is further investigated by conducting an experiment by increasing the number of steps to 100 in the interpolation trajectory to determine if the interpolation ventures off the data manifold. Figure 7.12 shows the same style loss plot with an increased number of steps. For all the models except the AAE, the pattern shown in the style loss is that the variance is even lower, compared to the experiment with 10 steps. From this it is unclear whether the interpolation ventures on or off the data manifold. However with the AAE there is some evidence to suggest that the interpolation as somewhat ventured off the data manifold (shown by the variance in the style loss).

4. It is interesting that the longest trajectory experiments consistently show the cross-dissolve effect in the interpolated images. It was expected that some points in the latent space would be decoded and yield arbitrary noise in the reconstructed images, since the length of these trajectories cover a significantly large area in the latent space. Especially in the case of the AE, where there is no regularisation in the latent space, the images are consistently decoded as a cross-dissolve between the starting and ending images. Because the images do not have arbitrary noise, the style loss does not suddenly fluctuate.

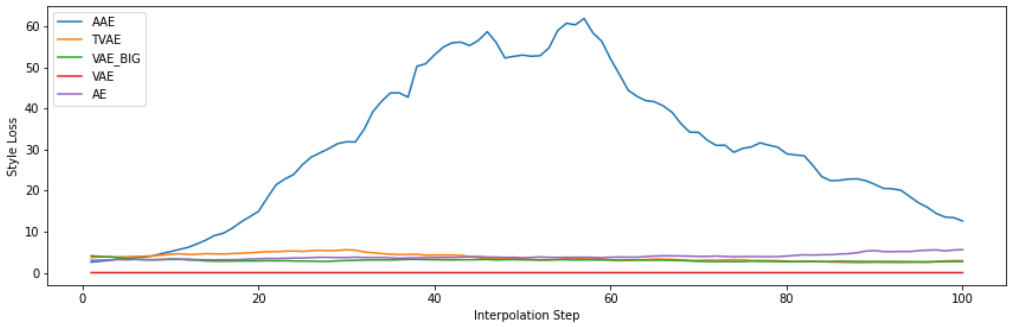


Figure 7.12: Style Loss for the Longest Trajectory in Romanticism with 100 Interpolation Steps.

7.3 Inter-class Interpolation

In the intra-class interpolation experiments style loss is measured within one class at a time. With inter-class interpolation, the interpolations are between two classes.

7.3.1 Inter-Class Mean Style Loss

In this experiment 100 trajectories are sampled from each possible combination of classes (pairs of styles); that is for each combination of Impressionism - Romanticism, Impressionism - Realism, Impressionism - Expressionism, etc. The goal is to investigate the mean style loss between each possible combination of classes.

Figure 7.13 shows the mean style loss over 100 sampled trajectories for each model. Each line on the plot is the mean style loss between two classes. The following observations are made:

1. It was expected that all of the models would exhibit the horseshoe pattern, as discussed in the previous section. This is because, between each of the classes, there should be an area where a point cannot be easily be assigned to one of the classes. Therefore, at this boundary area between classes in the latent space, it was expected that style loss would be at a maximum. From Figure 7.13 it can be seen that this is not the case.

   For the AAE, the shape of the style loss line plot exhibits the horseshoe shape. This is also consistent with what is found in the previous experiments. But the other models (TVAE, β-VAE, VAE, AE) do not evidence the horseshoe shape. Instead the style loss plots have the characteristic wave pattern, which has been consistently observed for these models. There are at least three explanations why the style loss does not increase in the middle of these trajectories, to form the horseshoe shape. Firstly, perhaps the latent space does not have a distinct boundary between the different classes, which is reasonable, since the PCA and clustering experiments were not unanimous in suggesting that clusters in the latent space exist. Secondly, it could be that style loss as a metric is not capable of detecting the boundary between the classes. Thirdly, because the interpolations are linear, there is no guarantee that the interpolation will venture on or off the data manifold at any

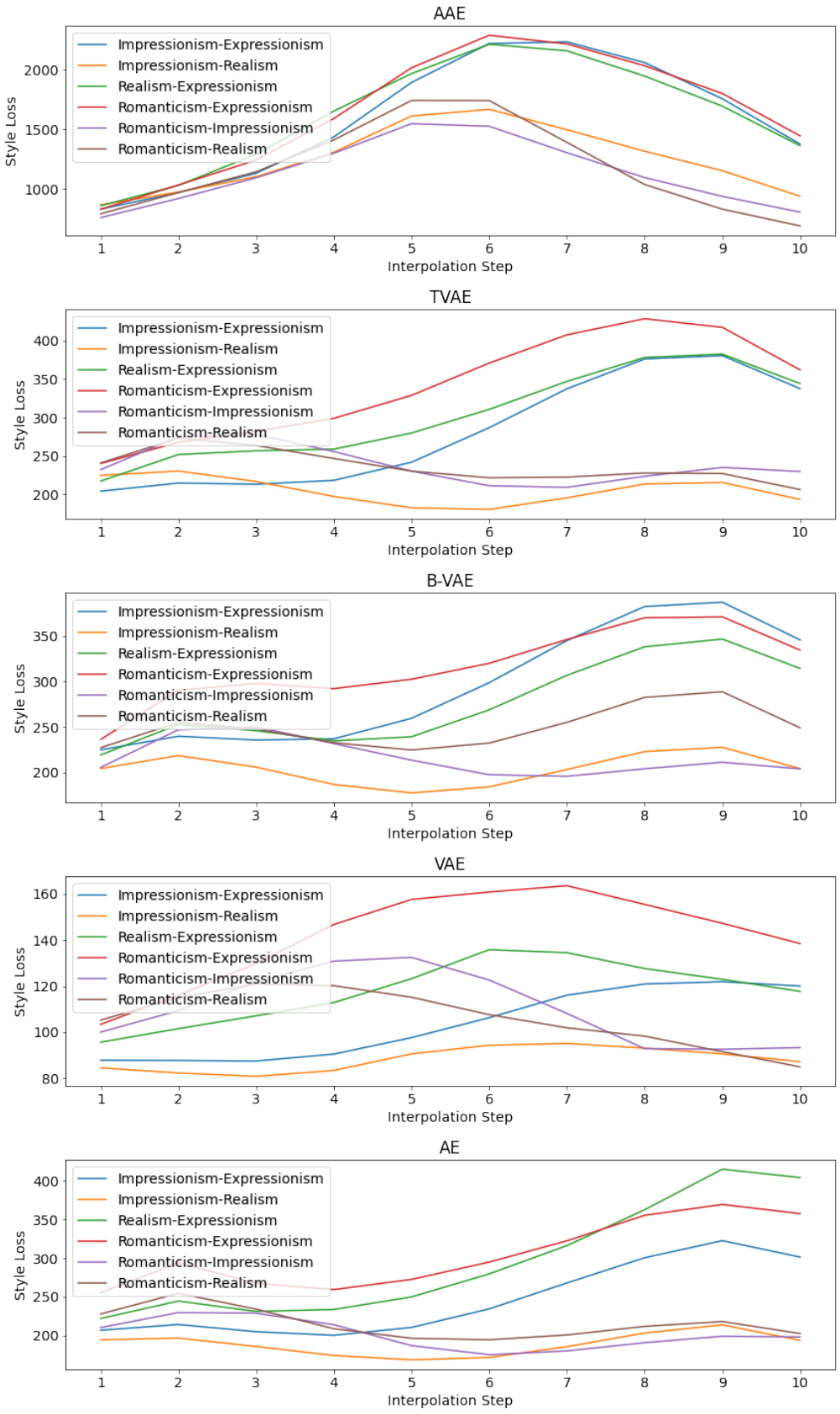


Figure 7.13: Inter-class Mean Style Loss.

point. A major area of further work will be to use geodesic interpolation, where the interpolation is forced to stay on the data manifold [Geng *et al.* 2020].

2. What is consistent between the inter-class experiments and intra-class experiments, is that the wave pattern consistently corresponds to images that evidence a cross-dissolve effect, and the horseshoe pattern consistently corresponds to a transformational effect (see section 2). The reason why a cross-dissolve would lead to a low style loss is because the interpolated images evidence a mixture of the starting and ending images. Therefore because the stylistic characteristics are ambiguous in the cross-dissolve images, the style loss is low. Figure 7.15 is an example where the AAE has a cross-dissolve in the interpolated images. The style loss in Figure 7.14 also consequently evidences the wave pattern rather than the horseshoe pattern.

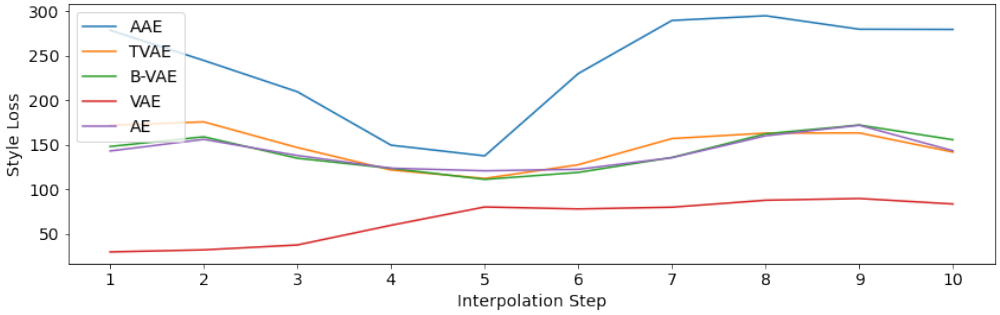


Figure 7.14: Style Loss between Realism and Impressionism.

Figure 7.15: Interpolation between Realism and Impressionism.

3. For all of the models except the VAE, the Expressionism class prompts a particular style loss response. In Figure 7.13 the green, blue and red lines are all related to the Expressionism class. For all of the models except the VAE, when the interpolation approaches the Expressionism class, the style loss is increased (see steps 7 -10 for those models). This indicates that the region in the latent space corresponding to the Expressionism class is distinctly different. The mean distance between samples is shown in Table 7.1. The Expressionism class consistently has a higher

mean distance across all the models. This means that the Expressionism samples are distributed more sparsely than the other classes. It is arguable that because the Expressionism class is distributed more sparsely that the areas on the data manifold between these samples are not as regularised compared to the other classes, and therefore the style loss is higher. This is an insightful finding in the use of style loss since the plots clearly show that the Expressionism class is distinguished from the other classes. This insight might not have been discovered if the latent space was only explored with t-SNE plots, or viewing the reconstruction samples.

4. The distribution of the style loss for the Romanticism-Expressionism experiment is shown in Figure 7.16. The distributions for the other inter-class experiments are shown in the Appendix. The distribution of style loss is comparable to the intra-class experiment (Figure 7.4). By investigating individual samples from this experiment in the next section, it is confirmed that the standard deviation is smaller when the interpolated images are cross-dissolved.

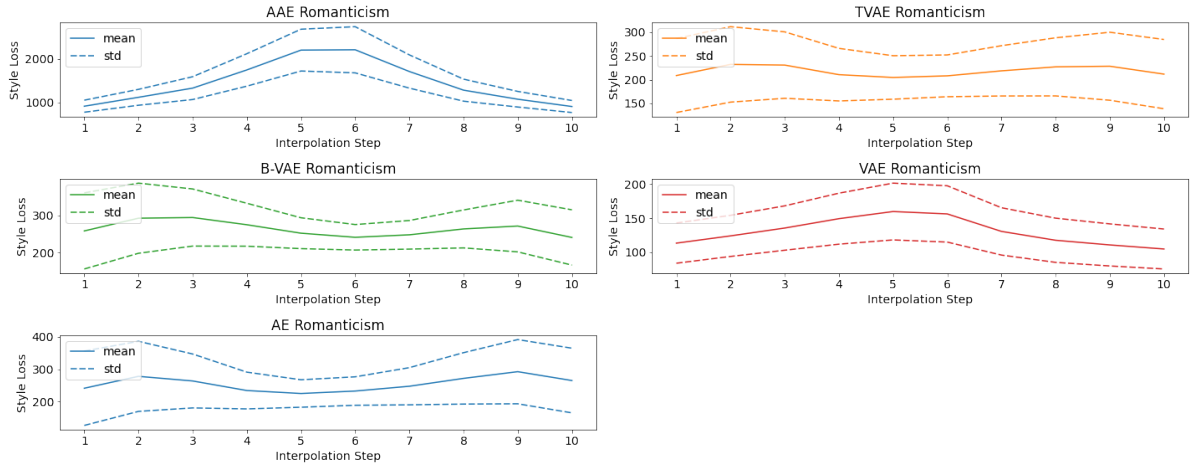


Figure 7.16: Style Loss Distribution between Romanticism and Expressionism.

7.3.2 Shortest Trajectories

For the inter-class trajectories, the longest trajectories between two classes are not included in the experiments. This is because the longest trajectories could arbitrarily pass through multiple classes in the latent space and it would be difficult to compare such trajectories to one another. The shortest trajectories are the trajectories between each pair of classes, where the distance is shortest. See Figure 7.1d for the intuition of this experiment. An example of the shortest trajectories between classes is shown in Figures 7.17 and 7.18.

What is striking in this example is the content of the artworks that are in the shortest trajectories for each of the models. Consider carefully the reconstructed images for the TVAE. The shapes of the objects in the starting and ending images are indeed very similar. In this example the cart on the beach is similar in shape and position to the

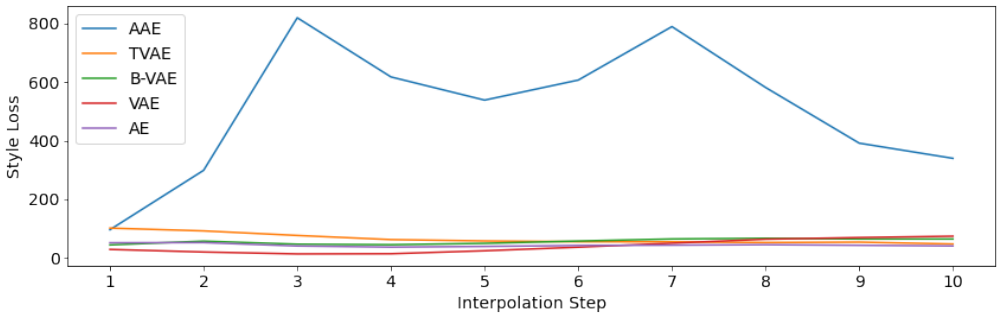


Figure 7.17: Style Loss for the Shortest Trajectory between Impressionism and Romanticism.

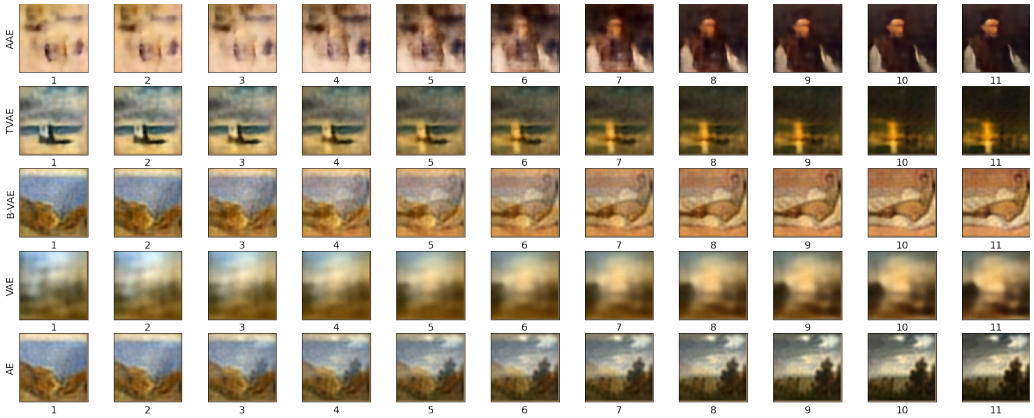


Figure 7.18: Interpolation along the shortest Trajectory between Impressionism and Romanticism.

sunset over the lake. This indicates that the latent space is strongly organised around the content of the images and not necessarily stylistic characteristics.

A second observation here is that the style loss of all the models except the AAE is very flat. By inspecting the images in the interpolation, the images appear to be a near perfect cross-dissolve between the starting and ending image. This suggests that when there is only a small change in style loss along the interpolation trajectory, the image interpolation is approximately linear.

7.4 Conclusion - Style Loss Experiments

In this Chapter the results from the interpolation experiments are presented. Through reviewing several examples, a number of insights could be observed about the latent space and the data manifold, which answers research questions 2 and 3. The main insight that is seen in this Chapter is that there are distinct patterns in the style loss plots. There are two distinct patterns. The first pattern is that of a wave-pattern, which is mainly seen in the AAE, and corresponds to the case where there is a transformational effect in the image interpolation. The second pattern is that of a wave-pattern. This corresponds to the case where there is a clear cross-dissolve between the starting and

ending images in the interpolation trajectory. Therefore it can be concluded that the pattern observed in style loss correlates to the type of transformation that the images undergo along the interpolation trajectory. This uncovers a fundamental difference between the latent spaces of the AAE and the other models. Arbitrary points on the data manifold appear to maintain more structural information in the decoded images for the AAE, whereas arbitrary points on the data manifolds of the other models merely yielded cross-dissolves in the decoded images. A summary of the insights is given in the concluding Chapter of this report.

Chapter 8

Conclusion

8.1 Introduction

This Chapter summarises the contents and findings of the research report. This is done by providing a summary of the research procedure, crystallizing the main insights, discussing assumptions, limitations and further work, and finally concluding the report.

8.2 Summary of Research Procedure

In this research report, it is argued that representation learning is an important component of machine learning. In representation learning the latent space is an abstracted domain of the input data, where the data has been embedded. Autoencoders provide an effective means of learning abstract representations of data because they are evaluated in terms of how well they can decode points from the latent space to match the original data. The goal of the research is to use style loss to compare the latent space of five autoencoder models (AAE, TVAE, β-VAE, VAE, AE). In the experiments a data set of famous artworks is used, and the artistic style (Impressionism, Expressionism, Realism, Romanticism) is used as the class labels.

In training the models, the VAE has the highest reconstruction loss (poorest reconstructions), and the AE has the lowest reconstruction loss (best reconstructions). The AE has the lowest reconstruction loss because the loss function only contains one loss term - the reconstruction loss. Therefore the model could minimise the optimisation problem based only on this loss term. For the VAE, the Kullback–Leibler divergence term in the loss function is overbearing, causing the reconstruction loss to be higher. Consequently it is found that the VAE reconstructs images poorly.

Before doing experiments with style loss, an initial investigation into the latent space of the models is done using two methods, namely PCA and clustering algorithms. By using PCA the distribution of the data can be understood on a high level. Clustering experiments are conducted to further investigate whether or not the data is grouped according to the class labels in the latent space. These experiments are done in order to guide the way in which image pairs are chosen for the style loss experiments. The findings for each of the models in the PCA and clustering experiments are:

1. **AAE.** The principal components of the data in the latent space showed that the data approximately has spherical covariance. This confirms that the AAE is able to learn the parameters such that the distribution of the data in the latent space matches the chosen Gaussian distribution (mean of zero, standard deviation of one). There is also some evidence given by the clustering experiments that groupings according to artistic style exist in the latent space (Figure 6.8).

2. **TVAE.** The TVAE did not exhibit spherical covariance in the PCA experiments like the AAE. Instead, the first principal component accounted for most of the variance in the data, indicating that the data is approximately oblong in its distribution (Figure 6.3). The clustering results showed that the clusters are more distinct in the latent space, as some of the clusters were predominant in one class (Figure 6.9).

3. **β-VAE, VAE, AE.** These models exhibited similar behaviour in both the PCA experiments and clustering experiments. In the PCA experiments, the principal components are comparable to that of the TVAE, but the clustering results only vaguely indicate groupings according to the class labels. Even though this is the case, this does not exclude these models from the style loss experiments, but it does provide some context as to how the latent space is organised (Figures 6.4, 6.6, 6.7, 6.10, A.1, A.2).

The focus of this research is on using style loss to quantitatively make measurements in the latent space to understand the latent space better. The way in which this is done is to choose pairs of points in the latent space, forming a trajectory between those points using linear interpolation, decoding each point in the trajectory, and then calculating the style loss for each pair of consecutive points in the trajectory. The style loss is then plotted on a line graph. Several experiments are designed to probe the latent space with various trajectories. The experiments are broken into two categories, namely, intra-class trajectories (within one class), and inter-class trajectories (between two classes). Rather than review the results of each of these experiments in this Chapter, the results are condensed into key insights presented in the following section.

8.3 Style Loss Experiments: Key Insights

The key insights in the research are as follows:

1. Two patterns in the style loss over the interpolation trajectories are observed. The first pattern, mainly seen in the AAE and VAE, is that the style loss takes a distinctive horseshoe shape (Figure 7.2 AAE), and the second pattern is that the style loss takes a wave like shape (Figure 7.2 TVAE). By investigating the image reconstructions, it appears that the horseshoe pattern occurs when the image is transformed along the interpolation, and the wave pattern occurs when there is a cross-dissolve between the images. The way this is interpreted is that when an image is cross-dissolved, the image mixes the artistic characteristics of more than one style, and because the stylistic characteristics are vague, the style loss is

lower in the middle of the trajectory. However, with the horseshoe pattern, the opposite occurs - the style loss is higher because style characteristics are gradually transformed along the interpolation trajectory. In light of this finding it may be argued that when the horseshoe pattern is observed, the interpolation remains on the data manifold, but when the wave pattern is observed, the interpolation ventures off the data manifold. The motivation for this is that it is expected that interpolation on the data manifold would cause the image to be transformed, rather than cross-dissolved.

2. Style loss is often correlated with the density of points in the latent space. Table 7.1 shows the mean distance between samples in the latent space for each class. The more sparsely distributed points are for a class, the higher style loss is for that class. See Figures 7.2 and 7.13.

3. The magnitude of the style loss of the AAE is often significantly higher compared to the other models. Reasons for this could be that the AAE typically transforms images, rather than merely cross-dissolving them (point 1), or perhaps it is because the latent space of the AAE is approximately normally distributed, which is due to the training procedure of the AAE.

4. The effect of the triplet loss term cannot be easily observed in the style loss plots. Particularly, it is found that the style loss for the TVAE are often similar to that of the β-VAE. Perhaps this indicates that the triplet loss term did not have such a big effect during the training of the model. However, since the interpolations are linear interpolations, and there is no grantee that the interpolation trajectory remains on the data manifold, the effectiveness of the triplet loss term cannot be ruled out. If the interpolations are geodesic, it is anticipated that the style loss of the TVAE trajectories would be further distinguished from that of the β-VAE.

5. The style loss of the intra-class and inter-class trajectories are quite similar. It was expected that the style loss would be higher for the inter-class interpolations, since there may exist regions between classes in the latent space that are not regularised; that is, that decoding those points would give an arbitrary image. However this is not the case. At least three reasons for this are plausible. Firstly, perhaps the latent space does not have a distinct boundary between the different classes, which is reasonable, since the PCA and clustering experiments were not unanimous in suggesting that clusters in the latent space exist. Secondly, it could be that style loss as a metric is not capable of detecting the boundary between the classes. Thirdly, because the interpolations are linear, there is no guarantee that the interpolation will venture on or off the data manifold at any point.

8.4 Assumptions

An assumption in this research report is that within each class (Expressionism, Impressionism, Realism, Romanticism), there are stylistic similarities. This motivates why the TVAE uses the class information during training. However, the artistic style with each

class is disparate, and it may well be that the artistic characteristics of images from one class, are actually stylistically similar to another class.

8.5 Limitations

The following are the limitations of the research:

1. A major contributing factor to the quality of the reconstructions is the capacity of the models during training. The most limiting factor in the research is the size of the latent space. In the experiments a latent space of size 4096 is used, but a number of other sizes were also attempted (1024, 2048). Increasing the size of the latent space beyond 4096 would cause the hardware available to run out of memory. It is anticipated that experimenting with even higher dimensionality may improve the reconstruction results.

2. Similarly to item 1 the images are resized to a size of 64 x 64. In the work of Makhzani *et al.* [2015], their adversarial autoencoder uses images of size 32 x 32, but the data distribution is more simple than the data used in this research report. It is common in computer vision to use images of size 128 x 128, 512 x 512, or even larger. One may argue that a lot of stylistic content of the images is lost in making the images small, but this was done to allow for a larger latent space, since the hardware available could not cope with larger image sizes.

8.6 Further Work

The following items could serve as possible extensions to this research project:

1. A major factor in the design of the interpolation experiments is that linear interpolation is used to define a trajectory between two points. However, by using linear interpolation it is likely that the interpolation ventures off the data manifold in the latent space. Arguably the most important way of furthering this research is to interpolate on the data manifold, as this would significantly change the trajectories, and in turn change the style loss.

2. An interesting area of research would be to generate interpolations while training the models, and then by using a discriminator network, ensure that interpolated images cannot be distinguished from images generated from points in the training data. This would ensure that interpolations would be on the data manifold, and possibly improve the generative capabilities of the decoder network. This could also ensure that the data manifold is convex, reducing the number of images that have a cross-dissolve.

3. By further improving the network architecture (bigger convolutional layers, bigger linear layers), the quality of the image reconstructions could further be improved.

4. It would be interesting to do similar style loss experiments on a data set that does not necessarily contain artworks. Since style loss is calculated by extracting the feature responses from VGG19, there is no reason why this cannot be done.

5. Another area of interest would be to work with an art expert while conducting the experiments. For example, in clustering the data in the latent space, could an art expert validate whether or not it is sensible for certain artworks to be grouped together? Perhaps they would be able to give insight on why certain points are close together, or why a certain point is an outlier in the data set. Their input could also assist in choosing which image pairs would be sensible to interpolate between in the latent space.

8.7 Conclusion

This report has shown that style loss can be used to explore the latent space. This was done by training five autoencoder models on a data set of artworks. A major finding is that two patterns of behaviour is observed in the style loss. The first pattern is a horseshoe pattern, which correlates to a transformational effect in the image interpolation; the second pattern is a wave pattern, which correlates to a images that evidence a cross-dissolve. Several other insightful findings are also discovered in the various interpolation experiments. The main extension of this research report would be to use geodesic interpolation rather than linear interpolation.

Appendix A

Additional Plots

A.1 Additional Clustering Plots

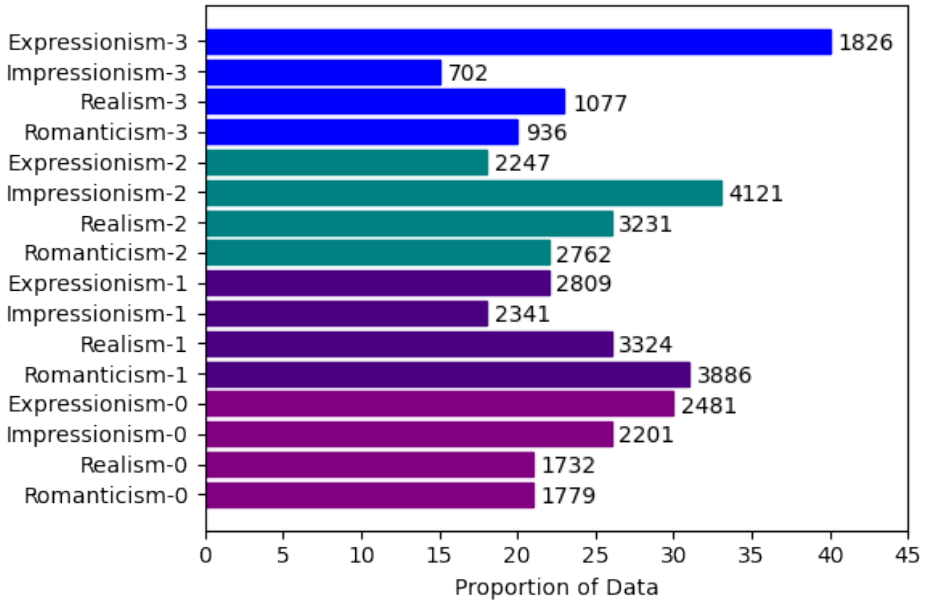


Figure A.1: VAE - Clustering in the Latent Space.

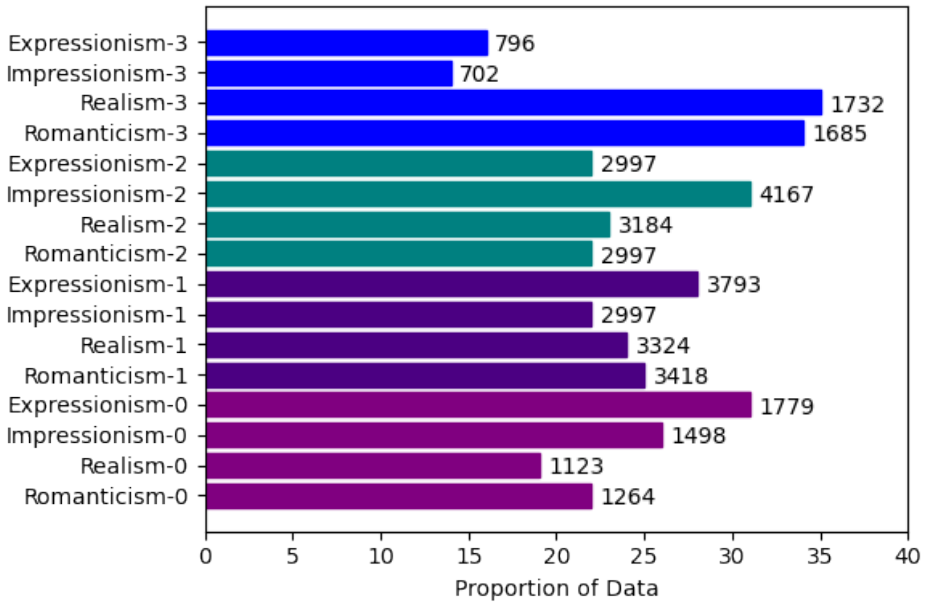


Figure A.2: AE - Clustering in the Latent Space.

A.2 Additional Style Loss Plots

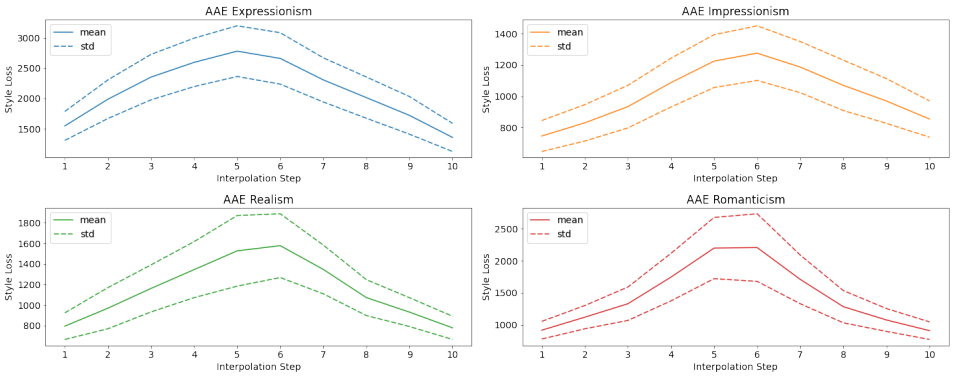


Figure A.3: Intra-class Style Loss Distribution of AAE.

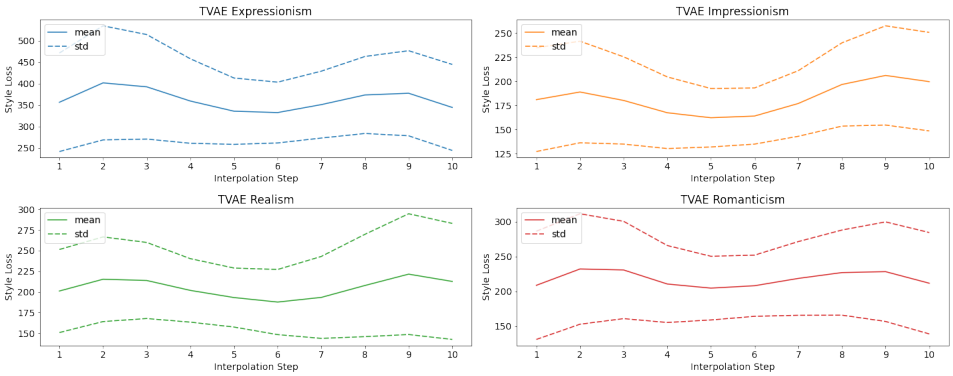


Figure A.4: Intra-class Style Loss Distribution of TVAE.

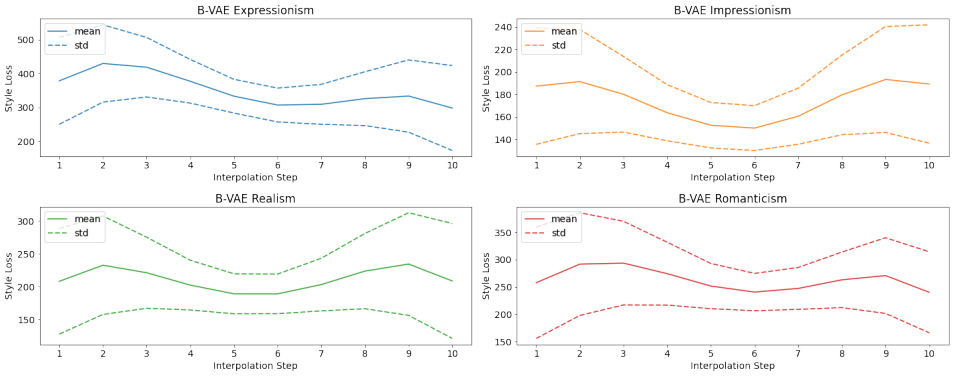


Figure A.5: Intra-class Style Loss Distribution of β-VAE.

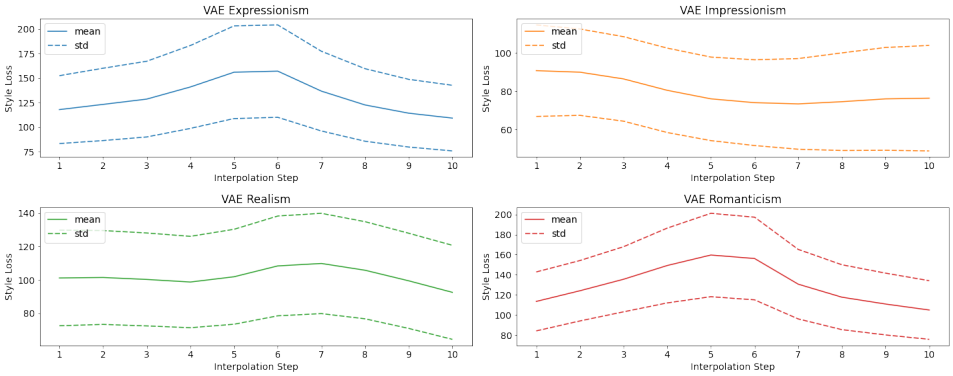


Figure A.6: Intra-class Style Loss Distribution of VAE.

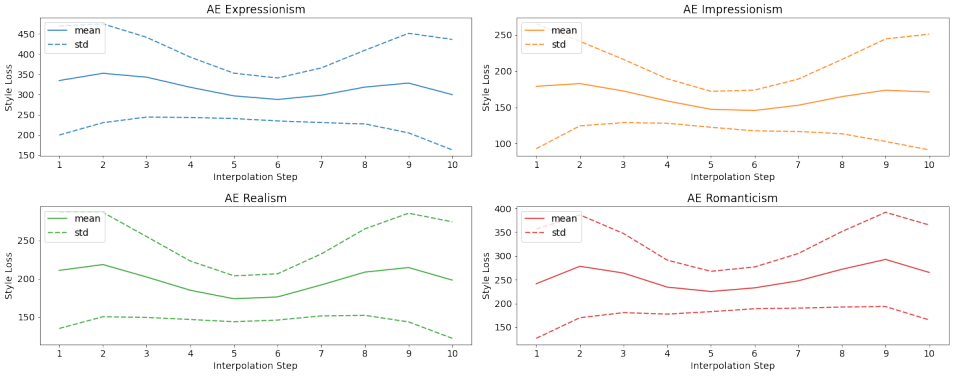


Figure A.7: Intra-class Style Loss Distribution of AE.

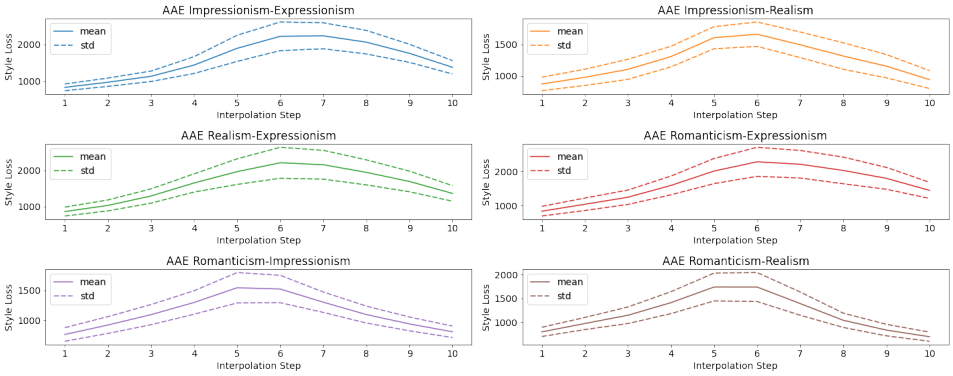


Figure A.8: Inter-class Style Loss Distribution of AAE.

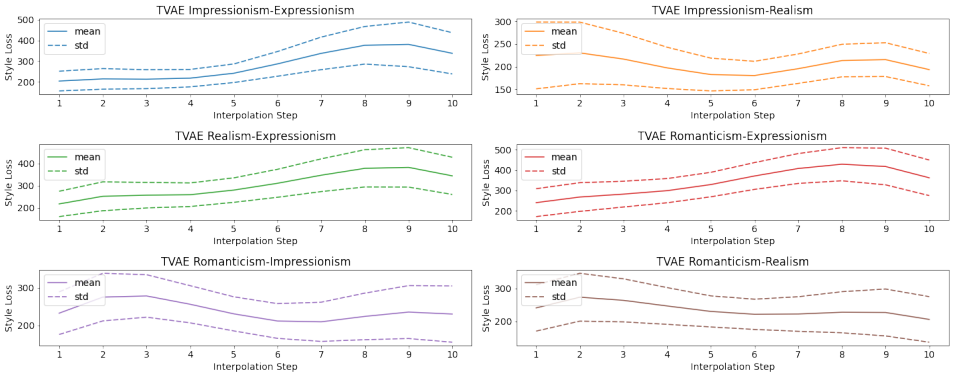


Figure A.9: Inter-class Style Loss Distribution of TVAE.

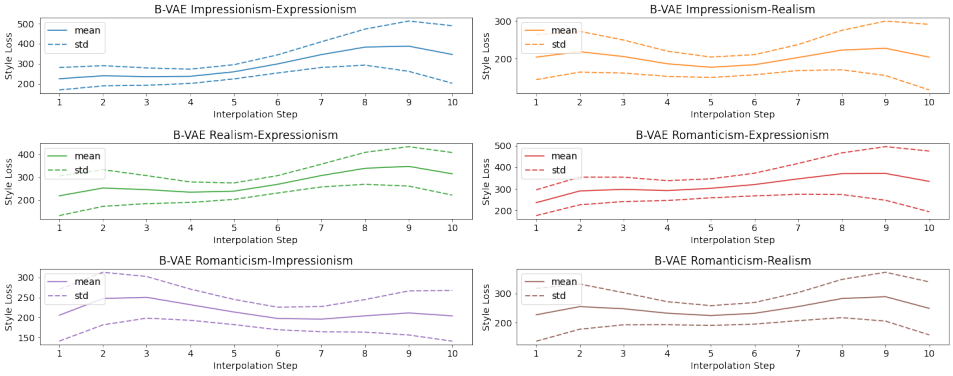


Figure A.10: Inter-class Style Loss Distribution of β-VAE.

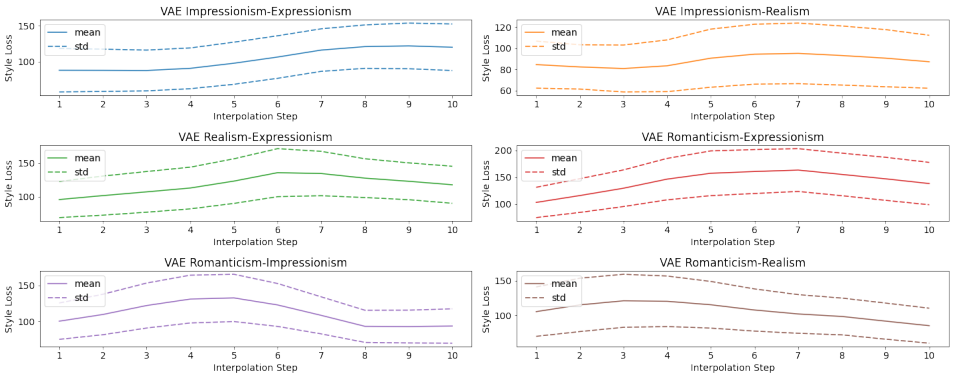


Figure A.11: Inter-class Style Loss Distribution of VAE.

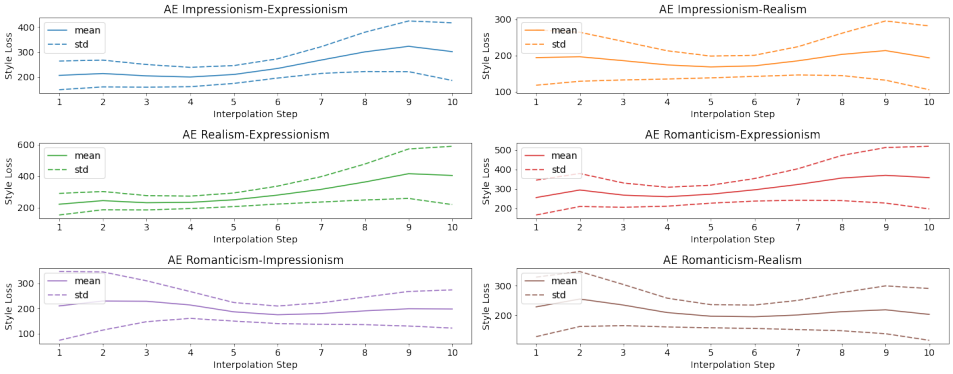


Figure A.12: Inter-class Style Loss Distribution of AE.

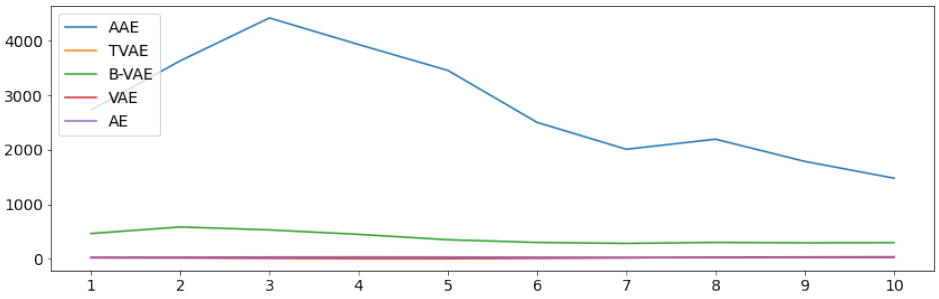


Figure A.13: An Example of Style Loss within Expressionism

Figure A.14: An Example of Interpolation within Expressionism

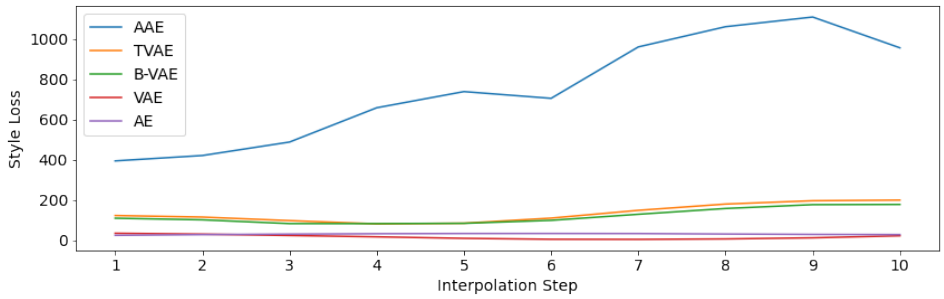


Figure A.15: An Example of Style Loss within Impressionism.

Figure A.16: An Example of Interpolation within Impressionism.

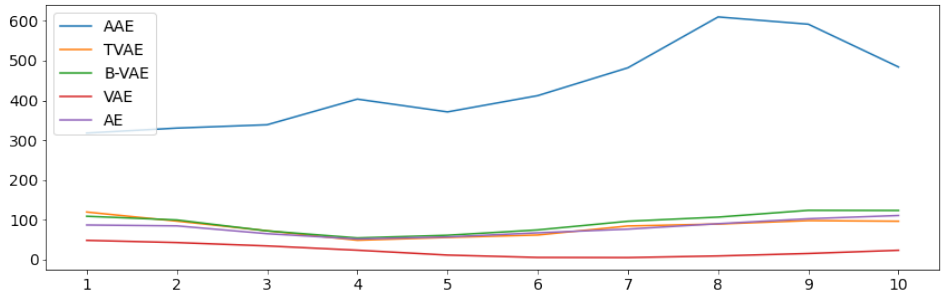


Figure A.17: An Example of Style Loss within Realism.

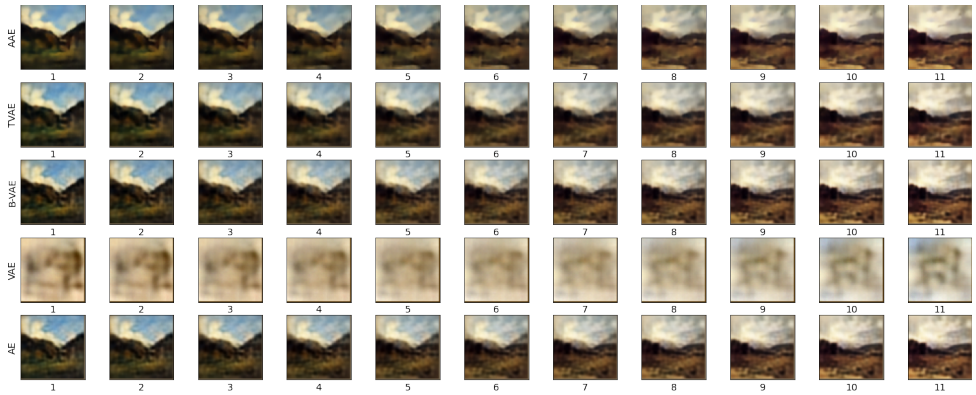


Figure A.18: An Example of Interpolation within Realism.

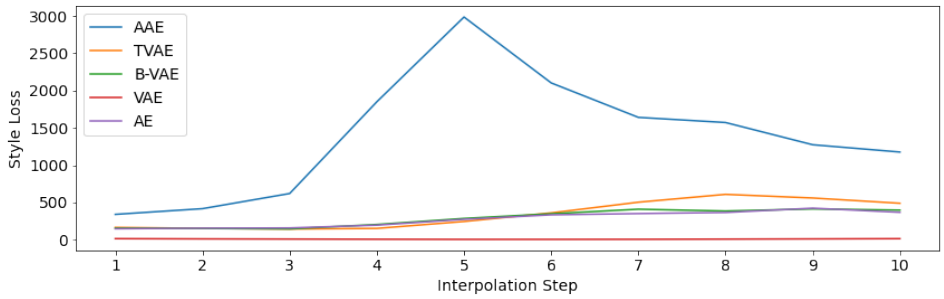


Figure A.19: An Example of Style Loss within Romanticism.

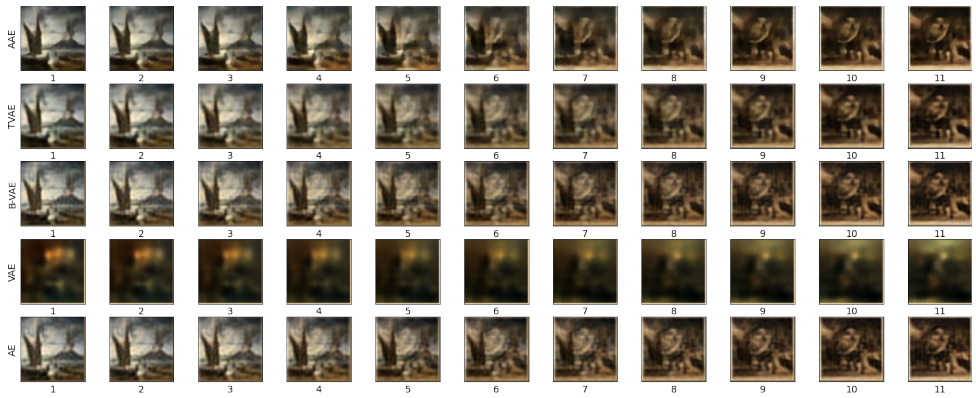


Figure A.20: An Example of Interpolation within Romanticism.

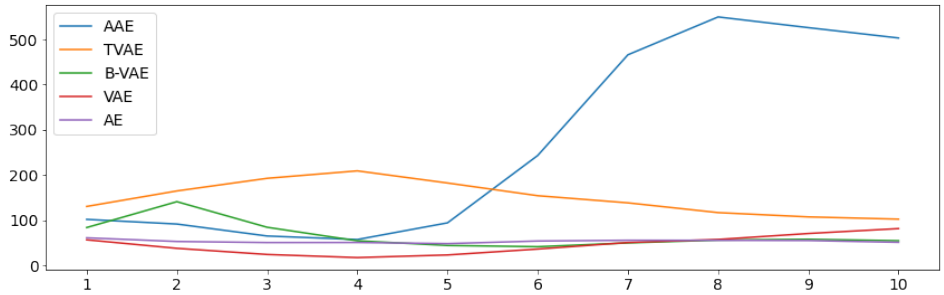


Figure A.21: An Example of Style Loss between Expressionism and Romanticism.

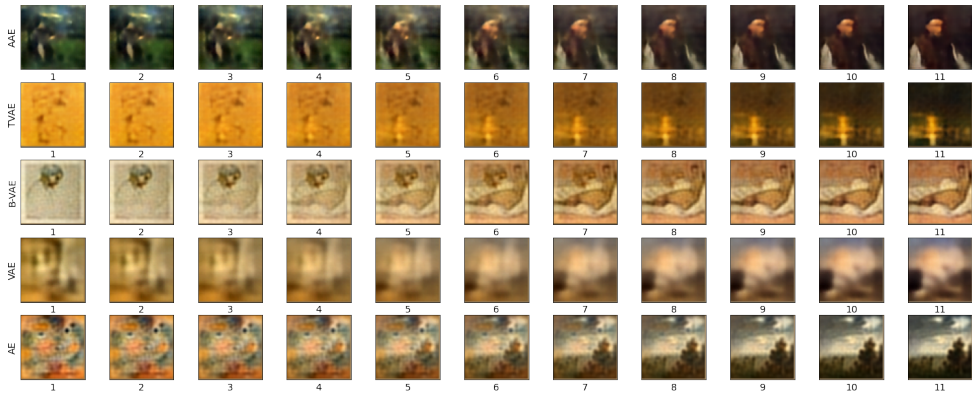


Figure A.22: An Example of Interpolation between Expressionism and Romanticism.

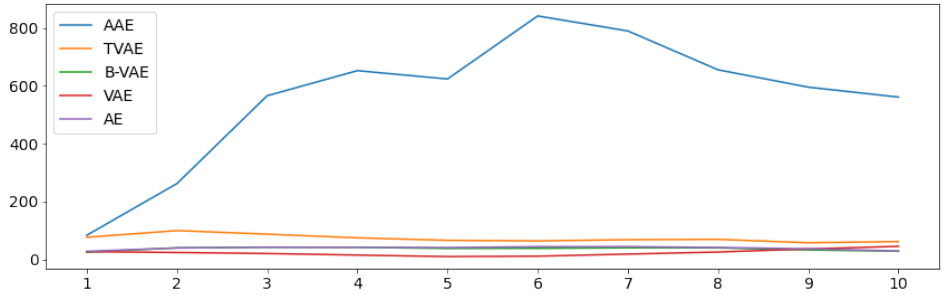


Figure A.23: An Example of Style Loss between Impressionism and Realism.

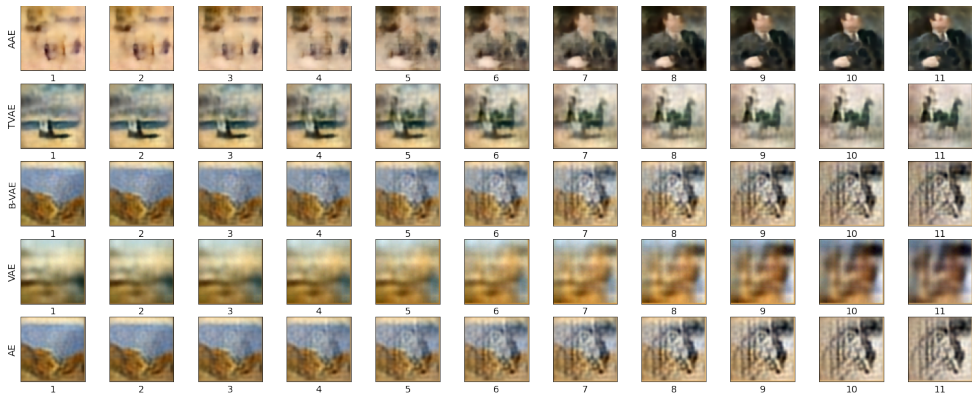


Figure A.24: An Example of Interpolation between Impressionism and Realism.

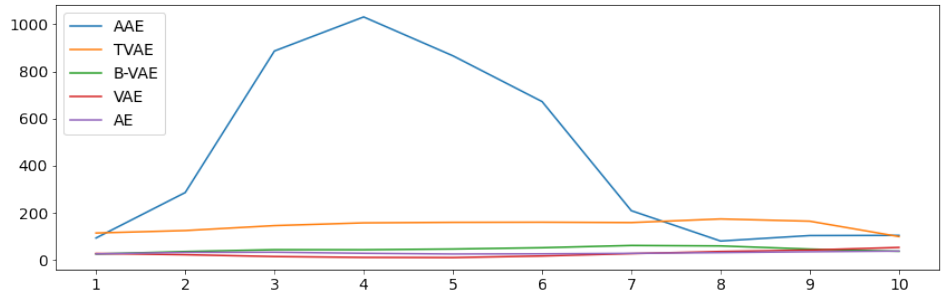


Figure A.25: An Example of Style Loss between Impressionism and Expressionism.

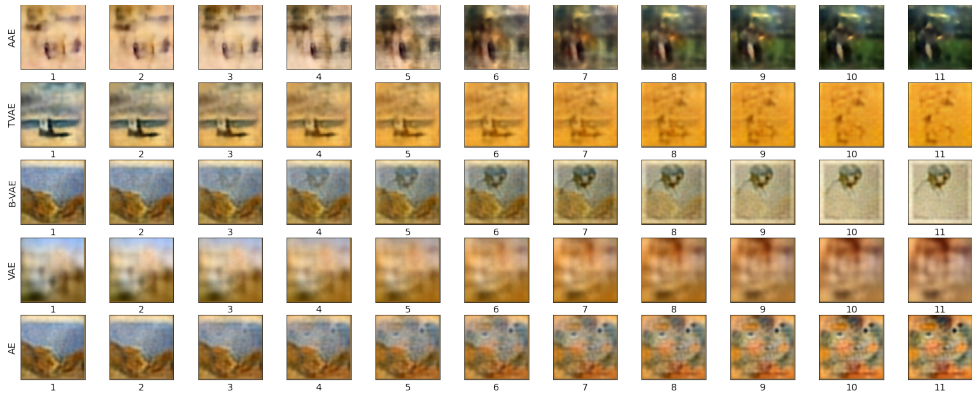


Figure A.26: An Example of Interpolation between Impressionism and Expressionism.

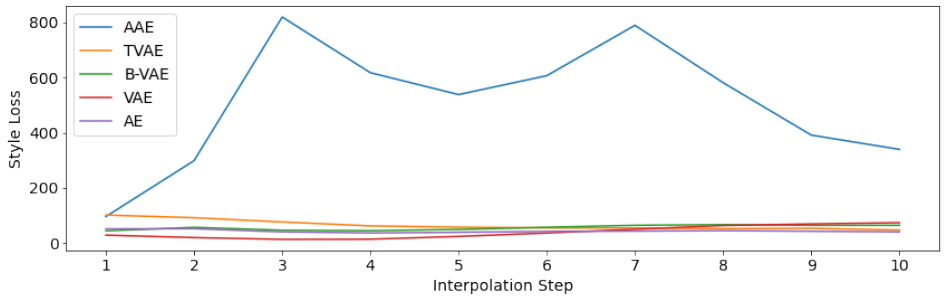


Figure A.27: An Example of Style Loss between Impressionism and Romanticism.

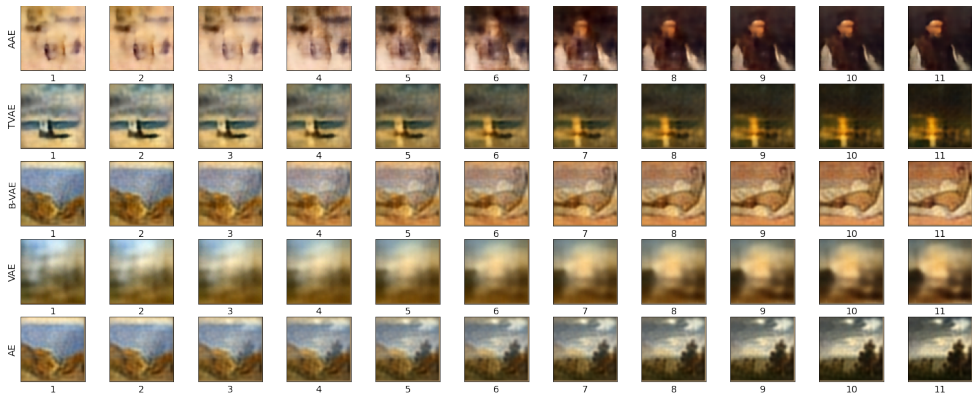


Figure A.28: An Example of Interpolation between Impressionism and Romanticism.

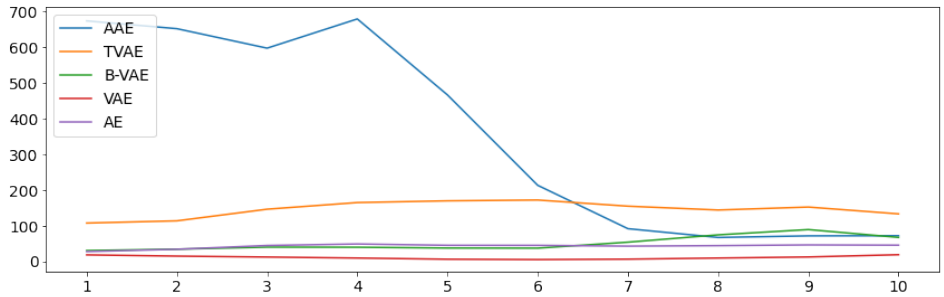


Figure A.29: An Example of Style Loss between Realism and Expressionism.

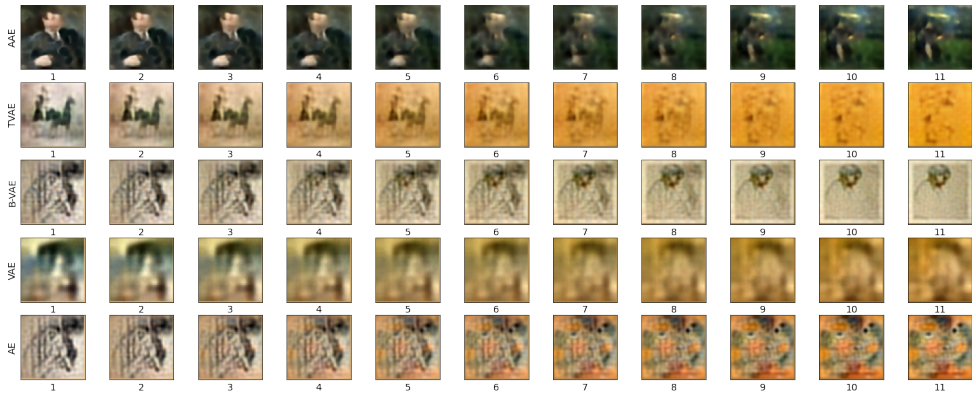


Figure A.30: An Example of Interpolation between Realism and Expressionism.

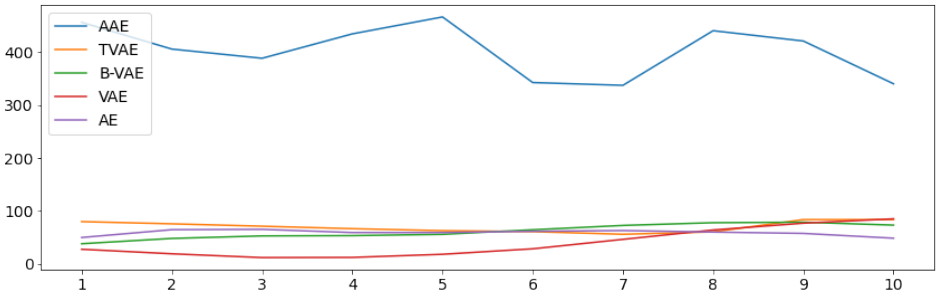


Figure A.31: An Example of Style Loss between Realism and Romanticism.

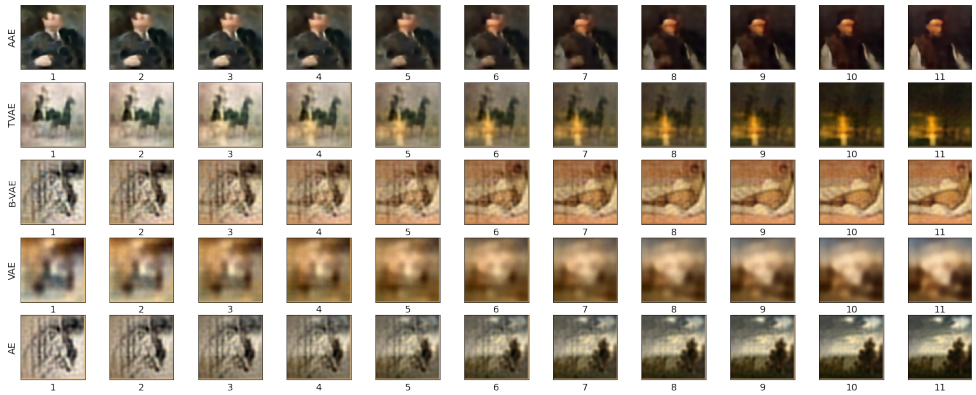


Figure A.32: An Example of Interpolation between Realism and Romanticism.

References

[Almotiri *et al.* 2017] Jasem Almotiri, Khaled Elleithy, and Abdelrahman Elleithy. Comparison of autoencoder and principal component analysis followed by neural network for e-learning using handwritten recognition. In *2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, pages 1–5. IEEE, 2017.

[Baek *et al.* 2002] Kyungim Baek, Bruce A Draper, J Ross Beveridge, and Kai She. Pca vs. ica: A comparison on the feret data set. In *JCIS*, pages 824–827, 2002.

[Bengio *et al.* 2013] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[Berthelot *et al.* 2018] David Berthelot, Colin Raffel, Aurko Roy, and Ian Goodfellow. Understanding and improving interpolation in autoencoders via an adversarial regularizer. *arXiv preprint arXiv:1807.07543*, 2018.

[Bomford and Britain) 1990] D. Bomford and National Gallery (Great Britain). *Impressionism*. Art in the making. National Gallery, 1990.

[Brahma *et al.* 2015] Pratik Prabhanjan Brahma, Dapeng Wu, and Yiyuan She. Why deep learning works: A manifold disentanglement perspective. *IEEE transactions on neural networks and learning systems*, 27(10):1997–2008, 2015.

[Bromley *et al.* 1994] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[Chen *et al.* 2016] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

[Chen *et al.* 2017] Min Chen, Xiaobo Shi, Yin Zhang, Di Wu, and Mohsen Guizani. Deep features learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data*, 2017.

[Creswell *et al.* 2018] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[Davidson *et al.* 2018] Tim R Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M Tomczak. Hyperspherical variational auto-encoders. *arXiv preprint arXiv:1804.00891*, 2018.

[De Ridder and Duin 1997] Dick De Ridder and Robert PW Duin. Sammon's mapping using neural networks: a comparison. *Pattern Recognition Letters*, 18(11-13):1307–1316, 1997.

[Fernie *et al.* 1995] E. Fernie, D.C.I.A.E. Fernie, Phaidon Press, Phaidon Verlag GmbH, G. Vasari, C. van Mander, G.P. Bellori, J.J. Winckelmann, J.W. von Goethe, J. Burckhardt, et al. *Art History and Its Methods*. Phaidon Press, 1995.

[Gatys *et al.* 2015] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[Geng *et al.* 2020] Cong Geng, Jia Wang, Li Chen, Wenbo Bao, Chu Chu, and Zhiyong Gao. Uniform interpolation constrained geodesic learning on data manifold. *arXiv preprint arXiv:2002.04829*, 2020.

[Goodfellow *et al.* 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[Goodfellow *et al.* 2020] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[Goodfellow 2016] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[Hicsonmez *et al.* 2020] Samet Hicsonmez, Nermin Samet, Emre Akbas, and Pinar Duygulu. Ganilla: Generative adversarial networks for image to illustration translation. *Image and Vision Computing*, 95:103886, 2020.

[Higgins *et al.* 2016] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[Hoffer and Ailon 2015] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer, 2015.

[Ishfaq *et al.* 2018] Haque Ishfaq, A Hoogi, and D Rubin. Tvae: Deep metric learning approach for variational autoencoder. In *Proc. ICLR Workshop*, 2018.

[Jin *et al.* 2009] Ruoming Jin, Yuri Breitbart, and Chibuike Muoh. Data discretization unification. *Knowledge and Information Systems*, 19(1):1–29, 2009.

[Kingma and Welling 2014] Diederik P Kingma and Max Welling. Stochastic gradient vb and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*, volume 19, 2014.

[Knyaz *et al.* 2017] Vladimir A Knyaz, Oleg Vygolov, Vladimir V Kniaz, Yury Vizilter, Vladimir Gorbatsevich, Thomas Luhmann, and Niklas Conen. Deep learning of convolutional auto-encoder for image matching and 3d object reconstruction in the infrared range. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2155–2164, 2017.

[Li *et al.* 2021] Ziqiang Li, Rentuo Tao, Hongjing Niu, Mingdao Yue, and Bin Li. Interpreting the latent space of gans via correlation analysis for controllable concept manipulation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 1942–1948. IEEE, 2021.

[Liu *et al.* 2019] Yang Liu, Eunice Jun, Qisheng Li, and Jeffrey Heer. Latent space cartography: Visual analysis of vector space embeddings. In *Computer Graphics Forum*, volume 38, pages 67–78. Wiley Online Library, 2019.

[Lovmar *et al.* 2005] Lovisa Lovmar, Annika Ahlford, Mats Jonsson, and Ann-Christine Syvänen. Silhouette scores for assessment of snp genotype clusters. *BMC genomics*, 6(1):1–6, 2005.

[Maaten and Hinton 2008] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[Makhzani *et al.* 2015] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[Mishra *et al.* 2017] Sidharth Prasad Mishra, Uttam Sarkar, Subhash Taraphder, Sanjay Datta, Devi Prasanna Swain, Reshma Saikhom, Sasmita Panda, and Menalsh Laishram. Multivariate statistical data analysis-principal component analysis (pca). *Int J Liv Res. 2017c*, 7(5):60–78, 2017.

[Nichol 2016] Kiri Nichol. *Painter by Numbers*. =https://www.kaggle.com/c/painter-by-numbers/data, 2016.

[Oring *et al.* 2020] Alon Oring, Zohar Yakhini, and Yacov Hel-Or. Faithful autoencoder interpolation by shaping the latent space. *arXiv preprint arXiv:2008.01487*, 2020.

[Patrini *et al.* 2020] Giorgio Patrini, Rianne van den Berg, Patrick Forre, Marcello Carioni, Samarth Bhargav, Max Welling, Tim Genewein, and Frank Nielsen. Sinkhorn autoencoders. In *Uncertainty in Artificial Intelligence*, pages 733–743. PMLR, 2020.

[Radford *et al.* 2015] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[Rumelhart *et al.* 1985] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. *Learning internal representations by error propagation*. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[Sainburg *et al.* 2018] Tim Sainburg, Marvin Thielk, Brad Theilman, Benjamin Migliori, and Timothy Gentner. Generative adversarial interpolative autoencoding: adversarial training on latent space interpolations encourage convex latent distributions. *arXiv preprint arXiv:1807.06650*, 2018.

[Schroff *et al.* 2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[Shen *et al.* 2020a] Jingyi Shen, Runqi Wang, and Han-Wei Shen. Visual exploration of latent space for traditional chinese music. *Visual Informatics*, 2020.

[Shen *et al.* 2020b] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020.

[Simonyan and Zisserman 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[Subramanian 2020] A.K Subramanian. *PyTorch-VAE*. https://github.com/AntixK/PyTorch-VAE, 2020.

[Szegedy *et al.* 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[Vinay *et al.* 2005] Vishwa Vinay, Ingemar J Cox, Ken Wood, and Natasa Milic-Frayling. A comparison of dimensionality reduction techniques for text retrieval. In *Fourth International Conference on Machine Learning and Applications (ICMLA'05)*, pages 6–pp. IEEE, 2005.

[Wang *et al.* 2014] Wei Wang, Yan Huang, Yizhou Wang, and Liang Wang. Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 490–497, 2014.

[Weinberger and Saul 2009] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[Zeiler and Fergus 2014] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.