

https://dspace.org/

Disentanglement using Vaes resembles distance learning and requires overlapping data

Michlo, Nathan Juraj

2022
https://hdl.handle.net/10539/37003

Downloaded from a DSpace repository. https://dspace.org/

Disentanglement Using VAEs Resembles Distance Learning and Requires Overlapping Data

A Computer Science Master of Science Dissertation

School of Computer Science & Applied Mathematics
University of the Witwatersrand

Nathan Juraj Michlo
1386161

Supervised by
Dr. Richard Klein
Dr. Steven James

August 18, 2022



A dissertation submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, South Africa, in fulfilment of the requirements for the degree of Master of Science

Abstract

Learning disentangled representations with variational autoencoders (VAEs) is often attributed to the regularisation component of the loss. In this work, we highlight the interaction between data and the reconstruction term of the loss as the main contributor to disentanglement in VAEs. We note that standardised benchmark datasets are constructed in a way that is conducive to learning what appear to be disentangled representations. We design an intuitive adversarial dataset that exploits this mechanism to break existing state-of-the-art disentanglement frameworks. We provide solutions in the form of a modified reconstruction loss suggesting that VAEs are distance learners, we also show that these loss functions can be learnt. From this idea, we introduce new scores that measure if disentangled representations using distances have been discovered. We then solve these scores by introducing a supervised metric learning framework that encourages disentanglement. Finally, we present various considerations for disentanglement research based on the subjective nature of disentanglement itself and the results from our work which suggest that VAE disentanglement is largely accidental.

Declaration

I, Nathan Juraj Michlo, hereby declare the contents of this research dissertation to be my own work. This dissertation 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.

Signature:  Date: 2022/08/19

Acknowledgements

This work was supported in part by the National Research Foundation of South Africa (Grant Numbers: 118075 and 117808).

Computations were performed using High Performance Computing infrastructure provided by the Mathematical Sciences Support unit at the University of the Witwatersrand.

Contents

Preface i
Abstract . . . ii
Declaration . . . iii
Acknowledgements . . . iv
Table of Contents . . . v

1 Introduction 1

2 Background 4
2.1 Variational Auto-Encoder . . . 4
2.1.1 VAE Formulation . . . 6
2.1.2 VAE Objective Function . . . 6
2.1.3 VAE in Practice . . . 7
2.1.4 Optimising VAEs with the AEVB Algorithm . . . 8
2.2 Disentanglement . . . 9
2.2.1 Disentanglement Datasets . . . 10
2.2.2 Disentanglement Scores . . . 11
2.3 Disentanglement Frameworks . . . 12
2.3.1 β-VAE . . . 13
2.3.2 Non-identifiability of Unsupervised Models . . . 13
2.3.3 Adaptive VAEs . . . 14
2.4 Related Work . . . 16

3 VAEs Learn Distances Over Data 18
3.1 Random Sampling Reorganises VAE Embeddings . . . 18
3.2 Existing Datasets . . . 20
3.2.1 Ground-Truth Distance . . . 20
3.2.2 Perceived Distance . . . 22
3.2.3 Perceived Overlap . . . 22
3.2.4 Perceived Distances Correspond To Ground-Truth Distances In Benchmark Datasets . . . 23
3.3 Adversarial Dataset . . . 25
3.3.1 XYSquares Dataset . . . 25
3.3.2 Experimental Setup . . . 26
3.3.3 Adversarial Experiments . . . 27
3.3.4 Varying Levels of Overlap . . . 28
3.4 Introducing Overlap . . . 30

3.4.1 Augmented Loss Experiments . . . 30
3.5 Identifying Factor Importance . . . 32
3.6 Summary . . . 33

4 Disentanglement Metrics **35**
4.1 Ground-Truth Factored Component Scores . . . 35
4.1.1 Ground-Truth Distance Correlation . . . 36
4.1.2 Axis-Alignment Ratio . . . 37
4.1.3 Linearity Ratio . . . 38
4.1.4 Evaluation . . . 40
4.2 Perceived Distances Over Data Correlate To Ground-Truth . . . 40
4.3 VAE Frameworks Learn Perceived Distances . . . 43
4.4 Summary . . . 45

5 Metric Learning **46**
5.1 Metric Learning Overview . . . 46
5.1.1 Triplet Loss . . . 46
5.1.2 Soft-Margin Triplet Loss . . . 47
5.2 Sampling Strategies . . . 47
5.2.1 Sampling Based on Categories . . . 48
5.2.2 Sampling Based on Number of Differing Factors . . . 49
5.2.3 Sampling Based on Ground-Truth Distance Along Factors . . . 49
5.2.4 Sampling Based on Normalised Ground-Truth Distance . . . 50
5.3 Triplet Auto-Encoders . . . 50
5.3.1 β-TVAE . . . 50
5.3.2 ℓ_1 versus ℓ_2 Distance . . . 51
5.3.3 Experimental Results . . . 51
5.4 Axis-Aligned Metric Learning . . . 54
5.4.1 Problems Estimating Shared Variables . . . 54
5.4.2 Summary Of Adaptive Methods . . . 56
5.4.3 Adaptive Triplet Loss . . . 56
5.4.4 Ada-TVAE . . . 58
5.4.5 Adaptive Weight Schedule . . . 58
5.4.6 Experimental Results . . . 59
5.5 Unsupervised Adaptive Triplet . . . 60
5.5.1 Sampling Based on Perceived Distance . . . 61
5.5.2 Results . . . 62
5.6 Summary . . . 63

6 Learning Overlap **65**
6.1 Learning To Disentangle . . . 65
6.1.1 Learning Loss Functions . . . 65
6.1.2 Augmented Loss Function . . . 66
6.1.3 Experimental Results . . . 67
6.2 Summary . . . 70

7 Considerations for Disentanglement Research **71**

7.1 Problems With Disentanglement . . . 71
7.2 Choice Of Ground-Truth Factors . . . 73
7.3 Random External Factors . . . 74
7.4 Summary . . . 76

8 Conclusion & Future Work **77**
8.1 Future Work . . . 77
8.2 Conclusion . . . 78

References **80**

A Supplementary Details **84**
A.1 Normalised Axis-Orientation Ratio . . . 84
A.2 VAE Implementation Details . . . 84
A.2.1 Beta Normalisation . . . 84
A.2.2 Symmetric KL . . . 85
A.2.3 Sampling Ada-GVAE Pairs . . . 85
A.3 VAE Experiment Details . . . 85
A.3.1 Optimiser and Batch Size . . . 85
A.3.2 Model Architecture . . . 85
A.3.3 Dataset Standardisation . . . 86
A.3.4 Experiment Sweeps . . . 86

Nomenclature

Notation used in this list may be overridden depending on context, however, if that is the case, explicit mention of variable and notation meanings will be given.

Variables	
a	Scalar
$\boldsymbol{a}$	Vector
A	Set or Matrix
a_j	The j_{th} scalar of vector $\boldsymbol{a}$
$\boldsymbol{a}^{(i)}$	The i_{th} vector of the ordered set A
$a_k^{(j)}$	The k_{th} scalar element of the j_{th} vector of the ordered set A. To minimise confusion, although still valid, we avoid the notation $a_{j,k} = a_k^{(j)}$
$\ddot{\mathcal{A}}$	An ordered set of pairs of elements sampled from $\ddot{\mathcal{A}} \subset \mathcal{A} \times \mathcal{A}$
General Notation	
$\|\boldsymbol{x}\|_p$	Is the generalised ℓ_p-norm, where $\ell_p(\boldsymbol{x}) = \|\boldsymbol{x}\|_p$. $\ell_1(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ is the Manhattan norm, $\ell_2(\boldsymbol{x}) = \|\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|$ is the Euclidean norm
$B \setminus A$	The set difference of B and A, also ambiguously written as $B - A$
Overloaded Notation Depending On Context	
$\|a\|$	Absolute value of variable a
$\|A\|$	Cardinality of set A, the number of elements in set A
$[n]$	Bracket notation for the set of natural numbers $\{1, ..., n\}$
$[\![P]\!]$	Iverson bracket returning 1 if P is true, otherwise 0
Inputs, Representations, Reconstructions	
$\mathcal{X}$	Ordered dataset $\mathcal{X} \subset \mathbb{R}^{\text{N}}$ consisting of observation vectors
$\mathcal{Z}$	Set of all representations or embedding vectors $\mathcal{Z} \subset \mathbb{R}^{\text{D}}$
$\boldsymbol{x}$	Observation or input vector $\boldsymbol{x} \in \mathcal{X}$
$\boldsymbol{z}$	Encoding or latent representation vector $\boldsymbol{z} \in \mathcal{Z}$
$\hat{\boldsymbol{x}}$	Decoding or reconstruction vector of a representation $\hat{\boldsymbol{x}} \in \mathbb{R}^{\text{N}}$
N	Dimensionality of observation vectors
D	Dimensionality of representation vectors
$\boldsymbol{x}^{(j)}$	The j_{th} observation from the ordered dataset $\boldsymbol{x}^{(j)} \in \mathcal{X} = \left\{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(n)}\right\}$
$\boldsymbol{z}^{(j)}$	The j_{th} representation corresponding to $\boldsymbol{x}^{(j)}$
$\hat{\boldsymbol{x}}^{(j)}$	The j_{th} reconstruction corresponding to $\boldsymbol{x}^{(j)}$ and $\boldsymbol{z}^{(j)}$
Auto-Encoder	
$f_{\boldsymbol{\phi}}(\boldsymbol{x})$	The deterministic encoder $f_{\boldsymbol{\phi}} : \mathcal{X} \mapsto \mathcal{Z}$ or rather $\boldsymbol{z} = f_{\boldsymbol{\phi}}(\boldsymbol{x})$
$g_{\boldsymbol{\theta}}(\boldsymbol{z})$	The deterministic decoder $g_{\boldsymbol{\theta}} : \mathcal{Z} \mapsto \hat{\mathcal{X}}$ or rather $\hat{\boldsymbol{x}} = g_{\boldsymbol{\theta}}(\boldsymbol{z})$

Variational Auto-Encoder	
$p_{\boldsymbol{\theta}}(\boldsymbol{z})$	Prior distribution
$q_{\boldsymbol{\phi}}(\boldsymbol{z}\|\boldsymbol{x})$	Probabilistic encoder, inference model, or approximate posterior
$p_{\boldsymbol{\theta}}(\boldsymbol{x}\|\boldsymbol{z})$	Probabilistic decoder, generative model, or conditional distribution
Variational Auto-Encoder - Factorised Gaussian Encoder	
$\mathcal{N}(\mathbf{0}, \boldsymbol{I})$	Prior distribution is the multivariate normal distribution with diagonal covariance $p_{\boldsymbol{\theta}}(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$	Probabilistic encoder is the multivariate normal distribution with diagonal covariance $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}\|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$, parameterised by the deterministic encoder $f_{\boldsymbol{\phi}} : \mathcal{X} \mapsto \mathbb{R}^{\mathrm{D}} \times \mathbb{R}^{\mathrm{D}}$ which outputs mean and standard deviation parameters $\boldsymbol{\mu}, \boldsymbol{\sigma} = f_{\boldsymbol{\phi}}(\boldsymbol{x})$
$\boldsymbol{\mu}^{(j)}$	The j_{th} mean parameter vector corresponding to $\boldsymbol{x}^{(j)}$
$\boldsymbol{\sigma}^{(j)}$	The j_{th} standard deviation parameter vector corresponding to $\boldsymbol{x}^{(j)}$
Ground-Truth Factors And Datasets	
F	The number of individual ground-truth factors that generate a dataset and thus also the dimensionality of each vector of factors $\boldsymbol{y}$
f_i	The size of an individual factor $i \in [\mathrm{F}]$ or rather the number of allowed values for individual factors, where usually $f_i \geq 2$
$\mathcal{Y}$	The set of all ground-truth factor vectors $\mathcal{Y} = [f_1] \times ... \times [f_{\mathrm{F}}]$
$\boldsymbol{y}$	A single ground-truth factor vector $\boldsymbol{y} \in \mathcal{Y}$ of dimensionality F, made up of individual factors $y_i \in [f_i]\ \forall i \in [\mathrm{F}]$
$\boldsymbol{y}^{(j)}$	The j_{th} vector of ground-truth factors used to generate $\boldsymbol{x}^{(j)}$
$\mathcal{Y}^{(a,i)}$	Ordered traversal over factor $i \in [\mathrm{F}]$, through the point $\boldsymbol{y}^{(a)} \in \mathcal{Y}$, with a traversal size of $\|\mathcal{Y}^{(a,i)}\| = f_i$
Perceived Overlap	
$\mathrm{v}_{\mathrm{pcv}}$	Measure of perceived overlap between observations $\mathrm{v}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \leq 0$
$\mathrm{d}_{\mathrm{pcv}}$	Distance function between two observations $\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \geq 0$. Perceived distance is the inverse of perceived overlap $\mathrm{d}_{\mathrm{pcv}} = -\mathrm{v}_{\mathrm{pcv}}$
Adaptive Methods	
k	The number of ground-truth factors *differing* between two observations
$\mathrm{D} - k$	The number of ground-truth factors *shared* between two observations
$\bar{S}$	The set of *differing* factor indices $\bar{S} = [\mathrm{D}] \setminus S$ between two observations
S	The set of *shared* factor indices $S \subseteq [\mathrm{D}]$ between two observations
a	A function that produces an average of two posterior distributions
δ_i	The distance between corresponding latent units $i \in [\mathrm{D}]$
τ	The *averaging threshold* controls which elements are considered shared
Triplet Loss	
a, p, n	Indices of anchor, positive and negative items in a triplet
d	Distance function within triplet loss $\mathrm{d}(a, b)$ or $\mathrm{d}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)})$
s	Distance function used to sample triplets: $\mathrm{s}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \leq \mathrm{s}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(n)})$
ψ	Triplet distances greater than the *hard-margin* have no effect
Adaptive Triplet Loss	
η	The *averaging strength* hyper-parameter adjusts the threshold τ
ω	The *shared weight*, adaptive triplet scales distances using this value

Chapter 1

Introduction

A fundamental challenge in machine learning is to discover useful representations from high-dimensional data, which can then be used to solve subsequent tasks effectively. Recently, deep learning approaches have showcased the ability of neural networks to extract meaningful features from high-dimensional inputs for tasks ranging from classification [Krizhevsky *et al.* 2012] to reinforcement learning [Mnih *et al.* 2015]. While these learnt representations have obviated the need for manual feature selection or preprocessing pipelines, they are often not semantically meaningful, which can negatively impact interpretability, fairness [Locatello *et al.* 2019a] and downstream task performance [Locatello *et al.* 2019b].

Prior work has therefore argued that it is desirable to learn a representation that is *disentangled* [Bengio *et al.* 2013]. While there is no consensus on what constitutes a disentangled representation, it is generally thought that such a representation should be factorised so that each latent variable corresponds to a single explanatory variable responsible for generating the data [Burgess *et al.* 2017]. For example, a single image from a video game may be represented by latent variables governing the x and y positions of the player, enemies and collectable items.

A common approach to discovering these seemingly disentangled representations are Variational Auto-Encoders (VAEs) [Kingma and Welling 2014], which are trained on unlabelled data to learn a lower-dimensional representation capable of reconstructing the given input. Unfortunately, it has been proven that unsupervised methods cannot reliably learn representations without the introduction of supervision or inductive biases [Locatello *et al.* 2019b]. Fortunately, the recently introduced Ada-GVAE framework has partially overcome this problem by using a weakly supervised signal to discover these underlying factors [Locatello *et al.* 2020], but there still remains room for improvement.

Interestingly, VAEs do not have an explicit mechanism that encourages the learning of disentangled representations. However, this behaviour is usually attributed to the regularisation term and the information bottleneck principle, where the decoder minimises errors in the reconstruction process due to random sampling [Burgess *et al.* 2017; Mathieu *et al.* 2019; Rolinek *et al.* 2019]. Despite this hypothesis, there is still no explicit reason for why the representations learnt by these frameworks should align with generative factors in the data. Nonetheless, these frameworks have been shown to produce disentangled representations (as measured by appropriate metrics [Eastwood and Williams 2018; Chen *et al.* 2018; Zaidi *et al.* 2020]) when trained on synthetically generated data such as 3D Shapes [Burgess and Kim 2018].

In this work, we seek to understand why VAEs implicitly learn these disentangled representations by investigating the interaction between the reconstruction loss of the VAE and the input data. We find compelling evidence that disentanglement occurs not because of special algorithmic choices or the regularisation term, but because of the overlap between observations in the datasets themselves. In particular, we find that *standardised benchmarks [LeCun et al. 2004; Reed et al. 2015; Matthey et al. 2017; Burgess and Kim 2018; Gondal et al. 2019] are constructed in such a way that they accidentally encourage the models to learn what appear to be disentangled representations.*

The main contributions of this work are summarised as follows:

1. We introduce the idea of *perceived overlap* and *perceived distance* between pairs of data points, measured in terms of the reconstruction loss of VAE frameworks. We show that perceived distances in existing datasets naturally correspond to the distances between ground-truth factors of these datasets, and that VAEs learn these distances. This explains why the learnt representations can appear disentangled.
2. We demonstrate the ineffectiveness of state-of-the-art models by designing a simple adversarial dataset with constant perceived distance between elements, over which VAE-based frameworks fail to learn disentangled representations.
3. We give an example of using a hand-crafted reconstruction loss to disentangle the adversarial dataset. This loss changes the VAE's perceived overlap across the dataset so that the framework captures ground-truth factors, which suggests that VAEs are more akin to distance learners.
4. We introduce three new disentanglement scores for evaluating learnt representations. Each score captures a different property of factored representations that can arise from learning the distances between ground-truth factors in the data.
5. We use supervision to rectify the distances learnt within VAEs. We propose a metric learning framework that is modified to encourage disentanglement by estimating which ground-truth factors have changed between pairs of observations. Our framework improves upon existing unsupervised and weakly-supervised VAEs.
6. We introduce a supervised framework for learning reconstruction losses that improve disentanglement, as hand-crafting the loss may be unintuitive.
7. We highlight various issues that relate to the subjective nature of disentanglement itself. Even if the data is kept constant, the choice of ground-truth factors themselves can affect measurement of disentanglement. Furthermore, the desired ground-truth factors learnt by a model can be masked by real-world noise within the data. We introduce two additional datasets that accentuate these issues.
8. We contribute a modular and general-purpose disentanglement framework built with PyTorch [Paszke *et al.* 2017] called *Disent*,[1] implementing common models, metrics and datasets. We use the Disent framework to implement our own contributions and experiments.[2]

[1]The Disent framework is available at `https://github.com/nmichlo/disent`.
[2]Research code and implementations are available at `https://github.com/nmichlo/msc-research`

Having outlined our main contributions, we give a brief overview of the structure of this work. We begin with the necessary background for VAEs and disentanglement in Chapter 2 as well as a related work. Chapter 3 forms the foundation of our work and investigates the mechanism in which VAEs learn distances over data and how this can lead to disentanglement. Chapter 4 formalises this concept by introducing three new disentanglement scores that measure different attributes of disentangled representations learnt from distances. Chapter 5 proposes a novel disentangled metric learning framework that uses supervision to rectify the distances learnt within VAEs and improve disentanglement. Chapter 6 investigates an alternative approach for improving disentanglement where we instead learn new reconstruction loss functions for use with VAEs. Based on all previous chapters, Chapter 7 highlights various considerations for researchers and pitfalls with regard to disentanglement. Finally, in Chapter 8, we suggest future avenues for research and conclude our work.

Chapter 2

Background

This chapter introduces foundational ideas in the areas of *representation learning* and *disentanglement*. We begin by outlining the *Variational Auto-Encoder* (VAE) in Section 2.1, upon which all other works in this chapter extend. The VAE is an unsupervised generative model that learns compressed representations over the data.

We next describe *Disentanglement*, which is the idea of improving representation learning methods such that they are human-interpretable and intuitive. In Section 2.2 we give an overview of disentanglement as well as datasets and metrics used for benchmarking learnt representations.

Various approaches have been proposed for improving disentanglement. The most notable is the unsupervised β-VAE in Section 2.3.1 which is a simple modification to the original VAE that improves regularisation. However, a recent proof described in Section 2.3.2 shows that because both the VAE and β-VAE are unsupervised methods, they are incapable of effectively learning disentangled representations, relying far more on supervised model selection. This *non-identifiability* result, however, can be bypassed using some form of inductive bias or weak-supervision. Recent semi-supervised approaches to disentanglement called *Adaptive VAEs* are outlined in Section 2.3.3. These methods estimate the number of factors that have changed between pairs of observations to enforce partially shared representations and improving disentanglement. The new disentanglement approaches we introduce in this work take inspiration from this estimation procedure.

2.1 Variational Auto-Encoder

Auto-Encoders consist of two coupled, yet separately parametrised models, an *encoder* and a *decoder*. The encoder takes an observation as input and outputs a latent representation, while the decoder accepts this latent representation as an input and attempts to reconstruct the observation. Usually, the dimensionality of the representation is much less than that of the input and output. This reduction in dimensionality causes the model to compress the dataset it is trained on; only the useful information needed to reconstruct the observations is encoded. We highlight this architecture in Figure 2.1.

Variational Auto-Encoders (*VAEs*), while architecturally similar to Auto-Encoders, are instead generative models, with their underlying formulations differing significantly. The *probabilistic encoder* in VAEs is also known as the *recognition* or *inference* model,

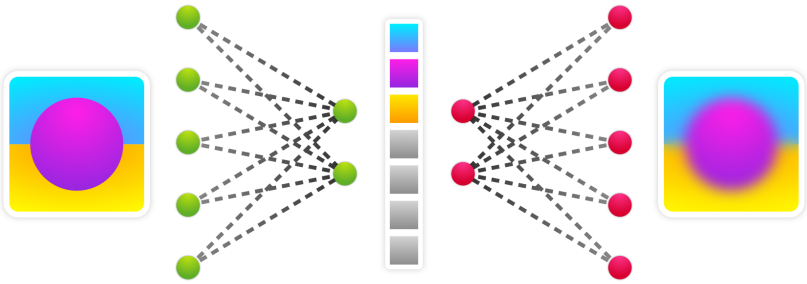


Figure 2.1: Architectural diagram of typical Auto-Encoder (AE). The input to an AE is a single observation and the output of the model should be a reconstruction of the input. The fundamental idea is that the dimensionality at the middle of the model is much less than the input or the output, causing the model to compress the data. Within an AE, these compression (encoding) and decompression (decoding) procedures are deterministic. Ideally, the model should learn representations that are useful. For example, as visualised, the input observation has floor, object and wall colours which are captured by the representation.

while the *probabilistic decoder* is known as the *generative* model [Kingma and Welling 2019]. The fundamental difference is that the encoder parameterises distributions over the representations that are then sampled from during training. We highlight this architectural difference between VAEs and standard AEs in the examples from Figure 2.2 and Figure 2.1.

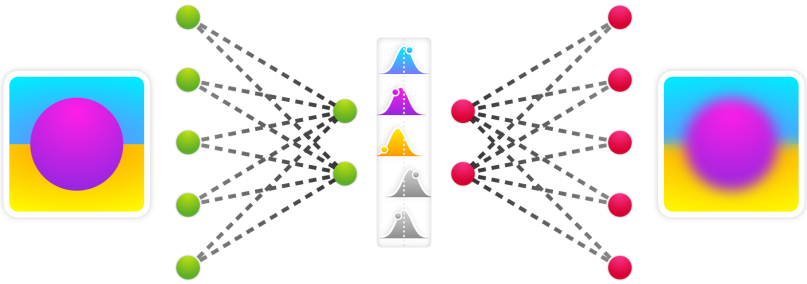


Figure 2.2: Architectural diagram of typical Variational Auto-Encoder (VAE). Like a standard Auto-Encoder (AE) in Figure 2.1, the output of the VAE should be a reconstruction of the input. Similarly, the dimensionality of the representation or embedding is usually much less than the input and output, causing the model to compress the dataset. The difference, however, is that the encoder is probabilistic, outputting distributions over latent variables. During training, samples are taken from these distributions, resulting in a non-deterministic training procedure. Ideally, the model should learn distributions that are useful. For example, as visualised, the input observation has floor, object and wall colours which are captured by these distributions.

2.1.1 VAE Formulation

The following VAE formulation is adapted from Kingma and Welling [2014 2019] and Locatello *et al.* [2020] to standardise notation throughout the chapter.

Assume $\mathcal{X} = \left\{\boldsymbol{x}^{(0)}, ..., \boldsymbol{x}^{(n)}\right\}$ is a set of continuous or discrete, independent and identically distributed (i.i.d) observations $\boldsymbol{x} \in \mathbb{R}^{\mathrm{N}}$, generated by some random process involving an unobserved random variable $\boldsymbol{z} \in \mathbb{R}^{\mathrm{D}}$ of lower dimensionality. Additionally, the true *prior distribution* $\boldsymbol{z} \sim p_*(\boldsymbol{z})$ and true *conditional distribution* $\boldsymbol{x} \sim p_*(\boldsymbol{x}|\boldsymbol{z})$ are unknown.

The goal of the generative model is to approximate the true generative process. We define this approximation as the *probabilistic decoder* $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})$, which is parametrised by $\boldsymbol{\theta}$. $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{z})$ are the differentiable conditional and prior distributions, respectively. After training (see Section 2.1.4) we can sample from this prior distribution $\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\boldsymbol{z})$ to generate novel reconstructions $\hat{\boldsymbol{x}}$ (see Section 2.1.3).

Given the generator, we use Bayes rule to compute the *posterior distribution* $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z}) \,/\, p_{\boldsymbol{\theta}}(\boldsymbol{x})$ which describes the underlying latent factors of the data. However, with any moderately complex generator function, it is highly likely that the *marginal likelihood* $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})d\boldsymbol{z}$ will be intractable. Thus, by definition, the posterior will also be intractable, resulting in its inability to be evaluated or differentiated. To work around this problem, we define the *probabilistic encoder* $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, with weights $\boldsymbol{\phi}$, as an approximation to the intractable true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$.

With these definitions, we can now describe the learning process for model parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ in Sections 2.1.2 to 2.1.4.

2.1.2 VAE Objective Function

The optimisation objective of the VAE framework is maximising the *evidence lower bound* (ELBO) or the *variational lower bound*. This is done by minimising the VAE loss given by Equation 2.1. Detailed derivations and explanations are given by [Kingma and Welling 2014]. For notational simplicity, we exclude the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ of the model when writing the VAE loss.

$$\mathcal{L}_{\mathrm{VAE}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \mathcal{L}_{\mathrm{rec}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \mathcal{L}_{\mathrm{reg}}(\boldsymbol{x}) \tag{2.1}$$

The VAE loss is comprised of two components, the reconstruction term $\mathcal{L}_{\mathrm{rec}}$ and the regularisation term $\mathcal{L}_{\mathrm{reg}}$, given by Equations 2.2 and 2.3 respectively. Minimising these terms has simultaneous effects. The regularisation term encourages the distributions over the representations learnt by the encoder to match the prior distribution, while the reconstruction term improves the outputs from the decoder. These terms usually contradict in practice, with strong regularisation leading to worse reconstructions but embeddings that are often more human interpretable or disentangled (see Section 2.2) [Higgins *et al.* 2017; Burgess *et al.* 2017].

$$\mathcal{L}_{\text{reg}}(\boldsymbol{x}) = -D_{\text{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{z})\right) \tag{2.2}$$

$$\mathcal{L}_{\text{rec}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right] \tag{2.3}$$

Derivative VAE-based approaches often make slight modifications to the loss [Higgins *et al.* 2017; Zhao *et al.* 2017; Hou *et al.* 2017; Kumar *et al.* 2018; Chen *et al.* 2018; Kim and Mnih 2018; Locatello *et al.* 2020], but the terms of these modified loss functions can usually still be grouped into the same regularisation and reconstruction components.

2.1.3 VAE in Practice

For simplicity, one of the most common choices in VAE literature is a *factorised Gaussian encoder* where the posterior is modelled using a multivariate Gaussian distribution with diagonal covariance [Kingma and Welling 2019] and parametrised by a neural network. Similarly, the prior over latent units is chosen to be the standard multivariate normal distribution.

Formally, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the mean vector and diagonal covariance vector parametrising this multivariate normal distribution. This multivariate distribution can be further decomposed into the product of normal distributions parametrised by μ_i and σ_i as seen in Equation 2.4. Additionally, the prior over latent variables is the multivariate normal distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathbf{I})$, with a mean of $\mathbf{0}$ and diagonal covariance $\mathbf{I}$.

$$\begin{aligned} q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2)) \\ &= \prod_i \mathcal{N}(z_i; \mu_i, \sigma_i^2) \\ &= \prod_i q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}) \end{aligned} \tag{2.4}$$

The current formulation of the VAE objective cannot be differentiated with respect to the encoder parameters ϕ. To enable model training through backpropagation, we use the *reparametrisation trick*. We express the random variable $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ in terms of an alternative random variable $\boldsymbol{\epsilon}$, that is independent of $\boldsymbol{x}$ and $\boldsymbol{\phi}$ [Kingma and Welling 2019]. We use this new random variable to reparameterise $\boldsymbol{z}$ using a differentiable and invertible function $\boldsymbol{z} = g(\boldsymbol{\epsilon}, \boldsymbol{\phi}, \boldsymbol{x})$ [Kingma and Welling 2014]. We can now rewrite the VAE loss from Section 2.1.2 using the parameterisation $\mathcal{L}_{\text{VAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}, \boldsymbol{\epsilon})$.

Following the reparametrisation trick, we implement sampling from the multivariate normal posterior parameterised by the encoder in Equation 2.5. [1] A visual example of the encode-sample-decode pipeline of a VAE is given in Figure 2.3. In practice, we

[1] The operator $\odot$ is the elementwise-product between vectors.

directly output $\log \boldsymbol{\sigma}^2$ from the encoder rather than $\boldsymbol{\sigma}^2$ so that the values of $\boldsymbol{\sigma}^2$ are never negative or zero.

$$\begin{aligned} \boldsymbol{\epsilon} &\sim \mathcal{N}(0, \mathbf{I}) \\ (\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2) &= \text{EncoderNeuralNet}_{\boldsymbol{\phi}}(\boldsymbol{x}) \\ \boldsymbol{z} &= \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon} \end{aligned} \tag{2.5}$$

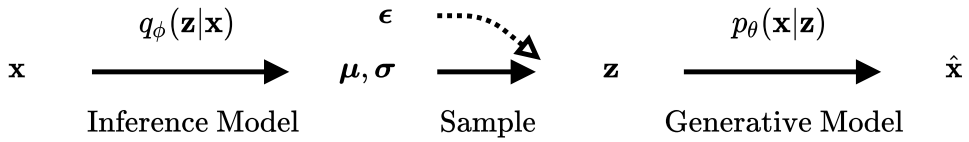


Figure 2.3: Architecture diagram of a typical VAE using a factorised Gaussian encoder. The input $\boldsymbol{x}$ is fed through the inference model to produce $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ which parameterise the multivariate normal posterior distribution. Differentiable sampling from this distribution is implemented using the *reparameterisation trick*. The random variable $\boldsymbol{z}$ is rewritten using another random variable $\boldsymbol{\epsilon}$ along with the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. The generative model then decodes the sampled representation to obtain the reconstruction $\hat{\boldsymbol{x}}$.

We note that alternative, more complex distributions can be used, such as the multivariate Gaussian with full covariance, the Bernoulli distribution for binary data [Kingma and Welling 2014], or even a combination of continuous and discrete latent variables as in the case of the appropriately named JointVAE [Dupont 2018]. However, throughout this work we use the multivariate Gaussian distribution with diagonal covariance as described above.

In practice, the probabilistic reconstruction terms are also replaced with deterministic approximations such as the *Mean Squared Error* (MSE) loss which is derived from the normal distribution, or the *Binary Cross-Entropy* (BCE) loss which is derived from the Bernoulli distribution. A prevalent error in research is that BCE loss is used for datasets which have continuous $[0, 1]$ instead of binary values $\{0, 1\}$, in such cases the MSE loss should be used instead. To avoid such issues, we use the MSE loss by default throughout this work.

2.1.4 Optimising VAEs with the AEVB Algorithm

With an understanding of the theory behind VAEs, as well as an example of a practical implementation, we can finally train VAEs using the Auto-Encoding Variational Bayes (AEVB) algorithm proposed by Kingma and Welling [2014].

The AEVB algorithm, as seen in Algorithm 1, is a dual stochastic optimisation method that operates over random minibatches sampled from the dataset $\mathcal{B} \subseteq \mathcal{X}$. An additional $|\mathcal{B}|$ corresponding set of noise samples $\mathcal{E}$ are used to enable the reparametrisation trick. The computed loss for each step is the average loss over all samples in that minibatch,

see Equation 2.6. Computing the loss over minibatches is intended to approximate the loss that would be computed over the entire dataset, since computing the loss directly over the entire dataset is often intractable.

$$\tilde{\mathcal{L}}_{\text{VAE}}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathcal{B}, \mathcal{E}) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \mathcal{L}_{\text{VAE}}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathcal{B}_i, \mathcal{E}_i) \tag{2.6}$$

Algorithm 1: Auto-Encoding Variational Bayes (AEVB) algorithm proposed by Kingma and Welling [2014].

$(\boldsymbol{\phi}, \boldsymbol{\theta}) \leftarrow$ Initialise parameters
while *parameters* $\boldsymbol{\phi}, \boldsymbol{\theta}$ *are not converged* **do**
  $\mathcal{B} \leftarrow$ Random minibatch drawn from full dataset X
  $\mathcal{E} \leftarrow$ Sample noise $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$ for each observation in $\mathcal{B}$ (see Section 2.1.3)
  Compute loss $\tilde{\mathcal{L}}_{\text{VAE}}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathcal{B}, \mathcal{E})$ over minibatch
  Compute gradients $\nabla_{\boldsymbol{\theta}, \boldsymbol{\phi}} \tilde{\mathcal{L}}_{\text{VAE}}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathcal{B}, \mathcal{E})$ over minibatch
  $\boldsymbol{\phi}, \boldsymbol{\theta} \leftarrow$ Update parameters using an SGD optimiser

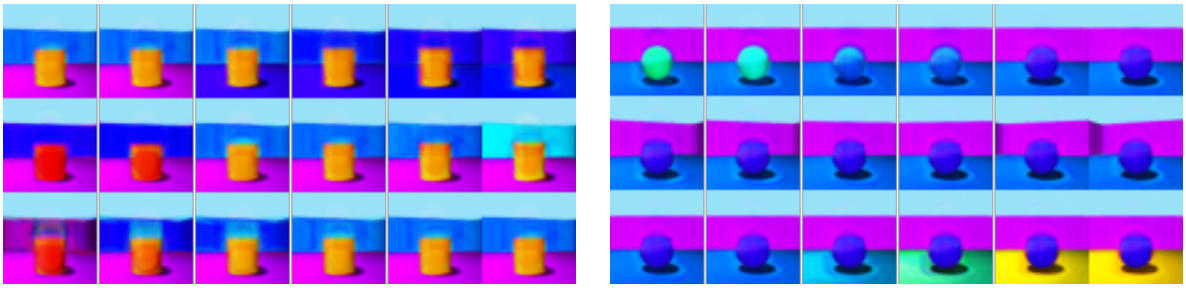
2.2 Disentanglement

Disentanglement is a form of representation learning where the task is to uncover the true underlying factors that generate data [Bengio *et al.* 2013; Burgess *et al.* 2017; Locatello *et al.* 2019b; Zaidi *et al.* 2020]. While there is no consensus as to what constitutes a disentangled representation, the usual goal is that learnt representations should be factored. A representation is considered factored when manipulation of one latent variable corresponds to a change in one underlying generative factor of the data. A generative factor can be anything, for example, the x and y position of an object, the shape, or even the colour of the object itself; however, these factors are inherently subjective.

It has been argued that learning disentangled representations over data has various benefits, including improved human interpretability, fairness when making predictions, and downstream task performance [Locatello *et al.* 2019ab]. Many VAE frameworks have been proposed that aim to improve disentanglement results. We highlight the notable frameworks that are applicable to our work in Section 2.3.

In the past, learnt representations have typically been evaluated through visual inspections of latent traversals; however, more recent approaches propose evaluation metrics to measure disentanglement properties. An example visual inspection is given in Figure 2.4 where a latent representation is sampled and the effect of changing only one unit of the representation is examined. The problem with using disentanglement metrics instead of visual inspections, is that the generative factors of the datasets need to be known. We cover common *ground-truth datasets* in Section 2.2.1 and common

disentanglement scores that make use of these datasets to evaluate representations in Section 2.2.2.



(a) Example Entangled Model **(b)** Example Disentangled Model

Figure 2.4: Example latent traversals of entangled and disentangled VAE models trained on the 3DShapes dataset [Burgess and Kim 2018]. Each row represents the variation of a single latent unit of an otherwise fixed representation. These representations are then decoded to produce each corresponding reconstruction image. We visually examine these latent traversals to ensure that only one ground-truth factor is affected. (a) The entangled model mixes multiple ground-truth factors into each latent unit, which is unintuitive from a human perspective. (b) The disentangled model separately encodes each ground-truth factor in its own latent unit, which is intuitive from a human perspective.

2.2.1 Disentanglement Datasets

When measuring disentanglement, we need ground-truth references to compare learnt representations against. Thus, the datasets used when evaluating disentanglement are usually generated from $\mathrm{F} \in \mathbb{N}_1$ ground-truth factors of variation.[2] Each individual ground-truth *factor* $i \in [\mathrm{F}]$ represents some property about the data that can be varied, and has size of $f_i \geq 1$ or rather a fixed number of values that it is allowed to take. $f_i \in \mathbb{N}_1$ is a positive integer.[3] The set of *all factor vectors* used for generating the dataset is written as $\mathcal{Y} = [f_1] \times ... \times [f_{\mathrm{F}}]$.[4] A single sampled factor vector from the dataset $\boldsymbol{y} \in \mathcal{Y}$ is the vector $\boldsymbol{y} = (y_1, ..., y_{\mathrm{F}})$, such that for each $i \in [\mathrm{F}]$ we have $1 \leq y_i \leq f_i$. The dataset observations $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ are now generated from the ground-truth factors $\boldsymbol{y} \in \mathcal{Y}$ by some ground-truth generative process $\boldsymbol{x} = g_?(\boldsymbol{y})$ that we do not have access to during the training process. An example of this generative process is given in Figure 2.5.

Common datasets used in disentanglement literature that are generated from ground-truth factors include dSprites [Matthey *et al.* 2017], 3D Shapes [Burgess and Kim 2018], Cars 3D [Reed *et al.* 2015], Small NORB [LeCun *et al.* 2004] and MPI3D [Gondal

[2]The set of all natural numbers excluding zero is denoted $\mathbb{N}_1 = \{1, 2, 3, ...\}$.
[3]Note that $[\mathrm{F}]$ is the bracket notation for the set of natural numbers $\{1, ..., \mathrm{F}\}$.
[4]If $\mathcal{A}$ and $\mathcal{B}$ are sets, then the Cartesian product is denoted $\mathcal{A} \times \mathcal{B} = \{(a, b) \mid a \in \mathcal{A} \text{ and } b \in \mathcal{B}\}$

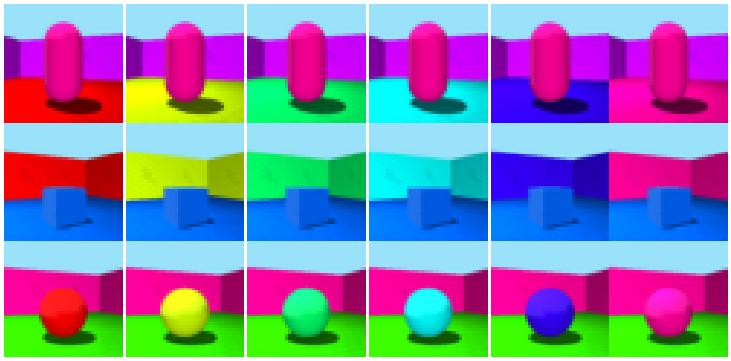


Figure 2.5: Each row is a traversal along a single ground-truth factor i, which combined are used to generate the 3D Shapes [Burgess and Kim 2018] dataset. A traversal only varies its single associated factor $i \in [\mathrm{F}]$ keeping the remaining $j \in [\mathrm{F}] \setminus \{i\}$ factors fixed. The first row varies the factor that corresponds to the floor colour, the second row the wall colour, and the third row the object colour.

et al. 2019]. While not generated from ground-truth factors, the annotated CelebA [Liu *et al.* 2015] dataset is also commonly used; however, visual inspections are often required to evaluate learnt representations in this case instead of metrics. Throughout this work, we evaluate disentanglement using the four common datasets: dSprites, 3D Shapes, Cars 3D and Small NORB. These datasets provide a range of more intuitive (Shapes 3D & dSprites) to more real and subjective (Cars 3D & Small NORB) disentanglement tasks, however, we examine further properties in later chapters.

2.2.2 Disentanglement Scores

To evaluate the disentanglement of learnt representations, various *disentanglement scores* or metrics have been proposed [Higgins *et al.* 2017; Eastwood and Williams 2018; Kim and Mnih 2018; Chen *et al.* 2018; Kumar *et al.* 2018; Ridgeway and Mozer 2018; Suter *et al.* 2018; Locatello *et al.* 2019a; Sepliarskaia *et al.* 2019]. Different disentanglement scores have different assumptions and biases as to what constitutes a disentangled representation, and thus in practice multiple different disentanglement scores should be used [Locatello *et al.* 2019b].

We adopt the standardised descriptions of disentanglement metrics from Zaidi *et al.* [2020] where the authors group metrics into three main categories based on the disentanglement property that is measured. **Modularity** is the measure of factor independence or how much factors are unaffected by others in the representation space. **Compactness** is the measure of how much of the representation space controls a single factor. **Explicitness** is the measure of the information content of representations or how well one can retrieve the true factors from the representation space.

Widely used disentanglement metrics include:

- **Beta-VAE Score** [Higgins *et al.* 2017]: Or *Z-diff score* [Zaidi *et al.* 2020], measures

the modularity of representations. The score operates by sampling and encoding multiple observations with one ground-truth factor fixed between them. The intuition is that there should be a single single latent variable which has a distance of zero between other representations in the batch, since this latent unit corresponds to the fixed ground-truth factor. A linear classifier is trained upon such sampled batches to predict which factor was fixed. The final accuracy of this classifier is taken to be the resulting β-VAE score.

- **Factor-VAE Score** [Kim and Mnih 2018]: Or *Z-min Variance score* [Zaidi *et al.* 2020], measures the modularity of representations. This score improves upon the β-VAE score (Z-diff score) by scaling representations according to their standard deviation when computing distances and performing classification.
- **DCI Disentanglement Score** [Eastwood and Williams 2018]: DCI consists of three scores: the *Disentanglement score* (measuring modularity), the *Completeness score* (measuring compactness) and the *Informativeness score* (measuring explicitness) [Zaidi *et al.* 2020]. Throughout this work we use the DCI Disentanglement score which is computed from the inner parameters of regression models trained to predict factors from representations. Typically, the Gini importance values from random forest methods are used to calculate the score [Zaidi *et al.* 2020].
- **MIG Score** [Chen *et al.* 2018]: The Mutual Information Gap (MIG) measures the compactness of representations, or whether factors are expressed by only one latent variable. MIG computes the Mutual Information (MI) between each latent variable and ground-truth factor. The difference between the highest and second highest values for each factor is computed and then normalised over the sum of MI for that factor. The final normalised differences are then averaged over all factors as the final MIG score.

We describe the Factor-VAE score because it closely resembles our proposed metrics in Chapter 4. Otherwise, throughout this dissertation, we mainly choose the widely used MIG and DCI Disentanglement scores for evaluating representation disentanglement. While these metrics may not be appropriate for all tasks since disentanglement is ultimately subjective, we do find that these scores are a good measure of disentanglement throughout this work with regards to our goals.

2.3 Disentanglement Frameworks

With the rise in popularity of disentanglement over recent years, many unsupervised approaches aimed at improving disentanglement have been suggested [Higgins *et al.* 2017; Zhao *et al.* 2017; Hou *et al.* 2017; Kumar *et al.* 2018; Chen *et al.* 2018; Kim and Mnih 2018]. Unfortunately, Locatello *et al.* [2019b] show that learning disentangled representations in an unsupervised manner is unreliable and fundamentally impossible without inductive biases. The same authors introduce a state-of-the-art weakly-supervised method called the *Ada-GVAE* (Adaptive GVAE) in Locatello *et al.* [2020]. This weakly-supervised framework significantly outperforms all of the aforementioned unsupervised methods.

2.3.1 β-VAE

The β-VAE (Beta-VAE) was proposed by [Higgins *et al.* 2017] and is a simple modification to the original unsupervised VAE framework outlined by Kingma and Welling [2014] aimed at improving disentanglement. The β-VAE adds a single hyper-parameter $\beta \geq 0$ as a coefficient to the regularisation component of the standard VAE loss function (see Equation 2.1). When $\beta = 1$ the β-VAE loss is equivalent to the original VAE loss.

$$\mathcal{L}_{\beta\text{-VAE}} = \mathcal{L}_{\text{rec}} + \beta\mathcal{L}_{\text{reg}} \tag{2.7}$$

The non-negative coefficient β constrains the capacity of the latent information channel $\boldsymbol{z}$. Increasing the strength of the regulariser with larger values for β generally results in better disentanglement scores, but more blurry reconstructions, since information that reduces the reconstruction loss needs to be prioritised. Conversely, lower values for β generally result in worse disentanglement scores but better reconstructions [Burgess *et al.* 2017], since it is easier to transmit information through the latent bottleneck.

2.3.2 Non-identifiability of Unsupervised Models

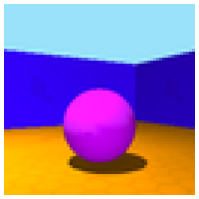
The key idea behind this section is that of *identifiability*. A model is said to be identifiable if, theoretically, it is possible to learn its true underlying parameters after infinite observations [Hsu *et al.* 2012]. However, Locatello *et al.* [2019b] provide a proof for non-identifiability in the case of unsupervised disentanglement methods, termed the *impossibility result*. The proof shows that with access only to ground-truth observations $\boldsymbol{x}$ these methods cannot distinguish between two equivalent models that have learnt different latent representations $\boldsymbol{z}^{(a)}$ and $\boldsymbol{z}^{(b)}$. This stems from the fact that both generative models have the same marginal distributions by construction in Equation 2.8.

$$\int p(\boldsymbol{x}|\boldsymbol{z}^{(a)})p(\boldsymbol{z}^{(a)})d\boldsymbol{z}^{(a)} \;=\; p(\boldsymbol{x}) \;=\; \int p(\boldsymbol{x}|\boldsymbol{z}^{(b)})p(\boldsymbol{z}^{(b)})d\boldsymbol{z}^{(b)} \tag{2.8}$$

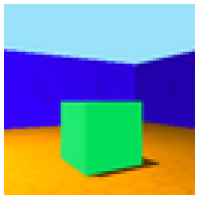
The result of this proof is that while unsupervised methods can encourage desirable properties, supervision is required to identify disentangled models correctly. Furthermore, Locatello *et al.* [2019b] confirm this proof through experimental results over common unsupervised frameworks. The work shows that ultimately hyper-parameters and random seeds, not special algorithmic choices, have a far more significant effect on disentanglement in the unsupervised case. Locatello *et al.* [2019b] thus argue that future investigations into disentanglement should place a greater focus on choosing suitable inductive biases or another form of supervision to bypass the non-identifiability result.

2.3.3 Adaptive VAEs

Following on from the non-identifiability result, Locatello *et al.* [2020] proposed a weakly supervised disentanglement framework that estimates which factors of variation in the latent space have changed between observation pairs. The latent variables estimated to be shared between the observations in the pair are averaged together before being decodes, thus encouraging shared representations as seen in Figure 2.6. Furthermore, assuming infinite data, a proof is provided which shows that the non-identifiability result is bypassed by this framework. This is due to the use of weak supervision to provide non-i.i.d observation pairs, instead of the single i.i.d observations which are typical in the unsupervised setting.



$\mathbf{z}_1 \quad [1, 7, 4, 4, 0, 0]$



$\mathbf{z}_2 \quad [1, 7, 8, 4, 2, 0]$

Figure 2.6: Example pair of observations sharing a subset of latent factors, except for the object colour and shape, represented by the red components of $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$. The 3DShapes dataset pictured and similar datasets procedurally generated from independent ground-truth factors are often used to benchmark disentanglement frameworks. The 6 ground-truth factors in Shapes3D are the floor colour, wall colour, object colour, scale, shape and orientation [Burgess and Kim 2018].

Formally, let k be the total number of ground-truth factors in which the observations $\boldsymbol{x}^{(a)}$ and $\boldsymbol{x}^{(b)}$ differ, D be the number of latent variables such that $\boldsymbol{z} \in \mathbb{R}^{\mathrm{D}}$, $S \subseteq [\mathrm{D}]$ be the subset of *shared* indices of size $\mathrm{D} - k$, and $\bar{S} = [\mathrm{D}] \setminus S$ be the remaining set of *differing* indices. [5] During training, for a pair of parameterised posterior distributions corresponding to $\boldsymbol{x}^{(a)}$ and $\boldsymbol{x}^{(a)}$, the constraint in Equation 2.9 is enforced. This constraint ensures that the latent units which are considered shared should also have the same values or parameterisations.

$$\begin{aligned} q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}) &= q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)}) \quad \forall i \in S \\ q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}) &\neq q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)}) \quad \forall i \in \bar{S} \end{aligned} \tag{2.9}$$

This constraint is enforced by computing new aggregate posteriors for both observations as seen in Equations 2.10 and 2.11. The new aggregate posteriors are obtained

[5] Note that $[\mathrm{D}]$ is the bracket notation for the set of natural numbers $\{1, ..., \mathrm{D}\}$.

by applying an averaging function $\mathrm{a}(\cdot,\cdot)$ between all latent distributions corresponding to shared indices in S and leaving the remaining distributions unchanged.

$$\tilde{q}_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}) = \begin{cases} \mathrm{a}\left(q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}),\ q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)})\right) & \text{if shared: } i \in S \\ q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}) & \text{if differing: } i \in \bar{S} \end{cases} \tag{2.10}$$

$$\tilde{q}_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)}) = \begin{cases} \mathrm{a}\left(q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}),\ q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)})\right) & \text{if shared: } i \in S \\ q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)}) & \text{if differing: } i \in \bar{S} \end{cases} \tag{2.11}$$

Before the aggregate posteriors can be computed, k and S need to be estimated. For each pair of latent distributions the KL divergence δ_i is computed (see Equation 2.12), followed by a threshold τ which lies half way between the min and the max over all delta values (see Equation 2.13). The set S is computed (see Equation 2.14) as the set of indices with smaller KL Divergences than this threshold. The estimate for τ relies on the assumption that not all factors have changed between samples. While it may not be the case in practice, the correct k is always returned if the encoder is correctly disentangled. We present simplified overview of this procedure, which represents the core of the Adaptive VAE algorithm, in Figure 2.7.

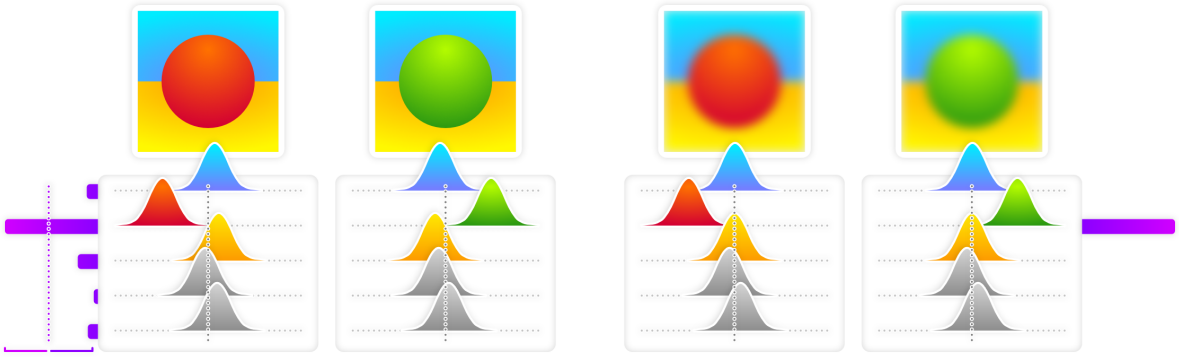


Figure 2.7: An overview of the Adaptive VAE pipeline, which estimates which factors have changed between pairs of input observations. (left) Bars on the left indicate the distances δ_i between the corresponding latent distributions parameterised by the encoder. The threshold τ is computed as halfway between the min and the max distance. Latent units with distances less than the threshold δ_i are considered shared and are then averaged together to produce the new distributions on the right. (right) Bars on the right indicate the distances δ_i between the corresponding distributions after the shared latent units have been estimated and averaging has been performed. These distributions are then sampled from during training to obtain the representations and corresponding reconstructions from decoder. The averaging procedure encourages the difference between observations to be entirely explained by the non-shared or non-averaged latent units.

$$\delta_i = \tilde{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}) \parallel q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)})\right) \tag{2.12}$$

$$\tau = \frac{1}{2}\left(\min_{i\in[\mathrm{D}]}\delta_i + \max_{i\in[\mathrm{D}]}\delta_i\right) \tag{2.13}$$

$$S = \{i \mid \delta_i < \tau,\ \forall\ i \in [\mathrm{D}]\} \tag{2.14}$$

In the original work, Locatello *et al.* [2020] compute δ_i using the standard KL divergence $D_{\mathrm{KL}}\left(p \parallel q\right)$. Dittadi *et al.* [2020] propose that the symmetric KL divergence defined as $\tilde{D}_{\mathrm{KL}}\left(p \parallel q\right) = \frac{1}{2}D_{\mathrm{KL}}\left(p \parallel q\right) + \frac{1}{2}D_{\mathrm{KL}}\left(q \parallel p\right)$ should be used instead. We adopt this improvement throughout our research.

Finally, the following aggregated variant of the β-VAE loss is optimised with the new aggregate posteriors replacing appropriate terms in the original loss functions:

$$\mathcal{L}_{\mathrm{AdaVAE}} = \frac{1}{2}\left(\mathcal{L}_{\beta\mathrm{VAE}}^{\tilde{q}_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(a)})} + \mathcal{L}_{\beta\mathrm{VAE}}^{\tilde{q}_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(b)})}\right) \tag{2.15}$$

The various adaptive methods proposed by Locatello *et al.* [2020] are named after the choice of averaging function. The *Ada-GVAE* uses the arithmetic mean [Hosoya 2019] as given in Equation 2.16 and the *Ada-ML-VAE* uses the product of the encoder distributions [Bouchacourt *et al.* 2018]. Both variants operate over multivariate Gaussian posteriors with diagonal covariance. For simplicity we refer to these adaptive methods as Adaptive VAEs.

$$\mathrm{a}(q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}),\ q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)})) = \mathcal{N}\left(\frac{1}{2}(\mu_i^{(a)} + \mu_i^{(b)}),\ \frac{1}{2}(\sigma_i^{2\ (a)} + \sigma_i^{2\ (b)})\right) \tag{2.16}$$

Results show that the more simple Ada-GVAE generally outperforms the Ada-ML-VAE [Locatello *et al.* 2020]. However, both show significant improvements over earlier unsupervised methods, with regards to initial seed invariance, training stability and the overall disentanglement performance.

2.4 Related Work

Many publications exist in the areas of VAEs and disentanglement. To the best of our knowledge, the following works are the most applicable to our research, which falls into the following broad categories: (i) explanations of disentanglement, (ii) the role of the reconstruction loss in disentanglement, (iii) problems with disentanglement, (iv) disentangling using non-VAE methods, and (v) learning to disentangle.

Firstly, Burgess *et al.* [2017] further the understanding of VAEs by relating them to the information bottleneck principle. This explains how random sampling leads to a local minimisation of the reconstruction loss by reorganising the latent space so that points

close in pixel space are close in the latent space. Mathieu *et al.* [2019] argue that VAEs do not explicitly encourage disentanglement through their design. Rather, they provide an explanation in terms of the diagonal prior typically used in VAEs combined with random sampling to produce a similar effect to PCA.

Secondly, Hou *et al.* [2017] swap out the reconstruction loss of VAEs for a perceptual loss function, which improves the representations learnt by the model for downstream tasks in a similar way to disentanglement. Zietlow *et al.* [2021] extend the analysis by Mathieu *et al.* [2019] for why VAEs disentangle. However, not much focus is placed on the reconstruction loss and its interaction between datasets or their ground-truth factors.

Thirdly, Locatello *et al.* [2019b] challenge the common assumptions about disentangled representation learning, stating that good representations cannot be reliably learnt with unsupervised methods, unless inductive biases are introduced. Furthermore, Gondal *et al.* [2019] show that often representations learnt on synthetic data do not transfer well to real-world data. Our work will show that disentanglement arises as an accidental consequence of the distances between observations in datasets aligning with ground-truth factors. Our work will also show that this suggests that there is an upper bound on how disentangled representations can be without supervision.

Fourthly, recent work by Burns *et al.* [2021] investigates learning disentangled representations using contrastive learning by adding regularisation. The advantage of this work is that it does not depend on a decoder or reconstruction loss, and representations can be learnt directly. Our work will investigate distance learning techniques which we will show can be modified to learn disentangled representations. The advantage of our techniques are that they can also be used without a decoder.

Finally, Mathieu *et al.* [2019] construct adversarial datasets using a mild transformation to hinder disentanglement performance. The mild transformation is obtained from trained models which achieve poor disentanglement scores. Our work will investigate similar techniques, however, we will instead phrase the problem as learning new reconstruction losses that can encourage or discourage behaviour, with a focus on enabling disentanglement.

Chapter 3

VAEs Learn Distances Over Data

In this chapter we investigate the reason for disentanglement in VAEs, we argue in Section 3.1 that the regularisation component of the VAE loss is not directly responsible for learning disentangled representations, rather that the interaction between random sampling and the reconstruction loss accidentally produces disentangled representations.

We then examine this mechanism in Section 3.2 using existing datasets for disentanglement and show that distances between their ground-truth factors correlate with the distances between individual data points in terms of the reconstruction loss chosen for VAEs. Furthermore, we show that VAEs mimic distances in the latent space. This reinforces the earlier reasons for why VAEs disentangle, and it further suggests that this mechanism is accidental.

In Section 3.3 we design a dataset that breaks disentanglement within state-of-the-art VAE approaches by exploiting this interaction between the reconstruction loss within VAE frameworks, which is typically chosen to be the *Mean Squared Error* (MSE). The adversarial dataset has constant distance between data points in terms of the reconstruction loss used by the VAE.

Finally, in Section 3.4 we show that by carefully designing and choosing a loss function such that the ground-truth factors are once again captured, we can disentangle the previously adversarial dataset. This loss function is designed specifically for our adversarial dataset and is intended only as an example. In practice a researcher would need to choose their loss function carefully to encourage disentanglement with VAE methods.

3.1 Random Sampling Reorganises VAE Embeddings

The component within VAE frameworks that encourages disentanglement is the random sampling from the probabilistic encoder during training. If the distributions parameterised by the encoder overlap sufficiently for different observations, the decoder will often attribute a random sample to an incorrect input (see Figure 3.1). Thus, a mistake will be made during the decoding process. The reconstruction loss of the VAE encourages the reorganisation of the latent space such that reconstruction mistakes are minimised, with nearby samples in the data space placed close together in the latent space [Burgess *et al.* 2017]. Distances between representations will thus start to mimic

distances between data in a VAE.

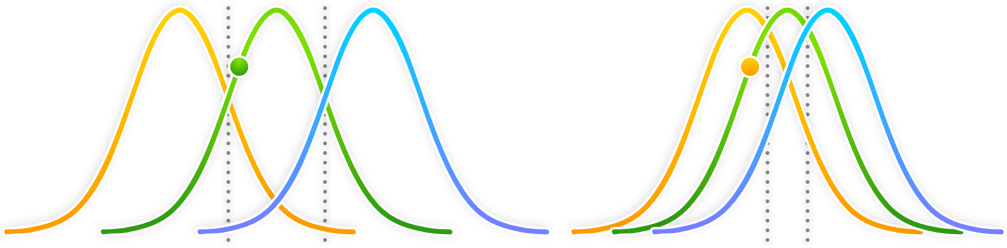


Figure 3.1: Three nearby distributions in the latent space are presented, each corresponding to different input observations. In each case, the VAE samples from the middle distribution. Left: With weaker regularisation, sampled points are more often attributed to the correct distributions and the decoder correctly reconstructs the observations. Right: With stronger regularisation, sampled points are more often attributed attributed to nearby distributions in the latent space and the decoder makes more mistakes when reconstructing the observation. The encoder thus tries to place observations that are close in the data space, close together in the embedding space.

The regularisation term in Equation 2.2 controls the overlap between latent distributions corresponding to different inputs. This is why tuning the strength of the regularisation term in β-VAE [Higgins *et al.* 2017; Locatello *et al.* 2020] based frameworks appears to improve disentanglement. Stronger regularisation leads to more latent overlap, making it easier for the decoder to reorganise the embedding space and mimic distances over data, but harder to reconstruct the input. Weaker regularisation lets the system degrade into a lookup table [Mathieu *et al.* 2019]; if few decoding mistakes are made, little reorganisation of the latent space occurs and no distances according to the loss are learnt.

The important distinction from literature is that the regularisation term does not directly produce disentangled results. Reorganisation of the latent space arises due to the interaction between the random sampling procedure and the reconstruction term in Equation 2.3; the regularisation term simply enables this interaction by keeping the probability of making decoding mistakes sufficiently high. This reorganisation of the latent space in VAEs means that the distances between different representations will often mimic distances between corresponding data. If these distances in the data space accidentally correspond to some notion of disentanglement, and the latent space is correctly rotated by chance, then individual latent units may encode portions of the distances that correspond to this same notion of disentanglement. It is known that VAEs are rotationally invariant [Mathieu *et al.* 2019], thus this mechanism for disentanglement is entirely accidental. When representations appear factored in this way, we term the idea *axis-alignment*, see Figure 3.2 for a visual example. We spend the remainder of this chapter investigating this mechanism and the various interacting components.

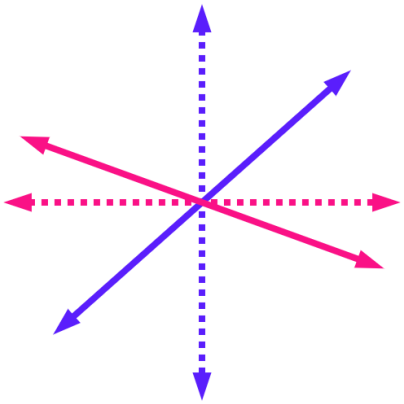


Figure 3.2: Example of *axis-alignment* of representations. If distances that correspond to some notion of disentanglement have been learnt, then the solid lines represent incorrectly rotated axes in the latent space that may correspond to axes in the data. However, the dotted lines represent the case when these axes in the latent space are correctly rotated. In this case, individual latent units may encode axes in the data that correspond to some notion of disentanglement, thus the representations will also appear factored.

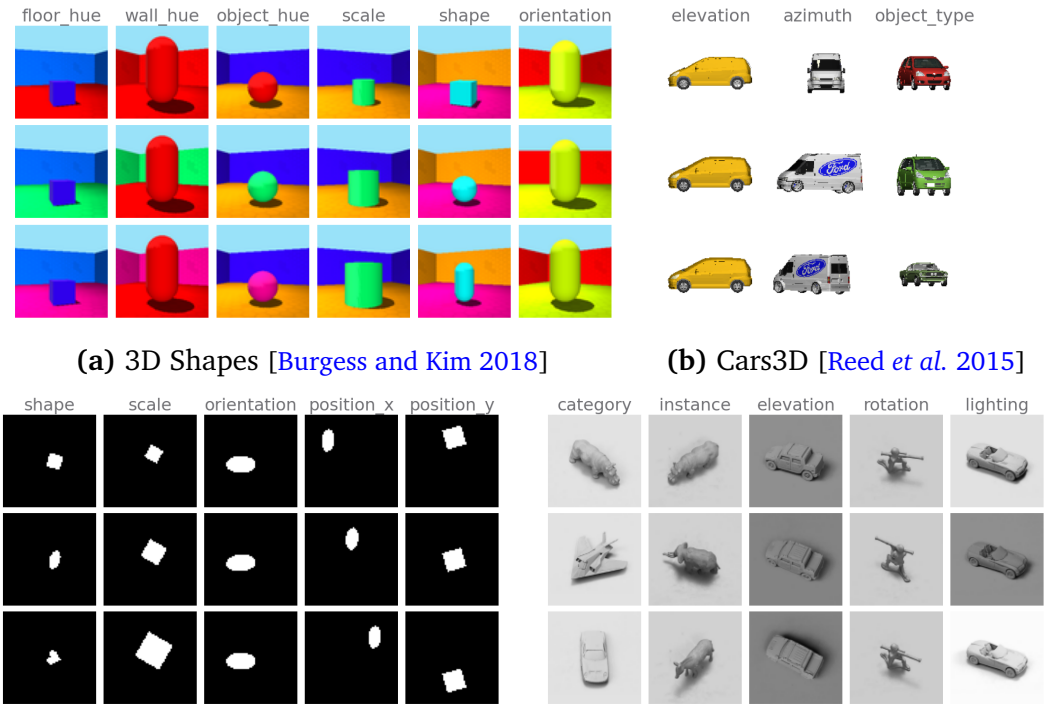
3.2 Existing Datasets

With an understanding of how VAEs reorganise the latent space, we seek to investigate how the construction of datasets interacts with this process. Consider the 3D Shapes dataset [Burgess and Kim 2018], visualised in Figure 3.3, which contains observations of shapes fixed in the centre of the image with progressively changing attributes or factors such as size and colour. If, as humans, we are given unordered observations from a traversal along the size factor of 3D Shapes, it would be easy to order these observations using a perceived increase or decrease in the size of the shape. We might even say that the shapes in the images overlap by different amounts. We may consider shapes that are closer in size to possess more overlap, and therefore also consider them to be closer together in terms of distance.

It is natural to extend this idea to VAEs constructing orderings over pairs of observations, using both the ground-truth factors as well as how the frameworks themselves perceive the distances between data points.

3.2.1 Ground-Truth Distance

Synthetic datasets [Burgess and Kim 2018; LeCun *et al.* 2004; Matthey *et al.* 2017; Reed *et al.* 2015; Gondal *et al.* 2019] used for benchmarking disentanglement frameworks are all generated from $\mathrm{F} > 0$ ground-truth factors of variation. We refer back to Section 2.2.1 where the set of all ground-truth factors for each dataset is the product of all individual ground-truth factors $\mathcal{Y} = [f_1] \times ... \times [f_\mathrm{F}]$. It is fitting to describe the *ground-*



(a) 3D Shapes [Burgess and Kim 2018] **(b)** Cars3D [Reed *et al.* 2015]

(c) dSprites [Matthey *et al.* 2017] **(d)** NORB [LeCun *et al.* 2004]

Figure 3.3: Common existing datasets used to benchmark disentanglement frameworks. These synthetic datasets are generated from the set product of all their ground-truth factors. Columns represent a traversal along a single factor, where only this chosen factor is varied while the others are kept constant. A factor should ideally represent one property about the data that can vary.

truth distances between observations $\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)} \in \mathcal{X}$ using the ℓ_1 or Manhattan distance between their corresponding ground-truth factors $\boldsymbol{y}^{(a)}, \boldsymbol{y}^{(b)} \in \mathcal{Y}$,[1] as in Equation 3.1.

$$\mathrm{d}_{\mathrm{gt}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = \|\boldsymbol{y}^{(a)} - \boldsymbol{y}^{(b)}\|_1 \tag{3.1}$$

We use the ground-truth distance extensively throughout this work when introducing supervision in later chapters, or when analysing how easy it is to disentangle existing datasets. While the ℓ_1 distance may not always be a natural measure of the ground-truth factors themselves, we choose the Manhattan distance because it fundamentally corresponds to the way the ground-truth datasets are generated.

[1]For convenience, note that $\boldsymbol{y}^{(a)} = a$ such that $\boldsymbol{x}^{(\boldsymbol{y}^{(a)})} = \boldsymbol{x}^{(a)}$

3.2.2 Perceived Distance

With the idea of ground-truth distances between observations, we need some measure of the distance between observations as perceived by VAE frameworks. We derive the *perceived distance* or *visual distance* between dataset elements from the noisy sampling procedure and the chosen reconstruction loss in a VAE framework.

Let $\boldsymbol{z}^{(b)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(a)})$ be a (possibly incorrect) sample from the posterior distribution corresponding to some input element $\boldsymbol{x}^{(a)} \in \mathcal{X}$. Since the regularisation term encourages latent distributions to overlap, this sample $\boldsymbol{z}^{(b)}$ may be incorrectly attributed by the decoder to a distribution corresponding to some other element from the dataset $\boldsymbol{x}^{(b)} \in \mathcal{X}$, with reconstruction $\hat{\boldsymbol{x}}^{(b)} \approx \boldsymbol{x}^{(b)}$.

As the VAE objective consisting of the regularisation and reconstruction losses is jointly optimised, the decoder becomes better at reconstructing the inputs. In an ideal scenario, the inputs map to outputs ($\hat{\boldsymbol{x}} \to \boldsymbol{x}$), and reconstructions are samples from our dataset: $\hat{\boldsymbol{x}} \in \mathcal{X}$. While this is not the case in practice due to the regularisation term, we derive the perceived distance in Equation 3.3 from this assumption that $\hat{\boldsymbol{x}} \to \boldsymbol{x}$. This allows us to directly compare the elements $\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)} \in \mathcal{X}$ within a dataset using the reconstruction loss as a distance function:

$$\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = \lim_{\hat{\boldsymbol{x}} \to \boldsymbol{x}} \mathcal{L}_{\mathrm{rec}}(\boldsymbol{x}^{(a)}, \hat{\boldsymbol{x}}^{(b)}) \tag{3.2}$$

$$= \mathcal{L}_{\mathrm{rec}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \tag{3.3}$$

The perceived distance above depends on the choice of reconstruction loss. The usual choice in literature is the pixel-wise Mean Squared Error (MSE) for data that is assumed to be normally distributed. We assume that MSE loss is used throughout the rest of the work, unless otherwise specified. However, any analysis should be very similar for other pixel-wise loss functions, such as Binary Cross-Entropy (BCE).

3.2.3 Perceived Overlap

In the previous section, we derive and define the perceived distance as a distance measure between data points in terms of the reconstruction loss of a VAE. We similarly define the inverse term, the *perceived overlap*, as a similarity measure between data points. The higher the perceived overlap, the more similar observations are and the lower the perceived distance, the lower the perceived overlap the less similar observations are in terms of the reconstruction loss and the higher the perceived distance. Perceived overlap in Equation 3.4 is simply the negated perceived distance term in Equation 3.3.

$$\mathrm{v}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = -\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \tag{3.4}$$

3.2.4 Perceived Distances Correspond To Ground-Truth Distances In Benchmark Datasets

In Section 3.2.1, all the ground-truth factors of a dataset are defined as the set $\mathcal{Y} = [f_1] \times \ldots \times [f_{\mathrm{F}}]$. In Equation 3.6, we now define a *factor traversal* $\mathcal{Y}^{(a,i)} \subset \mathcal{Y}$ as the ordered set of all the coordinates along a factor $i \in [\mathrm{F}]$ such that the set passes through a point $\boldsymbol{y}^{(a)} \in \mathcal{Y}$. The number of elements in the traversal is equal to the size of the chosen factor $|\mathcal{Y}^{(a,i)}| = f_i$, and each element in the traversal generates the same traversal $\forall \boldsymbol{y}^{(b)} \in \mathcal{Y}^{(a,i)},\ \mathcal{Y}^{(a,i)} = \mathcal{Y}^{(b,i)}$. Examples of ground-truth factor traversals are given in Figure 3.3.

$$\mathcal{Y}^{(a,i)} = \ldots \times \left\{ y_{i-1}^{(a)} \right\} \times [f_i] \times \left\{ y_{i+1}^{(a)} \right\} \times \ldots \tag{3.5}$$

$$= \left\{ (\ldots,\ y_{i-1}^{(a)},\ j,\ y_{i+1}^{(a)},\ \ldots) \mid \forall j \in [f_i] \right\} \tag{3.6}$$

In Equation 3.7, we compute the distance matrix $\tilde{D}^{(a,i)} \in \mathbb{R}^{f_i \times f_i}$, for some distance function d, between pairwise elements along a factor traversal $\mathcal{Y}^{(a,i)}$.[2]

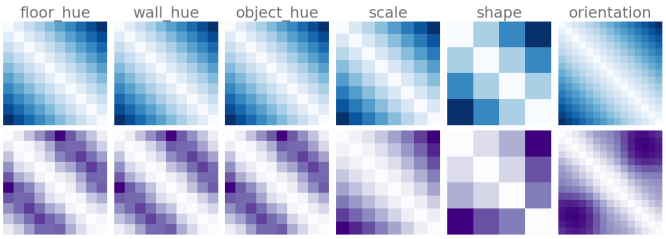
$$\tilde{D}^{(a,i)} = \left(\mathrm{d}(\boldsymbol{x}^{(u)}, \boldsymbol{x}^{(v)}) \right) \in \mathbb{R}^{f_i \times f_i} \quad \forall u, v \in \mathcal{Y}^{(a,i)} \tag{3.7}$$

To characterise the factors within our dataset, we compute the average distance matrices $D^{(i)} = \mathbb{E}_{a \in \mathcal{Y}}[\tilde{D}^{(a,i)}]$ for each factor $i \in [\mathrm{F}]$. We plot these results in Figure 3.4 for the ground-truth distance d_{gt} and perceived distance $\mathrm{d}_{\mathrm{pcv}}$. It is immediately obvious from these plots that the ground-truth and perceived distances by a VAE correspond. If a VAE reorganises the embedding space to minimise the reconstruction error due to random sampling (see Section 3.1), then the VAE would learn latent distances that correspond to the perceived distances over the data, however, these perceived distances and thus latent distances would then naturally correspond to the ground-truth distances. The VAE would thus discover distances that are similar to the ground-truth factor distances.

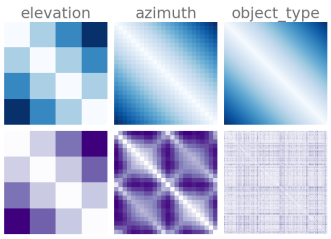
We further examine this in Figure 3.5, where we compute distance matrices over a trained β-VAE at various levels of the network, including the representation layer and reconstructions. At each level of the VAE, the learnt distances all correspond to the original perceived distances already present within the dataset due to the chosen loss.

Since a VAE with a factorised Gaussian prior is known to be rotationally invariant [Mathieu *et al.* 2019], the same distances between the means $\boldsymbol{\mu}$ of latent distributions can be learnt for any arbitrary rotation of the latent space. Our results in Figures 3.4 and 3.5 provide empirical evidence that VAEs mimic the distances already present in

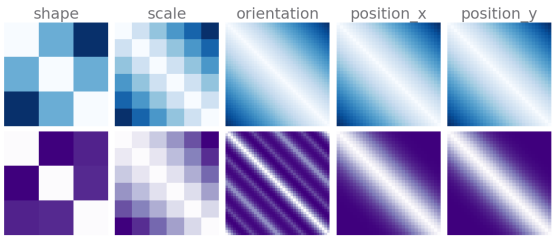
[2] We use the matrix notation where $A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} = (a_{ij}) \in \mathbb{R}^{m \times n}$



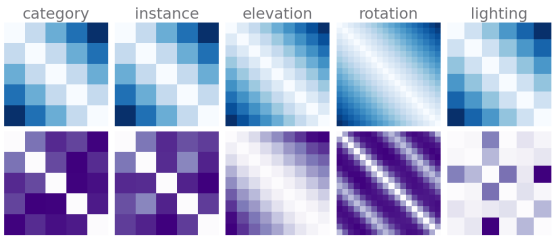
(a) 3D Shapes [Burgess and Kim 2018]



(b) Cars3D [Reed *et al.* 2015]



(c) dSprites [Matthey *et al.* 2017]



(d) NORB [LeCun *et al.* 2004]

Figure 3.4: Distances in the ground-truth factor space naturally correspond to distances in the data space for current synthetic datasets. Top Row: Average Manhattan distance matrices over factor traversals. Bottom Row: Average pixel-wise perceived distance matrices (MSE) over observations from the same factor traversals. Columns: Different ground-truth factors within each dataset.

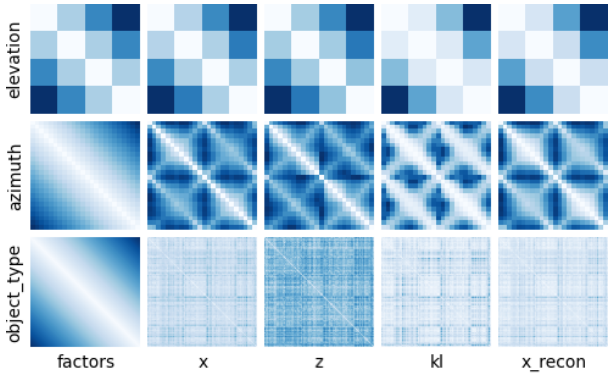


Figure 3.5: β-VAEs learn similar distances between observations at each level of the network depending on the reconstruction loss. Rows: Different factors of the Cars3D dataset (Top to bottom: elevation, azimuth, car type), Columns: Different sources of distance matrices computed over factor traversals (Left to right: ground-truth distances, perceived distances between observations, ℓ_2 distances over latent distribution means, KL divergences between latent distributions, perceived distances between reconstructions).

the dataset according to the reconstruction loss. This suggests that VAEs disentangle these datasets by accident; if ground-truth factors naturally correspond with distances in the dataset, and the latent space is correctly rotated, then individual latent units

can encode distances that correspond to individual ground-truth factors. If these perceived distances were to change such that they do not correspond to the ground-truth distances, VAEs might not be able to learn meaningful representations. This is highlighted by the fact that VAEs are already known to disentangle poorly on real-world data [Gondal *et al.* 2019]. Real-world data often has noise, imperfections or backgrounds which may mask the distances and factors that we care about.

3.3 Adversarial Dataset

In the previous section, we highlighted the striking similarity between the ground-truth distances and the perceived distances between observations in a dataset. This suggests that disentanglement occurs because latent distances learnt by VAEs accidentally correspond to ground-truth distances; this is because the latent space is reorganised to capture perceived distances in the data space.

Consider an example where a single chess piece moves across a chess board and there are no smooth transitions between grid points, since the piece is only valid when placed in the middle of squares. We describe such a dataset as having constant perceived overlap or constant perceived distance. It is also adversarial, because it is impossible to order observations using pairwise distance comparisons. While a change in the position of the chess piece means a change of the ground-truth factor, the perceived overlap between positions still remains constant due to the non-overlapping nature of the positions. It may be tempting to think that a harder case to solve is if the perceived distances simply do not correspond to ground-truth distances; however, an ordering (although incorrect) can still be found. Existing datasets such as Cars3D (see Figure 3.4) already satisfy this property, which may explain the generally worse disentanglement performance compared to other datasets.

Formally, we say that a dataset has constant overlap when the pairwise distances over factor traversals are all equal. Let $i \in [\mathrm{F}]$ be a factor and $\boldsymbol{y}^{(a)} \in \mathcal{Y}$ be ground-truth coordinate vector. Then, for all elements over the factor traversal $\forall \boldsymbol{y}^{(b)} \in \mathcal{Y}^{(a,i)}/\boldsymbol{y}^{(a)}$, the corresponding perceived overlap is constant such that $\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = C_f$ with $C_f \in \mathbb{R}$ and $C_f > 0$. Along factor traversals in such a dataset, no distinct ordering of elements can be found when a VAE tries to minimise the sampling error over the reconstruction loss. Going forward, we only consider the case where $\forall f \in [\mathrm{F}],\ C_f = C$ for some $C > 0$.

3.3.1 XYSquares Dataset

Taking inspiration from the chess piece example, we design a synthetic adversarial dataset called *XYSquares*, illustrated by Figure 3.6, to exploit the VAE weakness of constant overlap when traversing factors. The dataset consists of three squares, where each square is 8×8 pixels in size. With an observation size of 64×64 pixels, this leaves 8 grid positions along each x and y axis while avoiding any pixel-wise overlap. With this

construction, as a square moves, its pixels never overlap with previous locations on the grid. The three squares are each assigned a colour according to R $(1, 0, 0)$, G $(0, 1, 0)$ and B $(0, 0, 1)$ to additionally avoid any channel-wise overlap. With 6 ground-truth factors (three squares moving along two axes), each with 8 possible values, this gives a total dataset size of $8^6 = 262144$ observations.

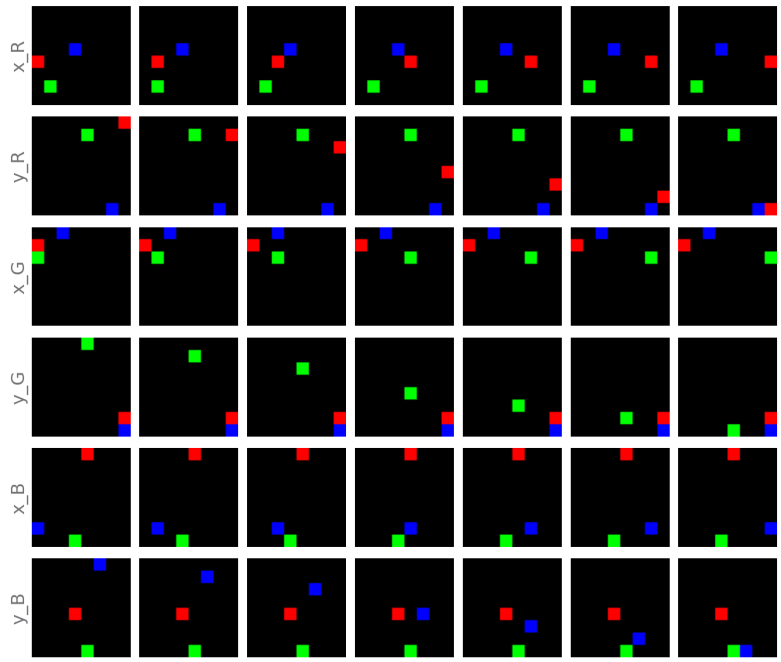


Figure 3.6: Rows represent ground-truth factor traversals over our adversarial XYSquares dataset. The dataset consists of 3 squares, each of which can move along 8 distinct x and y grid positions, for a total of 6 ground-truth factors and $8^6 = 262144$ observations. A pixel-wise distance measure has constant values between any two differing observations along a factor traversal.

We validate the design of the dataset in Figure 3.9 against other commonly used datasets by comparing the perceived distance between pairs of observations along factors. The results in Figure 3.4 show that when randomly sampling differing pairs along factor traversals, our dataset maintains constant pixel-wise distance values.

3.3.2 Experimental Setup

We now investigate the performance of VAEs on our new dataset. In particular, we use the unsupervised β-VAE [Higgins *et al.* 2017] and the state-of-the-art weakly supervised Ada-GVAE [Locatello *et al.* 2020]. The β-VAE scales the VAE regularisation term with a coefficient $\beta > 0$, while the Ada-GVAE encourages axis alignment and shared latent variables between pairs of observations. This is achieved by averaging together latent distributions between observation pairs that are estimated to remain unchanged when the KL divergence is below some threshold. We note that if the weakly supervised Ada-

GVAE performs poorly, then it is highly likely that another unsupervised method will also perform poorly.

We use the same standardised Adam [Kingma and Ba 2015] optimiser and convolutional neural architecture as Burgess *et al.* [2017], with fully connected networks around the bottleneck layer.

To evaluate disentangled representations, we use the MIG [Chen *et al.* 2018] (Mutual Information Gap) and DCI Disentanglement [Eastwood and Williams 2018] scores. MIG measures the mutual information between the highest and second highest latent units for each factor, and DCI Disentanglement measures how much each latent unit captures a ground-truth factor using a predictive model.

Finally, we perform an extensive hyper-parameter grid search on existing frameworks and datasets before running our own experiments. Hyper-parameters include the learning rate, size of the latent dimension, training steps, batch size and β values. See Appendix A for further details on all experiments conducted throughout the remainder of this work.

3.3.3 Adversarial Experiments

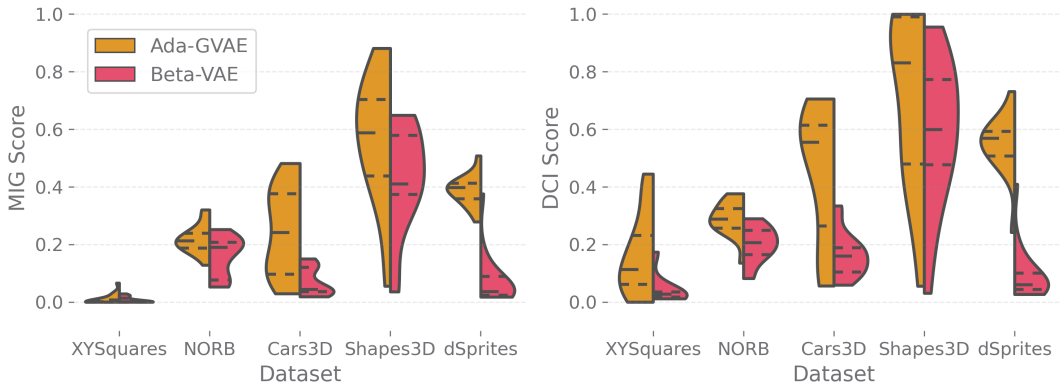


Figure 3.7: MIG score (Left) and DCI Disentanglement score (Right) for the weakly-supervised Ada-GVAE (Left half of densities) and β-VAE (Right half of densities) on commonly used datasets compared to our adversarial dataset (x axis) with constant pixel-wise perceived overlap. Our adversarial dataset hurts the disentanglement performance significantly. Quartiles are marked with horizontal lines. Multiple runs are performed over a sweep of β values and latent dimension sizes. See Appendix A for further experiment details.

The results in Figure 3.7 show that the disentanglement performance over XYSquares is extremely poor compared to existing datasets, even with the state-of-the-art Ada-GVAE. We are not concerned with the distribution of scores, rather the maximum score obtained for each model and dataset. This validates our adversarial dataset hypothesis

in Section 3.3. Not only is the disentanglement performance poor, but much smaller values for β are needed when tuning the regularisation loss.

3.3.4 Varying Levels of Overlap

We have examined the effect of training on existing datasets with significant amounts of overlap, as well as our own adversarial dataset with constant overlap. However, we have not investigated increasing levels of overlap in datasets. To do so, we modify XYSquares by decreasing the spacing between grid points while keeping the number of grid points constant along each factor, such that the size of the dataset remains the same at $8^6 = 262144$ observations. See Figure 3.8.

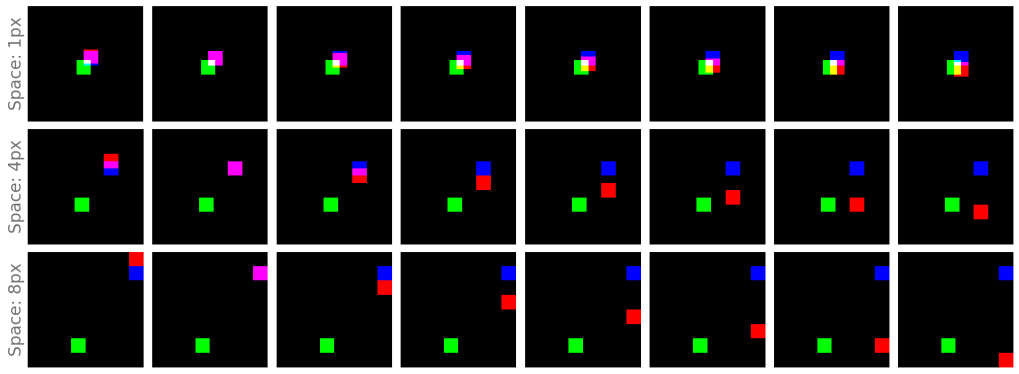


Figure 3.8: Different versions of the XYSquares dataset, where the spacing between grid locations is adjusted. The top row has the most degrees of perceived overlap between data points since the square spacing is at a minimum of 1px, the bottom row is the original adversarial dataset with constant spacing between square positions and thus constant perceived distance between data points. The size and ground-truth factors of the datasets otherwise remain constant.

The original adversarial dataset with a spacing of 8 has a constant distance value of $\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = C_8$. As the spacing s decreases from $8 \rightarrow 1$ over the datasets, the probability increases that any two observations re-sampled along a single factor traversal overlap $p(\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) < C_8)$ and should thus be placed closer together in the latent space. More overlap leads to more unique distance values which in turn allows for easier ordering of data points. We visualise this concept using ground-truth and perceived distance matrices in Figure 3.9. In other words, as the grid spacing decreases to allow the boxes to overlap spatially to a greater and greater degree, the perceived distance as seen by the VAE through the reconstruction loss (bottom row of Figure 3.9) becomes more and more similar to the ground truth factor distances (top row of Figure 3.9). The more similar these distances are, the more likely it is that the VAE will learn the ground-truth factors of variation.

We verify our statements through the experimental results in Figure 3.10, where the β-VAE and Ada-GVAE are trained on these datasets. As the spacing decreases and overlap

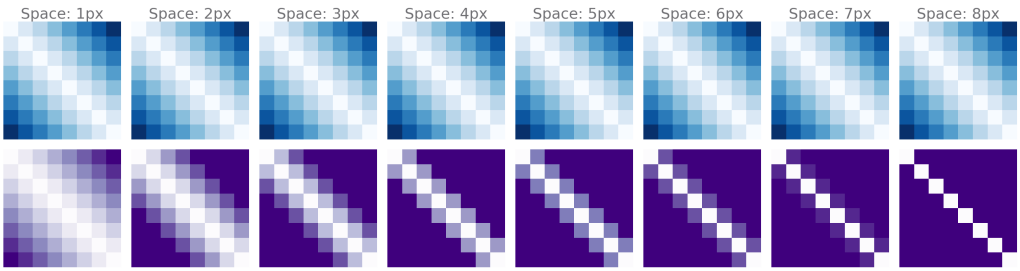


Figure 3.9: Top Row: Manhattan ground-truth distance matrices over factor traversals. Bottom Row: Pixel-wise perceived distance matrices between observations over factor traversals. Left to Right: The spacing between grid-points of XYSquares decreases from 8px to 1px introducing more overlap into the data space. More overlap leads to higher probability of being able to induce an ordering along a factor traversal. In XYSquares, each ground-truth factor has the same characteristics.

is introduced, the disentanglement performance improves, since it is easier for a VAE to introduce an ordering over representations. Even for the XYSquares dataset with 1 pixel of overlap between grid points, an ordering of elements along factor traversals can be induced. However, the probability of a VAE encountering these scenarios in the latent space due to random sampling is low, and thus it is still not always easy for the model to learn disentangled representations over such a dataset.

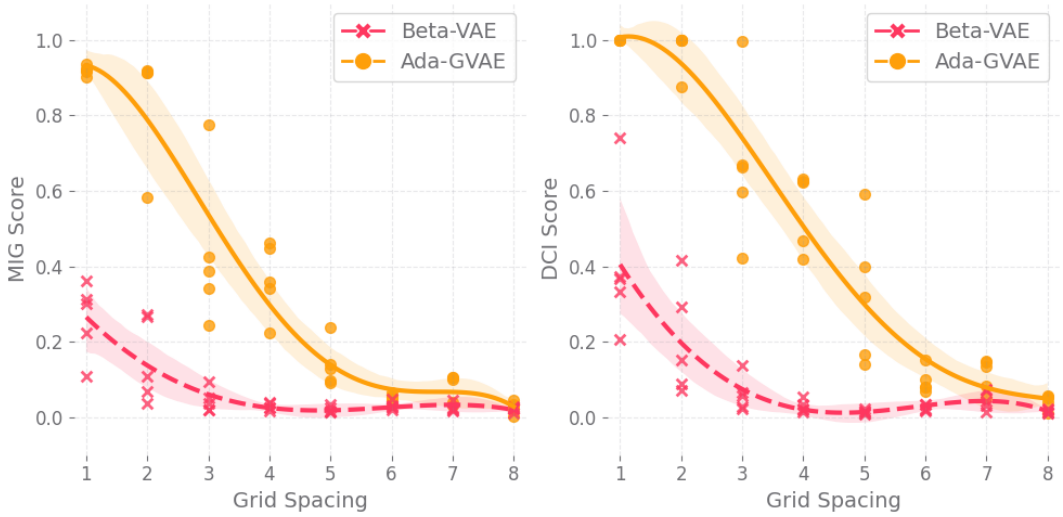


Figure 3.10: Regression plots over increasing spacing of the XYSquares dataset versus MIG score (Left) and DCI Disentanglement score (Right). As the weakly-supervised Ada-GVAE (solid lines) and β-VAE (dashed lines) are trained with these decreasing (left to right) levels of overlap, their disentanglement performance worsens. Each experiment is repeated 5 times with previously tuned hyper-parameters. See Appendix A for further experimental details.

3.4 Introducing Overlap

The previous section focused on increasing overlap by changing the underlying dataset; however, this still does not solve the case for the original constant-overlap XYSquares dataset. Throughout this chapter, we have provided evidence that VAEs disentangle based on their reconstruction loss. To solve this problem, we need to carefully choose a loss function that introduces appropriate overlap such that when perceived distances are measured, they also correspond to ground-truth distances.

The new loss function we choose cannot be a pixel-wise approach, as this does not capture the distances due to the spatial nature of the XYSquares dataset. For the sake of simplicity, we convert the existing pixel-wise loss function into a spatially aware loss function by introducing a differentiable augmentation to its inputs. An appropriate augmentation for the inputs to the reconstruction loss is a channel-wise box blur. This blur would improve the correspondence between ground-truth and perceived distances for the XYSquares dataset. The problem, however, is that the decoder needs to be able to reconstruct the data, and so purely replacing the pixel-wise loss with the augmented spatially-aware version may not succeed. Rather, in Equation 3.8, we append the augmented term to the existing loss and scale it by a constant $\alpha > 0$.

$$\mathcal{L}_{\text{Overlap}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \mathcal{L}_{\text{rec}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \alpha \, \mathcal{L}_{\text{rec}}(\text{blur}(\boldsymbol{x}), \, \text{blur}(\hat{\boldsymbol{x}})) \tag{3.8}$$

3.4.1 Augmented Loss Experiments

The loss function we choose uses a channel-wise box blur with a radius of 31, for a total kernel size of 63×63. We implement large filters efficiently using the Fast Fourier Transform. The size of the filter ensures that if two observations have active pixels on opposite sides of the images, then overlap will still be introduced between them. We set $\alpha = 63^2$. While this value appears large, a box blur kernel is normalised so that the sum of all its values is 1. We accordingly update our perceived distance measure and evaluate the distances within the XYSquares dataset for each factor in Figure 3.11 by training tuned β-VAEs. It is interesting to note that in the case of the pixel-wise loss over the adversarial dataset, a small subset of observations are placed closer together in the latent space. We hypothesise this is due to learning biases introduced by the neural networks themselves, since this is not captured by the various loss functions. We leave an investigation of this to future work.

The result of Figure 3.11 is that the VAE with the modified reconstruction loss learns distances that resemble the ground-truth factors of variation. Simply changing the reconstruction loss has introduced perceived overlap that mimics the ground-truth distances themselves and thus allows the VAE to learn these distances. In Figure 3.12, we validate that once again learning these distances using the spatially-aware loss function leads to improved disentanglement scores compared to the original pixel-wise loss. Our new loss significantly improves the disentanglement performance over the adversarial

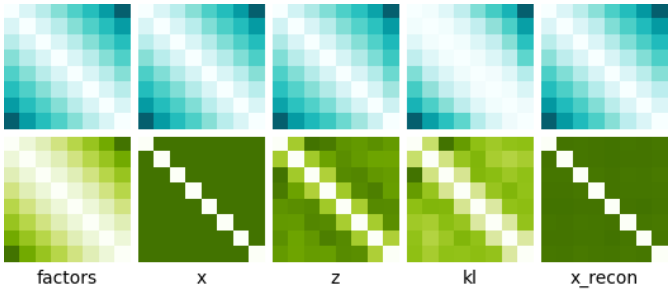


Figure 3.11: Similar to Figure 3.5, β-VAEs learn similar distances between observations at each level of the network depending on the reconstruction loss. This suggests that VAEs are approximate distance learners with respect to the reconstruction loss. Top row: box blur augmented MSE. Bottom row: pixel-wise MSE loss. Columns: Different sources of distance matrices computed over factor traversals within a VAE (Left to right: ground-truth distances, perceived distances between observations, ℓ_2 distances over latent distribution means, KL divergences between latent distributions, perceived distances between reconstructions).

dataset. This is because it allows our models to capture perceived distances between observations that once again align with the ground-truth factors.

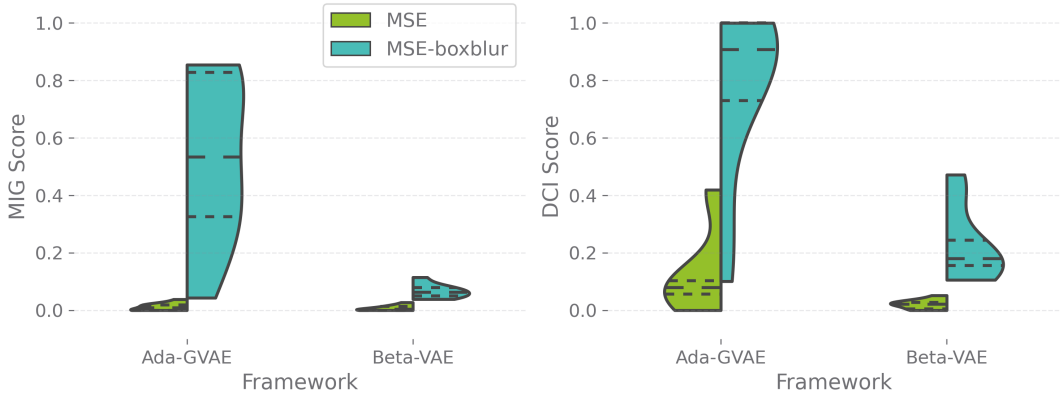


Figure 3.12: MIG and DCI scores for Ada-GVAE and β-VAE using the MSE loss and our modified loss function. Introducing a spatially aware loss function allows us to capture ground-truth distances between observations and allows the models to disentangle the adversarial XYSquares dataset.

While our choice of loss may not be optimal for disentanglement of these specific x and y factors from our adversarial dataset, disentanglement results are impressive. This is important, because it provides the intuition that changing the loss function can affect the ability of VAE frameworks to learn disentangled representations.

Now that we have built intuition, we leave the identification and learning of optimal loss functions for targeted disentanglement for Chapter 6. Before we cover these approaches, we first need to build an understanding of how to measure disentanglement

in our setting, as well as how we can guide a model using supervision to learn ground-truth distances.

3.5 Identifying Factor Importance

With examples of how the reconstruction loss affects the representations learnt by VAEs, we relate our work in this section to that of Burgess *et al.* [2017]. A factor in a dataset is considered more important if a VAE prefers to learn it before another factor. Burgess *et al.* [2017] identify this order of importance through a slow increase of the information capacity of VAEs during training. We note that simply by looking at the average perceived distance between observations along factor traversals, this ordering can be determined. Factors with a greater average distance will minimise the error in the reconstruction loss due to random sampling the most when learnt first. These factors, or components thereof, will thus generally be preferred.

To compute the average perceived distance along a factor f, we sample a ground-truth coordinate vector $\boldsymbol{y}^{(a)} \in \mathcal{Y}$ and then another random different coordinate vector $\boldsymbol{y}^{(b)} \in \mathcal{Y}^{(a,i)}$ over the traversal for factor f passing through $\boldsymbol{y}^{(a)}$. Note that $\boldsymbol{y}^{(a)} \neq \boldsymbol{y}^{(b)}$. Then, we compute the perceived distance between the corresponding observations $\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)})$. We repeat this process to compute the expected perceived distance along factor traversals, given by Equation 3.9.

$$d_i = \mathbb{E}_{a \in \mathcal{Y},\, b \in \mathcal{Y}^{(a,i)},\, a \neq b} \left[\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \right] \tag{3.9}$$

We determine the factor importance for dSprites as: $d_{\mathrm{x}} \approx 0.058$ and $d_{\mathrm{y}} \approx 0.057$ position, then $d_{\mathrm{scale}} \approx 0.025$, then $d_{\mathrm{shape}} \approx 0.022$, and finally $d_{\mathrm{orientation}} \approx 0.017$. This aligns with the order determined by Burgess *et al.* [2017]. Computing estimates over an entire dataset can be intractable, instead, we sample at least 50000 pairs per factor. Additionally, we compute the average perceived distance between any random pairs in the datasets (see Equation 3.10). For dSprites specifically, we have $d_{\mathrm{ran}} \approx 0.075$ This suggests that the ground-truth factors correspond to axes in the data that minimise the reconstruction loss and is further evidence as to why VAEs appear to learn disentangled results.

$$d_{\mathrm{ran}} = \mathbb{E}_{a \in \mathcal{Y},\, b \in \mathcal{Y},\, a \neq b} \left[\mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \right] \tag{3.10}$$

Finally, we compute and list the order of importance of factors for the remaining datasets in Table 3.1. Furthermore, we can visualise the distribution of distances along factor traversals and thus the importance of individual factors using cumulative frequency plots as in Figure 3.13. It is interesting to note the distinct shift in features for the adversarial dataset; there are no intermediate overlapping values, the cumulative proportion of values distinctly plateaus.

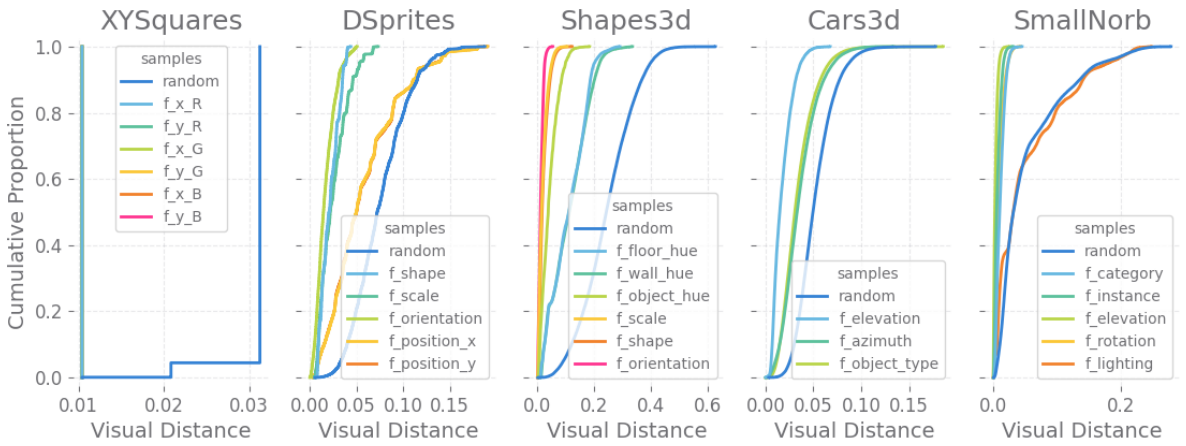


Figure 3.13: Graphs of the cumulative proportion of perceived distance values between pairs of observations sampled along factor traversals. The cumulative proportion of perceived distances between randomly sampled pairs of observations is also given. Factors which are more important for the VAE to learn first to minimise the reconstruction loss have higher average perceived distances (lines shifted further to the right). This corresponds to the experimental results from Burgess *et al.* [2017] which show that as the information capacity of a VAE is increased, it learns factors in order. For dSprites, this is x and y position, followed by scale, then shape, and finally orientation.

3.6 Summary

In Chapter 3 we define *perceived overlap* and its inverse *perceived distance* in Section 3.2.2 and Section 3.2.3, both measured in terms of the chosen reconstruction loss between data points. Our results suggest that VAEs learn distances in the representation space which correlate to these perceived distances in the data space according to the reconstruction loss. It is by chance that datasets are constructed in a way that these distances correlate to ground-truth distances (see Section 3.2.4) which is why learnt representations may then appear disentangled if they are sufficiently axis-aligned and thus appear factored.

We also show that if these distances that VAEs learn are not useful. For example, if there is constant perceived distance between data points in Section 3.3 or if there is little correlation between perceived distances and ground-truth distances, then there is little chance of learning useful disentangled representations. We then remedy this fact for the adversarial dataset in Section 3.4 by modifying the distances learnt by the VAE by changing the loss function. We give an example of a hand-crafted reconstruction loss that better captures the ground-truth factors by changing the perceived distances between data points; this allows the VAE to once again learn disentangled representations.

It should be noted that there are infinitely many datasets, each with different notions of

Dataset	Factor	Mean Dist.	Dist. Std.
Cars3D [Reed *et al.* 2015]	**random**	0.0519	0.0188
	azimuth	0.0355	0.0185
	object type	0.0349	0.0176
	elevation	0.0174	0.0100
3D Shapes [Burgess and Kim 2018]	**random**	0.2432	0.0918
	wall hue	0.1122	0.0661
	floor hue	0.1086	0.0623
	object hue	0.0416	0.0292
	shape	0.0207	0.0161
	scale	0.0182	0.0153
	orientation	0.0116	0.0079
Small NORB [LeCun *et al.* 2004]	**random**	0.0535	0.0529
	lighting	0.0531	0.0563
	category	0.0113	0.0066
	rotation	0.0090	0.0071
	instance	0.0068	0.0048
	elevation	0.0034	0.0030
dSprites [Matthey *et al.* 2017]	**random**	0.0754	0.0289
	position y	0.0584	0.0378
	position x	0.0559	0.0363
	scale	0.0250	0.0148
	shape	0.0214	0.0095
	orientation	0.0172	0.0106
XYSquares	**random**	0.0308	0.0022
	all: x & y	0.0104	0.0000

Table 3.1: Average perceived distances sampled between observation pairs taken along random factor traversals. Factors are sorted in order of importance. Factors listed first have higher perceived distances on average and should be prioritised by the model. For comparison, the average perceived distance between any random pair in the dataset is also given. The average perceived distances between pairs along factor traversals are usually less than the random distance, indicating that the ground-truth factors correspond to these axes in the data. Averages are taken over 50000 random pairs.

what constitutes disentangled representations. Researchers may also require different notions of disentanglement for different downstream tasks. For example, colours can be represented in both the RGB and HSV colour spaces. Both representations of colour can be considered disentangled, yet the required approach is ultimately subjective.

Chapter 4

Disentanglement Metrics

In Chapter 3 we showed that VAEs appear to learn latent distances over the data that correspond to their reconstruction loss. We argue that because VAEs usually learn rotationally invariant representations that mimic these distances and if these representations are then suitably axis-aligned, then the representations might accidentally appear factored and thus disentangled (see the example in Figure 3.2).

If having factored latent units is the disentanglement goal, then the result of this argument is that there are three conditions that need to be satisfied before a VAE is considered to have learnt suitably disentangled representations:

1. **Representation Distances Should Correlate To Ground-Truth Distances**: VAEs usually learn latent distances that correspond to perceived distances (see Section 3.2.2). Typical datasets are often accidentally constructed such that the distances between their ground-truth factors correlate to the perceived distances between observations in the dataset. This correlation implies that there is an upper bound on how disentangled the typically learnt representations can be.
2. **Ground-Truth Factors Need to Be Encoded Linearly**: Since VAEs typically learn representations that are rotationally invariant, before factored representations can be learnt, the VAE needs to learn representations that are linear along an arbitrary basis or rotation of the latent space for each ground-truth factor.
3. **Encodings Need to Be Axis-Aligned**: If VAEs have learnt the above two conditions, then the final condition for the representation to be considered sufficiently factored is that ground-truth factors should be captured by individual latent units. This means that ground-truth factors no longer correspond to arbitrary bases in the latent space, but rather the standard basis (see Figure 3.2).

In this chapter we introduce three new disentanglement metrics to quantify each of the above conditions; the *Ground-Truth Correlation* in Section 4.1.1, the *Linearity Ratio* in Section 4.1.3, and finally the *Axis Ratio* in Section 4.1.2. In Section 4.2 we evaluate the ground-truth correlation of distances within existing datasets, extending Section 3.2.4.

4.1 Ground-Truth Factored Component Scores

In this section we outline the three new disentanglement metrics described at the start of this chapter that make up the *Ground-Truth Factored Components*: The *Ground-Truth Correlation*, the *Linearity Ratio*, and finally the *Axis Ratio*.

4.1.1 Ground-Truth Distance Correlation

The *Ground-Truth Distance Correlation* measures how well latent distances between observations in the dataset correlate to distances between their corresponding ground-truth factors. We choose the *Spearman Rank Correlation Coefficient* to measure the monotonic relationship between these distances. We present two versions of the metric, one aggregated over samples from ground-truth factor traversals for each factor in the dataset, and one computed between distances of global samples from the dataset.

To compute the *global ground-truth distance correlation* we simply compute the rank correlation over the distances between all possible pairs of observations within the dataset and their corresponding ground-truth distances. In practice this is intractable, so we sample M random pairs of observations from the dataset $\ddot{\mathcal{Y}}_{\text{global}} \subset \mathcal{Y} \times \mathcal{Y}$, such that $|\ddot{\mathcal{Y}}_{\text{global}}| = M$, with corresponding encodings $\ddot{\mathcal{Z}}_{\text{global}} \subset \mathcal{Z} \times \mathcal{Z}$ and observations $\ddot{\mathcal{X}}_{\text{global}} \subset \mathcal{Z} \times \mathcal{Z}$. Note that we use $\ell_1(\ddot{\mathcal{Y}}_{\text{global}}) \in \mathbb{R}^M$ as the overloaded notation that produces the vector of pairwise distances between each corresponding element in $\ddot{\mathcal{Y}}_{\text{global}}$.[1] We give the equation for the *global ground-truth distance correlation* in Equation 4.1.

$$s_{\text{rcorr-global}} = \rho_{\text{Spearman}}(\mathrm{d}_{\text{gt}}(\ddot{\mathcal{Y}}_{\text{global}}),\ \ell_1(\ddot{\mathcal{Z}}_{\text{global}})) \tag{4.1}$$

To compute the *factor ground-truth distance correlation* we instead compute the aggregated Spearman rank correlation for each ground-truth factor $i \in [\mathrm{F}]$ where random pairs are sampled from random ground-truth factor traversals along that factor. This factor correlation is a more localised measure than the global correlation. We require both metrics to understand how distances have been learnt within an embedding space.

To compute this score, we use the same definition of ground-truth factor traversals as in Equation 3.6 from Section 3.2.4. For any factor $i \in [\mathrm{F}]$ generating the dataset, we sample an ordered factor traversal $\mathcal{Y}^{(a,i)} = (\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(f_i)})$ of size f_i from some randomly sampled origin $\boldsymbol{y}^{(a)} \in \mathcal{Y}$, and encode the corresponding observations $\mathcal{X}^{(a,i)} = (\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(f_i)})$ to obtain the ordered set of representations $\mathcal{Z}^{(a,i)} = (\boldsymbol{z}^{(1)}, ..., \boldsymbol{z}^{(f_i)})$. We then sample random pairs $\ddot{\mathcal{Y}}_{\text{factor},i} \subset \left\{(\boldsymbol{y}^{(u)}, \boldsymbol{y}^{(u)}) \in \mathcal{Y}^{(a,i)} \times \mathcal{Y}^{(a,i)} | \boldsymbol{y}^{(a)} \in \mathcal{Y}\right\}$ such that both observations belong to the same factor traversal. It is often intractable to compute the score over all pairs and so we sample $|\ddot{\mathcal{Y}}_{\text{factor},i}| = M$ pairs to compute the correlation for each factor. The observations $\ddot{\mathcal{X}}_{\text{factor},i}$ corresponding to factor pairs $\ddot{\mathcal{Y}}_{\text{factor},i}$ are encoded to obtain representation pairs $\ddot{\mathcal{Z}}_{\text{factor},i}$. We give the equation for each *individual factor ground-truth distance correlation* in Equation 4.2.

$$s_{\text{rcorr-factor},i} = \rho_{\text{Spearman}}(\ell_1(\ddot{\mathcal{Y}}_{\text{factor},i}),\ \ell_1(\ddot{\mathcal{Z}}_{\text{factor},i})) \tag{4.2}$$

[1] For some distance function $d(\cdot,\cdot)$, if $\ddot{\mathcal{A}}$ is an ordered set of pairs of vectors $\ddot{\mathcal{A}} \subset \mathcal{A} \times \mathcal{A}$ of size $|\ddot{\mathcal{A}}| = n$, then $d(\mathcal{A})$ is the overloaded notation that produces the vector of pairwise distances between each corresponding element, such that: $d(\ddot{\mathcal{A}}) = \left(d(\ddot{\mathcal{A}}_{1,1}, \ddot{\mathcal{A}}_{1,2}), \ldots, d(\ddot{\mathcal{A}}_{n,1}, \ddot{\mathcal{A}}_{n,2})\right)$

Finally, to produce the combined *factor ground-truth distance correlation* $s_{\text{rcorr-factors}}$ in Equation 4.3, we aggregate the individual scores for each factor $s_{\text{rcorr-factor},i}$ $\forall i \in [\text{F}]$ together using the geometric mean. We choose the geometric mean because it weights the values towards zero, and thus it is more difficult to obtain higher scores if there is one lower score.

$$s_{\text{rcorr-factors}} = \left(\prod_{i=1}^{\text{F}} s_{\text{rcorr-factor},i} \right)^{\frac{1}{\text{F}}} \tag{4.3}$$

4.1.2 Axis-Alignment Ratio

The *latent axis linearity ratio*, or rather the *axis-alignment ratio*, measures how well a ground truth factor traversal is explained by a single latent variable or standard basis vector. In other words, how well all embeddings of a factor traversal lie on a single D-dimensional line that is parallel to a latent variable. We provide a visual example in Figure 4.1.

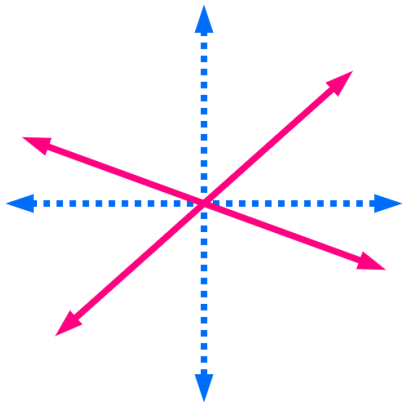


Figure 4.1: Example of *axis-aligned* representations. The solid lines represent some arbitrary linear rotation of the latent space, where each axis corresponds to a ground-truth factor. The dotted lines represent a similar rotation of the latent space that is instead axis-aligned such that individual latent units encode factors which correspond to the visualised axes. This property is measured by the *axis-alignment ratio*, the dotted axis will achieve a score close to 1, while the rotated and solid axis will achieve a low score towards 0.

For each ground-truth factor traversal we use the corresponding representations $\forall\, \boldsymbol{z} \in \mathcal{Z}^{(a,i)}$ to compute an estimate for the variance of each latent variable z_j $\forall j \in [\text{D}]$. We compute the score in Equation 4.4 as the maximum variance estimate corresponding to latent unit $k \in [\text{D}]$ over the sum of all variance estimates for each latent unit. As this value approaches 1, all variance of latent units over the factor traversal is explained by the single latent variable with maximum variance and no others.

$$\tilde{s}_{\text{axis},i} = \mathbb{E}_{\mathcal{Z}^{(a,i)} \; \forall \; \boldsymbol{y}^{(a)} \in \mathcal{Y}} \left[\max_{k \in [\mathrm{D}]} \left(\frac{\mathrm{Var}[z_k]}{\sum_{j=1}^{\mathrm{D}} \mathrm{Var}[z_j]} \right) \right] \tag{4.4}$$

The problem with Equation 4.4 is that the function produces scores in the range $\left[\frac{1}{\mathrm{D}}, 1\right]$. Since each variance value is positive and assuming that at least one variance value is non-zero, then this can easily be seen by decomposing $\tilde{s}_{\text{axis},i}$ into the components $\frac{a_{\texttt{max}}}{a_{\texttt{max}}+a_{\texttt{other}}}$, where $\texttt{max} = \text{argmax}_{k \in [\mathrm{D}]} \mathrm{Var}[z_k]$ and $a_{\texttt{other}} = \sum_{j \in [\mathrm{D}]/\{\texttt{max}\}} \mathrm{Var}[z_j]$. The value of $\tilde{s}_{\text{axis},i} = 1$ is at a maximum if $a_{\texttt{other}} = 0$. Similarly, $\tilde{s}_{\text{axis},i} = \frac{1}{\mathrm{D}}$ is at a minimum if $a_{\texttt{other}} = (\mathrm{D}-1) \cdot a_{\texttt{max}}$.

We seek to normalise these values such that the score is in the range $[0, 1]$, where 1 represents a perfect disentanglement score and 0 represents the worst disentanglement score or alignment of representations with a single axis. This is done by standardising the results using the number of values D as seen in Equation 4.5:

$$s_{\text{axis},i} = \frac{\tilde{s}_{\text{axis},i} - \frac{1}{\mathrm{D}}}{1 - \frac{1}{\mathrm{D}}} \tag{4.5}$$

The average ratio for each factor becomes the geometric mean over the expected value for each factor i. We approximate the expected value by sampling a fixed number of factor traversals for each factor i. The average score is given by Equation 4.6. We choose the geometric mean because it weights the values towards zero, if one factor obtains a lower score, then it is more difficult to obtain a combined higher score.

$$s_{\text{axis}} = \left(\prod_{i=1}^{\mathrm{F}} s_{\text{axis},i} \right)^{\frac{1}{\mathrm{F}}} \tag{4.6}$$

We provide the *normalised axis-orientation ratio* purely for interest in Appendix A.1. This metric may have interesting properties to researchers as it is a normalised version of the axis-alignment ratio that does not depend on the linearity of representations as described in the next section.

4.1.3 Linearity Ratio

The *arbitrary axis linearity ratio*, or rather the *linearity ratio*, measures how well a ground truth factor traversal is explained by the most important orthogonal basis vector. Alternatively, how well all embeddings of a factor traversal lie on an arbitrary single d-dimensional line of best fit. We give an example of this in Figure 4.2.

Instead of using the standard basis as with the *latent axis linearity ratio*, we find a new orthogonal basis for the embeddings $\mathcal{Z}^{(a,i)}$ of a factor traversal such that the most variance of the data is captured by the first basis vector. The variances over the coefficients

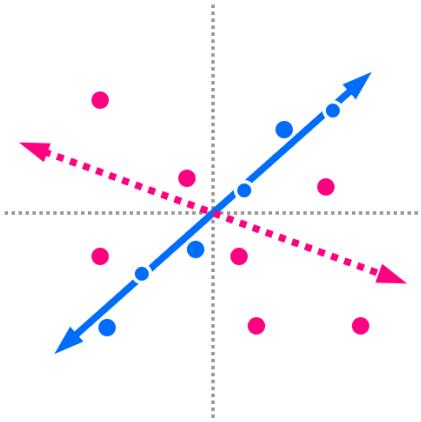


Figure 4.2: Example of *linear* representations. Each line with arrows corresponds to the representations of a single ground-truth factor traversal that have been encoded. Embeddings that lie along the dotted line do correlate well with this line and obtain low scores for the linearity ratio. Embeddings that lie along the solid line correspond well, most of the variance is explained by this single principle component. These embeddings thus obtain a high linearity ratio.

for the new basis are used to calculate the ratio in the same way as the *latent axis linearity ratio*. As the ratio approaches 1, all the variance is described by the most important basis vector and no others.

In practice we use Principle Component Analysis (PCA) [Pearson 1901; Hotelling 1933] over ground-truth traversal embeddings $\mathcal{Z}^{(a,i)}$ to obtain the orthogonal principle components $\boldsymbol{W} \in \mathbb{R}^{\mathrm{D}\times\mathrm{D}}$ and corresponding explained variance of the data $\boldsymbol{\lambda} \in \mathbb{R}^{\mathrm{D}}$. λ_1 explains the most variance while λ_{D} usually explains the least and has the lowest value. We discard the principle components and use the explained variance to approximate the *arbitrary axis linearity ratio* for a factor traversal using Equation 4.7.

$$\tilde{s}_{\text{linear},i} = \mathbb{E}_{\mathcal{Z}^{(a,i)} \,\forall\, \boldsymbol{y}^{(a)}\in\mathcal{Y}} \left[\max_{k\in[\mathrm{D}]} \left(\frac{\lambda_k}{\sum_{j=1}^{\mathrm{D}} \lambda_j} \right) \right] \tag{4.7}$$

With similar reasoning as Equation 4.5 in the previous section, we standardise Equation 4.7 such that it has a new range of $[0, 1]$. We give this equation in Equation 4.8:

$$s_{\text{linear},i} = \frac{\tilde{s}_{\text{linear},i} - \frac{1}{\mathrm{D}}}{1 - \frac{1}{\mathrm{D}}}. \tag{4.8}$$

The average ratio, given by Equation 4.9, is computed in the same way as for the *latent axis linearity ratio*.

$$s_{\text{linear}} = \left(\prod_{i=1}^{\text{F}} s_{\text{linear},i} \right)^{\frac{1}{\text{F}}} \tag{4.9}$$

It should be noted that we purposely design the linearity and axis-alignment ratios such that they depend on the number of latent units. The larger the representations, typically the more difficult it is for a VAE to achieve high scores on these metrics. We can modify the metric such that it is scale-independent by adopting a similar approach to the *Mutual Information Gap* [Chen *et al.* 2018] which is computed over the two largest components instead of the sum of all components, however, doing so may result in better scores that do not match further visual inspection.

4.1.4 Evaluation

We validate that if points are more closely represented by an arbitrarily rotated line, then the linearity ratio Equation 4.9 improves. Similarly, when this line is more closely aligned with an axis, then the axis-alignment ratio Equation 4.6 improves. We provide visual examples of these cases for our metrics in Figure 4.3, where we compute the proposed axis-alignment and linearity ratios for differing rotations and distributions of points. We note that the axis-alignment ratio is bounded above by the linearity score and is thus also an indirect measure of linearity. For a measure of rotation without considering the variance, we define the axis orientation score in Equation A.1 from Appendix A.1.

4.2 Perceived Distances Over Data Correlate To Ground-Truth

At the start of this chapter we outline the three components to learning disentangled representations. In Item 1 we allude to the fact that typical datasets are often accidentally constructed such that the distances between their ground-truth factors correlate to the perceived distances between observations in the dataset. Since VAEs are prone to learning perceived distances, this correlation implies that there is an upper bound on how disentangled typically learnt representations can be. If there is greater correlation between the perceived and ground-truth distances, then VAEs will usually learn representations where the latent and ground-truth distances are better correlated too.

Similar to how we compute the *ground-truth distance correlation* in Equation 4.1 of Section 4.1.1, we use a similar construction to instead compute the correlation between ground-truth distances and perceived distances between observations in terms of the reconstruction loss. We additionally compute the *Pearson Correlation Coefficient* alongside the *Spearman Rank Correlation Coefficient*.

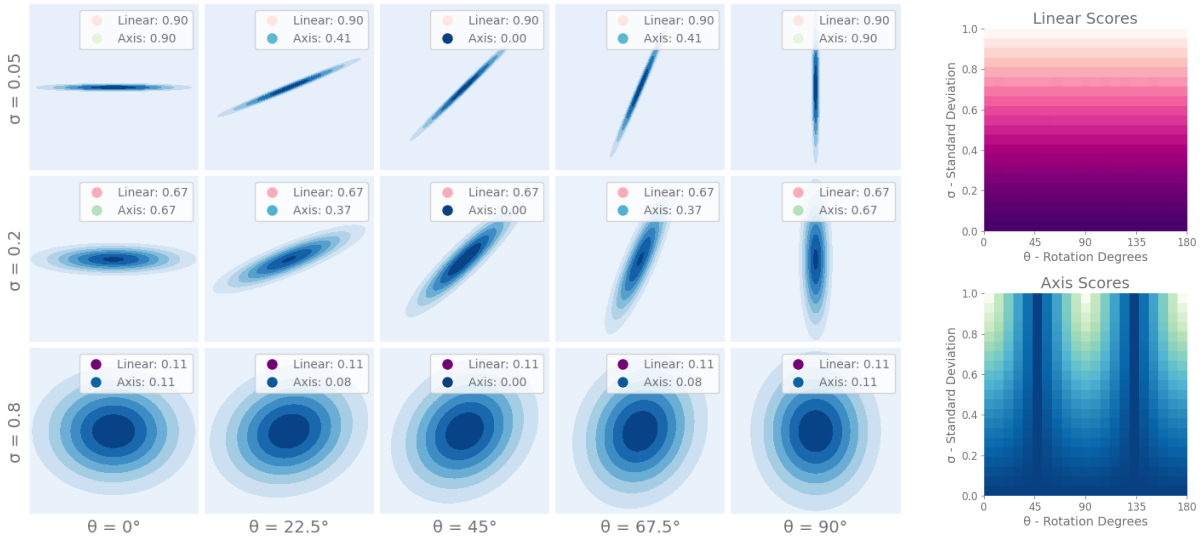


Figure 4.3: Visualisation of the *Axis-Alignment Ratio* and *Linearity Ratio* from Sections 4.1.2 and 4.1.3 in a 2 dimensional setting. Gaussian distributions of varying rotation (Left to Right) and width (Top to Bottom) are pictured. The linearity ratio is independent of the rotation and improves as points are more closely represented by a single line or single arbitrary basis. The axis-alignment ratio depends on the rotation, improving as points lie along a single axis and obtaining the worse score at $45\,\mathrm{deg}$. In the case where representations are axis-aligned, linearity and axis-alignment scores will be equal $s_{\text{linear}} = s_{\text{axis}}$.

We present our results in Table 4.1 for the common disentanglement datasets analysed in Section 3.2.4, as well as results in Table 4.2 for the adversarial XYSquares datasets with varying overlap from Section 3.3.4. We compute the different correlation values for perceived distances in terms of both the pixel-wise mean squared error (MSE), and the spatially aware version of MSE augmented with a box-blur as described in Section 3.4 intended to solve the adversarial XYSquares dataset.[2]

The results in Table 4.1 show that on average distances between data points in existing datasets are highly correlated with ground-truth distances. We observe that some datasets have higher correlation with ground-truth distances than other datasets; these datasets with higher correlations usually obtain better disentanglement scores when used to train VAEs as seen in Chapter 3. Furthermore, these results suggest that some ground-truth factors may be too large or unintuitive, because their correlation scores are very low. We specifically note the very low scores for the "object type" and "azimuth" factors from Cars 3D, the "orientation" factor from dSprites, as well as the "rotation" and "lighting" factors from Small NORB. The 3D Shapes dataset usually has very high correlation with ground-truth factors. This further emphasises the unusually high

[2]The example loss function is domain specific; loss functions should be designed per domain to improve ground-truth and perceived distance correlation to improve disentanglement.

disentanglement scores for this dataset, even in the unsupervised case (see Chapter 3).

The results in Table 4.2 verify that increasing the overlap in the XYSquares dataset by reducing the spacing between squares, drastically improves the correlation between ground-truth distances and distances over the data according the reconstruction loss. These results correspond well to that of the performance of the trained models over these datasets from Section 3.3.4. These results also show that instead using the hand-crafted loss from Section 3.4.1 also introduces overlap within the data by improving these correlations.

Dataset	Factor Name	Linear Corr. (MSE)	Rank Corr. (MSE)	Linear Corr. (MSE-box)	Rank Corr. (MSE-box)
Cars3D [Reed *et al.* 2015]	**random**	0.15	0.13	0.04	0.10
	elevation	0.90	0.93	0.69	0.88
	azimuth	0.30	0.34	0.08	0.25
	object type	0.04	0.04	0.00	0.01
3D Shapes [Burgess and Kim 2018]	**random**	0.53	0.45	0.29	0.29
	floor hue	0.62	0.76	0.60	0.74
	wall hue	0.60	0.74	0.55	0.72
	object hue	0.53	0.71	0.41	0.63
	scale	0.81	0.88	0.71	0.87
	shape	0.69	0.79	0.58	0.80
	orientation	0.84	0.92	0.74	0.87
Small NORB [LeCun *et al.* 2004]	**random**	0.10	0.14	0.07	0.14
	category	0.44	0.53	0.15	0.47
	instance	0.37	0.52	0.10	0.51
	elevation	0.81	0.90	0.51	0.64
	rotation	0.12	0.19	0.07	0.21
	lighting	0.07	0.29	0.07	0.28
dSprites [Matthey *et al.* 2017]	**random**	0.43	0.38	0.29	0.36
	shape	0.66	0.72	0.66	0.87
	scale	0.93	0.95	0.84	0.96
	orientation	0.13	0.17	0.15	0.21
	position x	0.66	0.75	0.63	0.83
	position y	0.65	0.75	0.63	0.83

Table 4.1: Average correlation between ground-truth distances and perceived distances (MSE) within the common datasets. It is interesting to compare these values to the visual plots of distances along ground truth factors in Figure 3.4. Values are computed over 1048576 random pairs, either sampled along factor traversals or sampled from the entire dataset in the random case. Swapping the MSE loss for the hand-crafted augmented loss *MSE-box* blur, as in Section 3.4.1, changes the perceived distances. This change does not always improve ground-truth distance correlation for all datasets, rather this loss is an example that is specific to XYSquares.

Dataset	Factor Name	Linear Corr. (MSE)	Rank Corr. (MSE)	Linear Corr. (MSE-box)	Rank Corr. (MSE-box)
XYSquares (1px)	all: x & y	1.00	1.00	0.98	0.99
XYSquares (2px)	all: x & y	0.94	0.99	0.99	0.99
XYSquares (3px)	all: x & y	0.86	0.95	0.99	0.99
XYSquares (4px)	all: x & y	0.75	0.85	0.99	0.99
XYSquares (5px)	all: x & y	0.72	0.85	0.99	0.99
XYSquares (6px)	all: x & y	0.67	0.85	0.98	0.98
XYSquares (7px)	all: x & y	0.60	0.85	0.97	0.98
XYSquares	all: x & y	0.52	0.58	0.96	0.97

Table 4.2: Average correlation between ground-truth distances and perceived distances within the adversarial XYSquares datasets with varying levels of overlap. The easiest version of the dataset to disentangle has a spacing of 1px between grid points, the hardest version to disentangle has constant overlap with a spacing of 8px between grid points. Swapping the MSE loss for the hand-crafted augmented loss, as in Section 3.4.1, changes the perceived distances and improves the ground-truth distance correlation.

4.3 VAE Frameworks Learn Perceived Distances

In Figure 3.5 from Section 3.2.4 we visually confirm the argument from Section 3.1 that VAEs reorganise the embedding space to mimic distances in the data space due to the interaction between random sampling and the regularisation term during training. This reorganisation of the embedding space locally minimises error according to the reconstruction loss due to random sampling.

We corroborate these results by training VAE frameworks on existing datasets and measuring the Spearman rank correlation between ground-truth distances and latent distances sampled over factor traversals as given by Equation 4.2. Additionally, we also compute the Spearman rank correlation between latent distances and perceived distances according to the reconstruction loss. We sample observations in the same way as for Equation 4.2, and aggregate the individual factor correlation scores from Equation 4.10 in the same way as Equation 4.3 to obtain $s_{\text{rcorr-data}}$. Note that we use $\mathrm{d}_{\text{pcv}}(\mathcal{X}_{\text{factor},i}) \in \mathbb{R}^M$ as the overloaded notation that produces the vector of pairwise distances between each corresponding element of $\mathrm{d}_{\text{pcv}}(\boldsymbol{x}^{(u)}, \boldsymbol{x}^{(v)}) \; \forall \; (\boldsymbol{x}^{(u)}, \boldsymbol{x}^{(v)}) \in \mathcal{X}_{\text{factor},i}$.

$$s_{\text{rcorr-data},i} = \rho_{\text{Spearman}}(\mathrm{d}_{\text{pcv}}(\ddot{\mathcal{X}}_{\text{factor},i}),\ \ell_1(\ddot{\mathcal{Z}}_{\text{factor},i})) \tag{4.10}$$

The results in Figure 4.4 show that VAEs learn distances in the latent space that correlate almost perfectly with perceived distances in the data space as previously hypothesised. However, our results also align well with the analysis in the previous section on the correlation upper bound between ground-truth factors and perceived distances. For further experiment details see Appendix A. It is interesting to note in this experiment that the correlation between both ground-truth distances and perceived distances, as well as latent distances and perceived distances is almost identical between frame-

works. This shows that the special algorithmic choices of these frameworks do not modify the underlying distances learnt. The final differences in disentanglement performance between the frameworks thus cannot be attributed to differences between distances learnt in the latent space. Rather the Ada-GVAE specifically produces more axis-aligned and linear representations along individual factors as captured by the linearity and axis-alignment ratios (see Sections 4.1.2 and 4.1.3). Since distances are not modified, these frameworks are still hindered by the aforementioned upper bound that is the correlation between the ground-truth distances and perceived distances.



Figure 4.4: Scores for β-VAE and Ada-GVAE frameworks over varying datasets and values of β. The correlation between all distances is almost identical between frameworks. Furthermore, the rank correlations between perceived distances over the data and learnt distances in the latent space are almost perfect. Remaining differences in disentanglement performance are captured by the linearity and axis-alignment scores. The Ada-GVAE performs better because it explicitly encourages factorisation. Entries are missing for the XYSquares dataset because the models fail to learn with higher values of β. See Appendix A for further experiment details.

4.4 Summary

In this chapter, we outlined the three components of learning disentangled representations that correspond to ground-truth distances within the dataset. These components were derived from the goal of obtaining factored representations, where each latent unit corresponds to one ground-truth factor in the dataset.

We outlined problems with existing disentanglement metrics in capturing these factors and then we proposed new disentanglement metrics that measure each of these components separately: the *ground-truth distance correlation* (Section 4.1.1), *linearity ratio* (Section 4.1.3) and *axis-alignment ratio* (Section 4.1.2). These metrics measure the correspondence between ground-truth factors within a dataset and the latent embeddings from a model. The advantage of measuring each component separately is that we can better analyse disentanglement frameworks to find the root cause of problems with disentanglement under our conditions.

In Section 4.2 we used variants of the metrics to analyse the accidental correspondence between perceived distances in terms of the reconstruction loss and ground-truth distances. This suggested that datasets are accidentally constructed in a way that may be conducive to disentanglement in a VAE setting.

Finally, in Section 4.3, we demonstrated an upper bound on the correlation between latent distances and ground-truth distances. This is because VAEs instead learn distances in the latent space that are highly correlated with perceived distances in the data space. Since our metrics showed that standard and state-of-the-art VAE frameworks learn similar distances between observations, we thus attributed disentanglement performance in VAE frameworks to better axis-alignment of the learnt representations. Learning disentangled representations that are truly factored and correspond to the ground-truth factors, may thus be impossible.

Chapter 5

Metric Learning

In the previous chapter we introduce new metrics for measuring the types of representations learnt by machine learning models. We argue in Chapter 4 that there are three components to disentanglement when the goal is learning factored representations: ground-truth distance correlation, linearity and axis-alignment. In this chapter we seek to improve disentanglement scores across all three metrics.

The first problem is that if VAEs learn distances that correlate with perceived distances in terms of the reconstruction loss, then there is an upper bound on how well distances can correlate with ground-truth factors. We solve this problem by biasing the distances VAEs learn in a supervised manner through the integration of *metric* or *similarity* learning in Section 5.3.

Finally, with a supervised solution for ground-truth distance correlation and linearity scores of learnt representations, we then seek to introduce axis-alignment in Section 5.4. We take inspiration from the Adaptive VAE methods by Locatello *et al.* [2020]. As argued in Chapter 4, if correct distances have been learnt, upon the introduction of axis-alignment we can learn representations that appear factored and thus disentangled according to ground-truth factors.

5.1 Metric Learning Overview

While not an explicit disentanglement approach, *metric* or *similarity* learning is the task of learning a distance function between observations. The variant of metric learning we use throughout this work is the soft-margin formulation of triplet loss described in Section 5.1.2, which constrains representations of *positive* samples to be closer together than representations of *negative* samples from *anchor* samples.

5.1.1 Triplet Loss

Triplet loss makes use of three observations $\mathcal{X}_{\text{triple}} = (\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(p)}, \boldsymbol{x}^{(n)})$. These observations are sampled using supervision such that the *anchor* $\boldsymbol{x}^{(a)}$ is considered closer to the *positive* $\boldsymbol{x}^{(p)}$ than it is to the *negative* $\boldsymbol{x}^{(n)}$. The intuition is that corresponding embeddings $\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}, \boldsymbol{z}^{(n)}$ output by a neural network should satisfy the original anchor-positive and anchor-negative distance constraints given by $d(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) < d(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)})$.

Triplet loss is defined in Equation 5.1.

$$\mathcal{L}_{\text{Triplet-Hard}}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}, \boldsymbol{z}^{(n)}) = \max\left(0, \quad \psi + \mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) - \mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)})\right) \tag{5.1}$$

Minimising the triplet loss means minimising the distance between the anchor and positive $d(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)})$, which is lower-bounded by 0, and maximising the distance between the anchor and negative $d(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)})$, which has no upper bound. However, the intuition is that a negative sample should not have an increased effect beyond a certain threshold. This is reflected by the addition of the *margin* $\psi > 0$, such that if $\mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)}) > \psi + \mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)})$ then the loss becomes zero.

Care needs to be taken as the performance of triplet loss is highly dependent on how positive and negative observations are sampled. Since the loss can become zero, no learning takes place in such instances. Xuan *et al.* [2020] include a summary of triplet mining strategies that seek to avoid this problem by carefully selecting triplets such that the distances are always valid, the loss is non-zero and learning thus takes place.

5.1.2 Soft-Margin Triplet Loss

To help alleviate such problems without the introduction of triplet mining we use the soft-margin formulation of triplet loss as described by Hermans *et al.* [2017] and given in Equation 5.2. The loss never becomes zero while still following the intuition that negative samples should have a decreased effect as they are further away from the anchor. The soft-margin formulation advantageously does this without an additional margin ψ hyper-parameter that needs to be tuned.

$$\mathcal{L}_{\text{Triplet-Soft}}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}, \boldsymbol{z}^{(n)}) = \ln\left(1 + \exp\left(\mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) - \mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)})\right)\right) \tag{5.2}$$

Performance of the soft-margin formulation is generally better than that of the original hard-margin version [Hermans *et al.* 2017]. We use the soft-margin formulation when referring to triplet loss throughout the rest of this work, unless otherwise specified.

5.2 Sampling Strategies

Similarity learning approaches fall under two general categories [Wang *et al.* 2014] depending on the distinguishing capability between the learnt embeddings:

- *category-level* similarity that only distinguishes between observations that fall under different categories.
- *fine-grained* similarity that can also distinguish between observations within the same category.

The main contributor to whether a triplet learning approach is *fine-grained* or *categorical* is how the observation triplets are selected. In practice these triplets are selected using the underlying labels for each observation in a dataset. A distance function can be used to compare these labels in a supervised manner and order observations when constructing a triplet for training. We use the notation $\mathrm{s}(\cdot,\cdot)$ as a distance function used to select triplets, instead of the distance functions $d(\cdot,\cdot)$ used between embeddings within triplet loss itself.

Given three randomly sampled observations $\boldsymbol{x}^{(\tilde{a})}, \boldsymbol{x}^{(\tilde{p})}, \boldsymbol{x}^{(\tilde{n})} \in \mathcal{X}$, we generate a triplet by reordering elements to obtain $\mathcal{X}_{\text{triple}} = (\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(p)}, \boldsymbol{x}^{(n)})$ using the distances between the labels as seen in Equation 5.3. Note that it may be necessary to repeat the sampling procedure so that the anchor-positive and anchor-negative distances are never the same $\mathrm{s}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(p)}) \neq \mathrm{s}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(n)})$.

$$
\begin{aligned}
\boldsymbol{x}^{(a)} &= \boldsymbol{x}^{(\tilde{a})} \\
\boldsymbol{x}^{(p)} &= \begin{cases} \boldsymbol{x}^{(\tilde{p})} & \text{if } \mathrm{s}(\boldsymbol{x}^{(\tilde{a})}, \boldsymbol{x}^{(\tilde{p})}) \leq \mathrm{s}(\boldsymbol{x}^{(\tilde{a})}, \boldsymbol{x}^{(\tilde{n})}) \\ \boldsymbol{x}^{(\tilde{n})} & \text{otherwise} \end{cases} \\
\boldsymbol{x}^{(n)} &= \begin{cases} \boldsymbol{x}^{(\tilde{n})} & \text{if } \mathrm{s}(\boldsymbol{x}^{(\tilde{a})}, \boldsymbol{x}^{(\tilde{p})}) \leq \mathrm{s}(\boldsymbol{x}^{(\tilde{a})}, \boldsymbol{x}^{(\tilde{n})}) \\ \boldsymbol{x}^{(\tilde{p})} & \text{otherwise} \end{cases}
\end{aligned}
\tag{5.3}
$$

Triplet loss is highly dependent on the strategy used to select triples [Xuan *et al.* 2020]. In the following sections we give various sampling strategies from literature or that are appropriate for learning ground-truth distances between data points. We also comment on whether or not these strategies lead to learning fine-grained or category-level similarity.

5.2.1 Sampling Based on Categories

Typical datasets which are intended for categorical level similarity contain observations $\boldsymbol{x}^{(i)}$ labelled with one of $C > 0$ classes $c^{(i)} \in [C]$. The usual goal is to group representations belonging to the same class together. This can be done by constructing triplets such that observations in the anchor-positive pair belong to the same class, while observations in the anchor-negative pair belong to differing classes. We convert this notion to a distance function in Equation 5.5 using Iverson bracket notation.[1]

$$
\mathrm{s}_{\text{category}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = \left[c^{(a)} \neq c^{(b)}\right] \tag{5.4}
$$

$$
= \begin{cases} 1, & \text{if } c^{(a)} \neq c^{(b)} \\ 0, & \text{otherwise} \end{cases} . \tag{5.5}
$$

[1] Iverson bracket notation $[P]$ returns 1 if the term P is true, otherwise it returns 0.

One way to reintroduce fine-grained similarity in such a case is to combine categorical triplet loss with auto-encoders. Since representations need to be unique to reconstruct individual observations, the system is thus also be able to distinguish between representations from the same category.

5.2.2 Sampling Based on Number of Differing Factors

The Adaptive-VAEs from Locatello *et al.* [2020] sample pairs of observations such that k factors differ between them. This sampling strategy can be defined using a distance function that measures the number of differing ground-truth factors between observations. We write Equation 5.6 using Iverson bracket notation:

$$\mathrm{s}_{\text{f-diff}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = \sum_{i=1}^{\mathrm{F}} \left[y_i^{(a)} \neq y_i^{(b)} \right]. \tag{5.6}$$

This distance function measures *category-level* similarity. It is unable to differentiate between pairs of observations that have the same number of differing factors k.

5.2.3 Sampling Based on Ground-Truth Distance Along Factors

In Section 3.2.1 we define the ground-truth distance between observations in terms of the Manhattan or ℓ_1 distance between their ground-truth factors. This is a more intuitive way to measure distance between observations than sampling based on the number of differing factors, as the magnitude of the difference is also encoded.

$$\begin{aligned} \mathrm{s}_{\text{f-dist}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) &= \mathrm{d}_{\mathrm{gt}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) && (5.7) \\ &= \|\boldsymbol{y}^{(a)} - \boldsymbol{y}^{(b)}\|_1 && (5.8) \\ &= \sum_{i=1}^{\mathrm{F}} |y_i^{(a)} - y_i^{(b)}| && (5.9) \end{aligned}$$

Using this distance results in **fine-grained** similarity measure. We argue that such a measure is more analogous to human notions of distance between factors; furthermore, we have shown in Chapter 3 and Chapter 4 that these distances often correspond to perceived distances. We also argue that this distance measure requires less knowledge of the domain, since we do not need to know which factors have changed, we only need some notion of distance between observations. For example, this may be introduced in the form of a temporal distance between successive states in reinforcement learning, but, we leave these approximations as future work.

5.2.4 Sampling Based on Normalised Ground-Truth Distance

The problem with the sampling strategy in the previous Section 5.2.3 is that there is often scale attached to ground-truth factors. If a dataset has individual factors that differ in size or rather the number of allowed values, then more importance may be placed upon larger individual factors when sampling. To remedy this problem we can scale each individual factor i such that the distance between the first (min) and the last element (max) of that factor is equal to 1. We assume that the size of each individual factor constructing the dataset is always greater than one: $f_i > 1 \; \forall i \in [\mathrm{F}]$. The normalised distance is given in Equation 5.11. This is the main sampling procedure we use for supervised construction of triplets throughout this work, unless otherwise specified.

$$\mathrm{s}_{\text{f-dist-scaled}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = \mathrm{d}_{\text{gt-scaled}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \tag{5.10}$$

$$= \sum_{i=1}^{\mathrm{F}} \frac{1}{f_i - 1} |y_i^{(a)} - y_i^{(b)}| \tag{5.11}$$

Care should be taken when implementing this distance function, since precision errors can arise due to the summation over the normalised values. Distances that should be equal might incorrectly be considered less or greater. To remedy this issue, we use an arbitrary precision implementation of these normalised values using integer numerators and denominators. We add these fractions together such that the results remain in fractional form. In the case of the XYSquares dataset, summing over floating point values results in an error rate of approximately 1.4%. Even though this error rate is low, it still has a significant effect on disentanglement performance.

5.3 Triplet Auto-Encoders

In the previous sections we give an overview of triplet loss and supervised sampling strategies to construct training triplets based on distances between ground-truth factors within the dataset. In this section we seek to improve both the *ground-truth distance correlation* and *linearity ratios* by combining triplet loss with VAEs to bias the distances learnt between embeddings in a supervised manner. We take inspiration from the supervised Triplet VAE (TVAE) [Karaletsos *et al.* 2015; Ishfaq *et al.* 2018], fine-grained deep ranking models described by Wang *et al.* [2014] and the soft-margin formulation of triplet loss by Hermans *et al.* [2017].

5.3.1 β-TVAE

While triplet loss was originally combined with VAEs by Karaletsos *et al.* [2015] for incorporating human knowledge and known relations between observations, we use a modification of the simplified triplet VAE (TVAE) [Ishfaq *et al.* 2018].

The TVAE is a supervised approach to deep metric learning that uses a Siamese network that shares model weights between three inputs. The triplet or inputs are an ordered tuple $\mathcal{X}_{\text{triple}} = (\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(p)}, \boldsymbol{x}^{(n)})$ of observations, the anchor $\boldsymbol{x}^{(a)}$, the positive $\boldsymbol{x}^{(p)}$ and the negative $\boldsymbol{x}^{(n)}$. The loss is the same as that of the standard VAE aggregated over the three inputs; however, the loss is additionally augmented by triplet loss. The TVAE assumes that the posterior distributions are normal distributions with diagonal covariance parameterised by $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. The triplet loss is calculated over the three computed mean vectors $\boldsymbol{\mu}^{(a)}$, $\boldsymbol{\mu}^{(p)}$ and $\boldsymbol{\mu}^{(n)}$ rather than the sampled representations themselves.

Ishfaq *et al.* [2018] outline the TVAE framework, which we modify by first replacing the VAE base with the β-VAE from Section 2.3.1, secondly replacing the standard triplet loss with the soft-margin formulation from Section 5.1.2, and finally weighting the triplet loss component with $\alpha > 0$. We term this variant the β-TVAE, the augmented loss is given in Equation 5.12:

$$\mathcal{L}_{\beta\text{-TVAE}} = \frac{1}{3} \sum_{\boldsymbol{x} \in \mathcal{X}_{\text{triple}}} \mathcal{L}_{\beta\text{-VAE}}(\boldsymbol{x}) + \alpha \mathcal{L}_{\text{Triplet-Soft}}(\boldsymbol{\mu}^{(a)}, \boldsymbol{\mu}^{(p)}, \boldsymbol{\mu}^{(n)}). \tag{5.12}$$

5.3.2 ℓ_1 versus ℓ_2 Distance

The Euclidean, ℓ_2, distance function is typically used with triplet loss. However, in high dimensional spaces, Aggarwal *et al.* [2001] show that Manhattan ℓ_1 distance is preferable over Euclidean ℓ_2 distance as the discriminative ability between embeddings is better. Furthermore, we argue that ℓ_1 distance is advantageous because it better aligns with the human notion of changing factors of variation and ground-truth distance as in Section 3.2.1. Thus, for the remainder of this work, unless otherwise specified, we use the ℓ_1 distance function within triplet loss for computing the distance between representations, such that $\mathrm{d}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) = \mathrm{d}_{\ell_1}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) = \|\boldsymbol{z}^{(a)} - \boldsymbol{z}^{(p)}\|_1$.

5.3.3 Experimental Results

The goal of combining triplet loss with VAEs is to improve scores on the *ground-truth distance correlation* and *linearity ratio* from Chapter 4. We achieve this goal by running experiments with the β-TVAE trained on triplets sampled using the ground-truth distance between factors $\mathrm{s}_{\text{f-dist}}$ as the supervision signal. The anchor, positive, negative triplets are sampled such that $\mathrm{s}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(p)}) < \mathrm{s}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(n)})$. We perform a grid search over common disentanglement datasets versus our adversarial XYSquares dataset, ℓ_1 versus ℓ_2, ground-truth $\mathrm{s}_{\text{f-dist}}$ versus normalised ground-truth $\mathrm{s}_{\text{f-dist-scaled}}$ distance, and the hard-margin versus the soft-margin formulation of triplet loss. Since supervision is used, we use a larger latent space size of 25 which additionally makes the disentanglement problem harder. Further experiment details are given in Appendix A.

Our results in Figure 5.1 show that compared to traditional methods the β-TVAE successfully improves the correspondence between ground-truth and latent distances. It

also improves the linearity of representations sampled along ground-truth factor traversals. Additionally, the maximum disentanglement scores obtained over a training run are on average better when ℓ_1 distance is used with triplet loss, which verifies our argument from Section 5.3.2. Furthermore, to decide between the hard-margin and soft-margin formulation of triplet loss we examine the minimum reconstruction loss obtained over each training run in Figure 5.3.

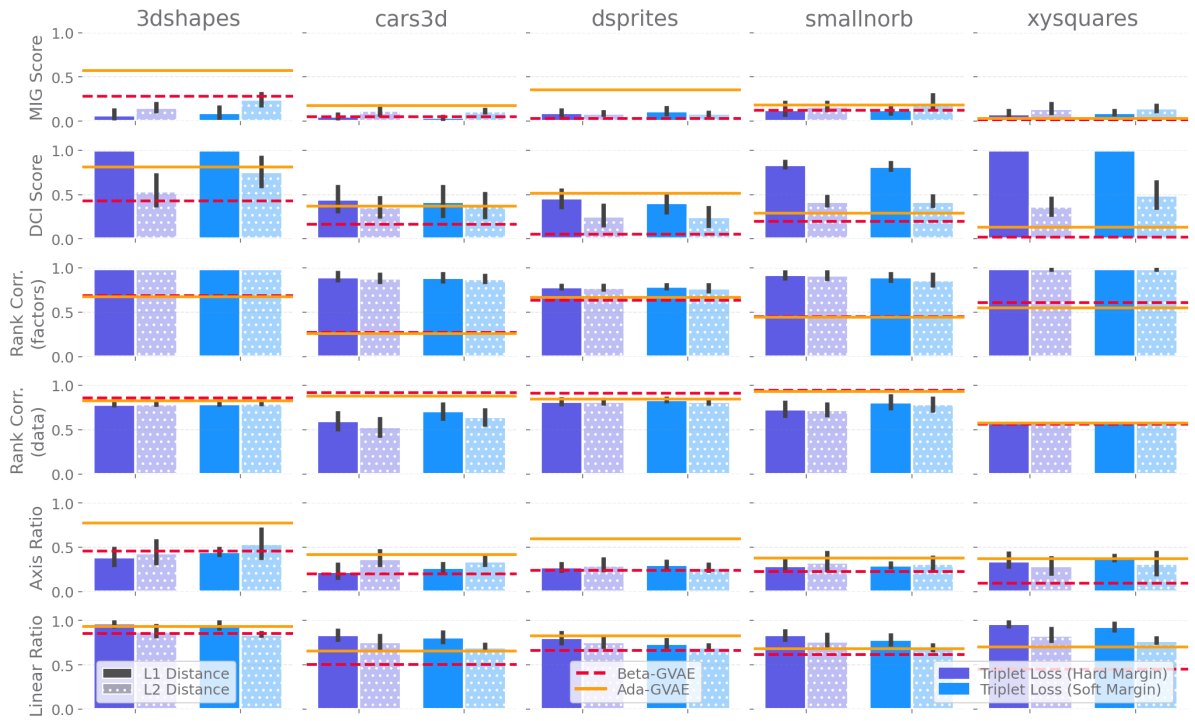


Figure 5.1: Disentanglement scores for the supervised β-TVAE framework over common disentanglement datasets versus our adversarial XYSquares dataset. This experiment indicates that ℓ_1 distance used for triplet loss gives better disentanglement results. Injecting distance information into VAEs with triplet loss successfully improves the ground-truth and latent distance correspondence (factor rank correlation), as well as the overall linearity ratios. We compare our results with those of Figure 4.4 by overlying the averaged scores for the β-VAE and Ada-GVAE frameworks as the horizontal dashed and solid lines, respectively. The error bars represent repeated runs with adjusted scales for triplet loss and the triplet margin.

Upon closer visual inspection of latent traversals in Figure 5.2, this linearity of the latent space along an arbitrary basis can be visually seen as the points moving in unison across the different latent traversals. These results are significant because with a suitable rotation of the latent space the learnt representations may appear factored and thus disentangled. We seek to enforce this axis-alignment to improve the *axis-alignment ratio* in the next section.

The reconstruction loss can be used as a measure of how much information is present in a VAE representation. Looking at disentanglement in Figure 5.1, the soft and hard

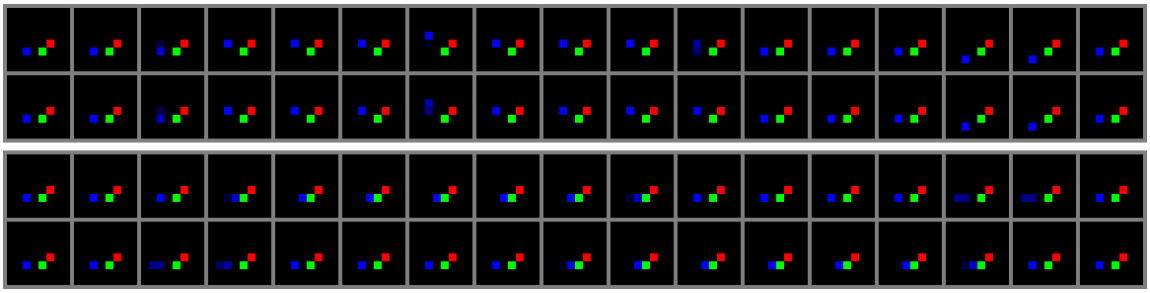
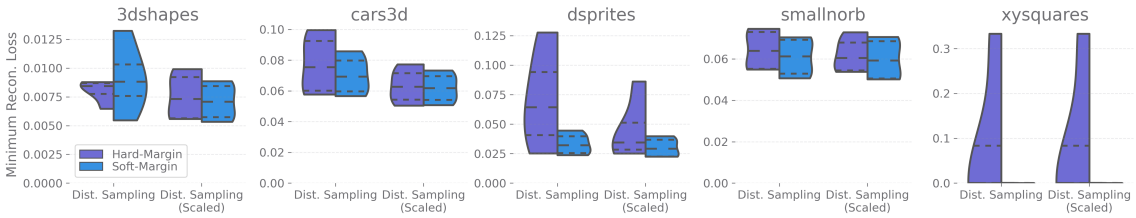
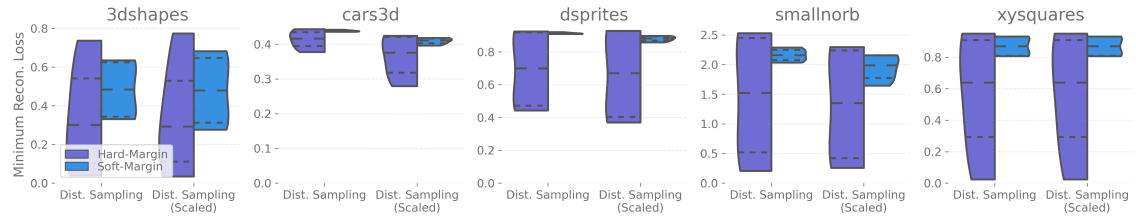


Figure 5.2: Traversals of different latent units of a β-TVAE model with almost perfect scores for the linearity ratio. The blue square follows similar movement patterns in the first and second rows, as well as the third and fourth rows, suggesting an arbitrary rotation of the latent space.



(a) Triplet Loss With VAE



(b) Triplet Loss With Encoder Only

Figure 5.3: Plots of the minimum reconstruction loss obtained during training runs from Figure 5.1 for different triplet loss configurations (Lower is Better). Only results for runs using the ℓ_1 distance are shown. (a) Standard β-TVAE trained with triplet loss, the gradient from the decoder flows trough to the decoder. Both hard and soft triplet losses obtain similar minimum scores over multiple runs. (b) Encoder trained only using triplet loss, the gradient from the reconstruction loss does not flow from the decoder through to the encoder. The hard-margin formulation of triplet loss is more versatile and often obtains much lower minimum reconstruction loss scores, however, careful tuning of the margin parameter ψ is required.

margin formulations of triplet loss usually produce similar results; however, when examining the reconstruction loss obtained in Figure 5.3, the hard-margin formulation is more versatile on average. The main focus of our experiments is to inject supervised information into the representations learnt by VAEs, however, triplet loss should also be capable of use without a decoder. For this reason, we use the hard-margin formulation of triplet loss for the remaining experiments, since it is more versatile and can be

used with and without the decoder, or with and without the gradient from the decoder. Our goal, however, is to guide the distances learnt by VAEs. We leave the decoder-less learning procedure as a point of interest, and we use the full VAE setup throughout the remainder of this work, with both the encoder and decoder.

5.4 Axis-Aligned Metric Learning

In the previous Section 5.3.3, we applied triplet loss to improve the correspondence between ground-truth and latent distances learnt by the model. However, standard triplet loss still does not provide direct pressure to learn factored representations by encouraging axis-alignment, and thus learnt representations still achieve low disentanglement scores on most metrics.

Before we can modify the β-TVAE to encourage axis-aligned representations, we need to formulate our approach in a general manner such that it can be applied to both standard encoders (eg. AEs) as well as probabilistic encoders (eg. VAEs). The original Adaptive VAEs estimate the shared factors between two input observations using a threshold τ, computed as the average of the minimum and maximum KL divergence between individual latent units. We instead compute this threshold over the absolute difference of the deterministic representations and additionally allow the strength of the threshold to be controlled in Section 5.4.1.

Finally, we derive the adaptive triplet loss in Section 5.4.3 that encourages latent embeddings to be axis-aligned with the ground-truth distances that generate the triplets. We do this by modifying the anchor-negative distance of triplet loss in Section 5.4.3, taking inspiration from the adaptive averaging approach of the Adaptive VAE methods from Locatello *et al.* [2020]. This new anchor-negative distance term weights individual latent units less if they are considered shared.

5.4.1 Problems Estimating Shared Variables

The original adaptive methods estimate shared latent units using the KL divergences between individual latent distributions themselves, see Section 2.3.3. There are three problems with this approach if we are to apply adaptive methods in a triplet setting:

1. The adaptive framework can only estimate distances between latent units over probabilistic encoders (eg. from VAEs) and not standard encoders (eg. from AEs).
2. KL distances between anchor-positives and anchor-negatives may not correspond to the triplet distance measures between those same representations.
3. There is no way to control the strength of the threshold when estimating shared latent units; in certain cases the threshold may be too high and discourage effective learning.

The first problem (Item 1) with the adaptive method from Locatello *et al.* [2020] con-

cerns estimating the shared latent units from the distances between embeddings of two different inputs $\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)} \in \mathcal{X}$. These distances δ_i between latent units $i \in \mathrm{D}$ are typically computed from KL divergence between corresponding latent distributions output by the probabilistic encoder: $\delta_i = \tilde{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(a)}) \parallel q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(b)})\right)$. This is not general since we cannot estimate shared latent units between representations output by a standard deterministic encoder. In this case, the only natural way to estimate such differences is by taking the absolute values of the differences between corresponding latent units: $\delta_i = |z_i^{(a)} - z_i^{(b)}|$. Extending this case back to probabilistic encoders, we use the same strategy as for inference after training, where representations are taken to be the mean of the distributions output by the encoder $\boldsymbol{z} = \boldsymbol{\mu}$, instead of sampling from these distributions themselves $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$. The resulting distance between latent units in the case of probabilistic encoders is thus: $\delta_i = |\mu_i^{(a)} - \mu_i^{(b)}|$.

The second problem (Item 2) is that since $\boldsymbol{\sigma}$ is learnt alongside $\boldsymbol{\mu}$ for a standard probabilistic encoder, if the KL divergence is used in a triplet scenario with inputs $\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(p)}, \boldsymbol{x}^{(n)}$, situations can arise for latent variables $i \in [\mathrm{D}]$ where the positive is incorrectly considered closer to the anchor than the negative. If the KL divergence considers latent variables closer together than some distance function, the equivalence of inequalities from Equations 5.13 and 5.15 is violated.

$$\tilde{D}_{\mathrm{KL}}\left(\mathcal{N}(\mu_i^{(a)},\ \sigma_i^{(a)}) \parallel \mathcal{N}(\mu_i^{(p)}, \sigma_i^{(p)})\right) \leq \tilde{D}_{\mathrm{KL}}\left(\mathcal{N}(\mu_i^{(a)},\ \sigma_i^{(a)}) \parallel \mathcal{N}(\mu_i^{(n)}, \sigma_i^{(n)})\right) \tag{5.13}$$

$$\sigma_{p,i}^{-2}\left[(\sigma_{a,i}^2+\sigma_{p,i}^2)(\mu_{a,i}-\mu_{p,i})^2+\sigma_{a,i}^4+\sigma_{p,i}^4\right] \leq \sigma_{n,i}^{-2}\left[(\sigma_{a,i}^2+\sigma_{n,i}^2)(\mu_{a,i}-\mu_{n,i})^2+\sigma_{a,i}^4+\sigma_{n,i}^4\right] \tag{5.14}$$

$$\not\Longleftrightarrow$$

$$|\mu_i^{(a)} - \mu_i^{(p)}| \leq |\mu_i^{(a)} - \mu_i^{(n)}| \tag{5.15}$$

The third problem (Item 3) is that there is no way to control the strength of the threshold when estimating shared latent units. While the threshold estimation of the original Adaptive VAE approaches work well in their original context, when adapting the approach for our purposes we find that the threshold is often too high and averages together too many latent units early on in training. We thus propose linearly interpolating between the min and the max deltas when computing the threshold using $\eta \in [0, 1]$, as given by Equation 5.16. We call η the *averaging strength* of the *averaging threshold* τ. When $\eta = 0$ no latent variables are considered shared, when $\eta = 0.5$ the equation is equivalent to the original approach. We generally recommend that values for η be kept in the range $[0, 0.5]$; values higher than 0.5 usually average together too many latent units.

$$\tau = (1 - \eta) \cdot \left(\min_{i \in [\mathrm{D}]} \delta_i\right) + \eta \cdot \left(\max_{i \in [\mathrm{D}]} \delta_i\right) \tag{5.16}$$

5.4.2 Summary Of Adaptive Methods

Integrating the changes that address the problems identified in Section 5.4.1, we produce a summary in Table 5.1 of the different averaging approaches using the original Adaptive VAE notation from Section 2.3.3. We assume that the VAEs use the standard factorised Gaussian posteriors as in Section 2.1.3 such that $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$.

<table>
<tr><th>Adaptive VAEs (Orig.)</th><th>Adaptive VAEs (Ours)</th><th>Adaptive AEs (Ours)</th></tr>
<tr><td>$\delta_i = \tilde{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(u)}) \parallel q_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}^{(v)})\right)$</td><td colspan="2">$\delta_i = |\mu_i^{(u)} - \mu_i^{(v)}|$</td><td>$\delta_i = |z_i^{(u)} - z_i^{(v)}|$</td></tr>
<tr><td>$\tau = \frac{1}{2}\left(\min_{i\in[\mathrm{D}]} \delta_i + \max_{i\in[\mathrm{D}]} \delta_i\right)$</td><td colspan="3">$\tau = (1-\eta)\cdot\left(\min_{i\in[\mathrm{D}]} \delta_i\right) + \eta\cdot\left(\max_{i\in[\mathrm{D}]} \delta_i\right)$</td></tr>
<tr><td colspan="4">$S = \{i \mid \delta_i < \tau,\ \forall\, i \in [\mathrm{D}]\}$</td></tr>
</table>

Table 5.1: A comparison of the original averaging approaches for the Adaptive VAE methods by [Locatello *et al.* 2020] versus the changes we introduce to fix the problems from Section 5.4.1. Our approaches standardise the adaptive averaging over both AEs and VAEs. Note that given two input observations, δ_i is the estimated distance between corresponding latent units $i \in [\mathrm{D}]$. τ is the *averaging threshold* that is estimated from these latent distances. η is the *averaging strength* that we introduce to control the averaging threshold. $S \subseteq [\mathrm{D}]$ is the set of all latent units that have been estimated to be shared.

5.4.3 Adaptive Triplet Loss

Standard triplet loss provides no direct pressure to learn factored representations or axis-aligned representations; for example x and y factors may be encoded with some arbitrary rotation of the latent space. To fix this, we take inspiration from the Adaptive method of Locatello *et al.* [2020] and those from Section 5.4.2 to construct Ada-Triplet (Adaptive Triplet). This adaptive version of triplet loss aims to encourage axis-alignment of learnt distances, such that representations appear factored.

Before we can construct this axis-aligned version of triplet loss, we identify key issues of directly applying the original Adaptive-VAE hard-averaging procedure from Section 2.3.3 to anchor-positives and anchor-negatives before being passed to standard triplet loss.

1. Applying the hard-averaging procedure to both the anchor-positive and anchor-negative pairs is unintuitive. There is no clear way to estimate the shared variables between all three possible pairs of representations.
2. If we pass averaged representations to the anchor-positive and anchor-negative distance functions of triplet loss, the gradient over the averaged latent units would

become zero. Take for example averaged representations $\tilde{\boldsymbol{z}}^{(a)}, \tilde{\boldsymbol{z}}^{(n)}$ that are computed from original representations $\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)}$, such that the latent units estimated to be shared are all equal $z_i^{(a)} = z_i^{(n)} \quad \forall i \in S$ and the non-shared latent units differ $z_i^{(a)} \neq z_i^{(n)} \quad \forall i \in \bar{S}$. The problem is that $\delta_i = 0 \; \forall i \in S$ will now be zero for all latent units considered shared.

To remedy these issues, we first require a soft-averaging approach that weights distances between shared latent units less rather than setting these distances to zero. Secondly, we require that the averaging procedure is only applied to the anchor-negative pair of triplet loss. Since the anchor-negative term should be further in distance than the anchor-positive we make the assumption that if the anchor-negative has already encouraged correct axis-alignment of the latent units then the anchor-positive will also be correctly aligned. With these requirements, we give the basic *Adaptive Triplet Loss* in Equation 5.17 and the soft-margin formulation in Equation 5.18. These new losses simply replace the anchor-negative distance term with the modified function $\mathrm{d}_{\mathrm{Ada}-\ell_p}$ which we seek to define.

$$\mathcal{L}_{\text{Ada-Triplet-Hard}} = \max\left(0, \quad \psi + \mathrm{d}_{\ell_p}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) - \mathrm{d}_{\mathrm{Ada}-\ell_p}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)})\right) \tag{5.17}$$

$$\mathcal{L}_{\text{Ada-Triplet-Soft}} = \ln\left(1 + \exp\left(\mathrm{d}_{\ell_p}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(p)}) - \mathrm{d}_{\mathrm{Ada}-\ell_p}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)})\right)\right) \tag{5.18}$$

To encourage axis alignment using the anchor-negative term, the intuition is that latent units i that are considered shared $i \in S$ should be weighted less $w_i = \omega$ and non-shared units $w_i = 1$ should remain unchanged. We define the *shared weight* as $0 < \omega < 1$. These weights are applied by elementwise multiplying $\odot$ the arguments by *shared weight vector* $\boldsymbol{w} = (w_1, ..., w_{\mathrm{D}})$ which we compute using Equation 5.19.

$$w_i = \begin{cases} 1 & \text{if not shared } i \in \bar{S} \\ \omega & \text{if shared } i \in S \end{cases} \tag{5.19}$$

The soft-averaging procedure is given by the modified anchor-negative term $\mathrm{d}_{\mathrm{Ada}-\ell_p}$ which we define in Equation 5.21.[2] Axis-alignment is encouraged since it is preferable to encode distances using the unweighted variables.

$$\mathrm{d}_{\mathrm{Ada}-\ell_p}(\boldsymbol{z}^{(a)}, \boldsymbol{z}^{(n)}) = \mathrm{d}_{\ell_p}(\boldsymbol{w} \odot \boldsymbol{z}^{(a)}, \boldsymbol{w} \odot \boldsymbol{z}^{(n)}) \tag{5.20}$$

$$= \left(\sum_{i=1}^{\mathrm{D}} (w_i \cdot \delta_i)^p\right)^{\frac{1}{p}} \tag{5.21}$$

We argue that this *adaptive triplet loss* formulation has additional advantages: 1. The

[2]The real number $p \geq 1$ is the degree of the p-norm or ℓ_p norm. This idea is similar to the generalised mean. $p = 1$ gives the Manhattan ℓ_1 norm, while $p = 2$ gives the euclidean ℓ_2 norm.

strength of the soft-averaging procedure can easily be controlled. 2. Only modifying the anchor-negative term is simple to implement and easier to reason about than estimating differences between all three observations simultaneously. 3. The unmodified original anchor-positive distance function may act as a regulariser, correcting mistakes made by the adaptive anchor-negative distance function. Since the anchor-negative term may push incorrect values further away as they are estimated, the anchor-positive term counteracts this by continually pulling together all latent units.

5.4.4 Ada-TVAE

In order to encourage axis-alignment we propose the Ada-TVAE (Adaptive Triplet VAE) framework using the same construction as that of the β-TVAE from Section 5.3.1. However, we simply replace the *triplet loss* term with that of *adaptive triplet loss* from Section 5.4.3. The final augmented loss becomes:

$$\mathcal{L}_{\text{Ada-TVAE}} = \frac{1}{3} \sum_{\boldsymbol{x} \in \mathcal{X}_{\text{triple}}} \mathcal{L}_{\beta\text{-VAE}}(\boldsymbol{x}) + \alpha \mathcal{L}_{\text{Ada-Triplet-Soft}}(\boldsymbol{\mu}^{(a)}, \boldsymbol{\mu}^{(p)}, \boldsymbol{\mu}^{(n)}) \tag{5.22}$$

5.4.5 Adaptive Weight Schedule

The problem with adaptive triplet loss is choosing values for ω as defined in Section 5.4.3. If we choose values of ω that are too low, then the adaptive triplet loss might collapse to a state where distances cannot be learnt, while values that are too high would approximate standard triplet loss and disentanglement might not occur.

Similarly, choosing different values for the averaging threshold τ will affect how easily latent variables are considered shared because of the averaging threshold τ. Values for τ that are too high may lead to a similar collapse where distances cannot be easily encoded, while values that are too weak may lead to no disentanglement as not enough latent units are considered shared and are thus weighted with ω.

We notice in our experiments that it is better to learn correct distances first, and then introduce the disentanglement pressure via the adaptive triplet loss. Without a slow increase of this adaptive pressure, training often decays immediately. The pressure is slowly increased over the course of a training run by increasing the *averaging strength* η or decreasing the *shared weighting* ω. Optimal axis-alignment may be obtained before the schedule is completed, after which the averaging will be too strong and training destabilises. One can often detect this by examining the reconstruction loss.

In our experiments, we test multiple hyper-parameter schedules for both ω and η. These schedules, given in Table 5.2, aim to slowly increase the adaptive pressure over the course of the training run. We linearly interpolate between the start and end values at each step of the training run, based on the percentage completion at each step.

Schedule Name	Shared Weight (ω)	Averaging Strength (η)
1. Threshold	1.0	$0.0 \rightarrow 0.5$
2. Weight (weak)	$1.0 \rightarrow 0.5$	0.5
3. Weight (strong)	$1.0 \rightarrow 0.0$	0.5
4. Both (weak)	$1.0 \rightarrow 0.5$	$0.0 \rightarrow 0.5$
5. Both (strong)	$1.0 \rightarrow 0.0$	$0.0 \rightarrow 0.5$

Table 5.2: Hyper-parameter schedules for the *shared weight* ω and the *averaging strength* η. We linearly interpolate between the start and end values at each step, over the course of the training run.

5.4.6 Experimental Results

The goal of combining the adaptive methods with triplet loss in Section 5.4.3 is to improve scores on the *axis-alignment ratio* from Chapter 4. The secondary goal of our adaptive triplet construction is to remedy the issues that we describe in Section 5.4.1, notably, estimating shared variables using the absolute distance rather than KL divergence.

We verify these goals by running experiments with the Ada-TVAE framework trained using different schedules for the adaptive hyper-parameters as described in Section 5.4.5. We perform a grid search over these schedules, the common disentanglement datasets versus our adversarial XYSquares dataset, the absolute distance versus KL divergence for threshold estimation, and ground-truth $\text{s}_{\text{f-dist}}$ versus normalised ground-truth $\text{s}_{\text{f-dist-scaled}}$ distance for sampling triplets. Further experiment details can be found in Appendix A.

Our results in Figure 5.4 show that compared to standard triplet loss in Figure 5.1, the adaptive triplet methods introduce axis-alignment. The Ada-TVAE not only obtains excellent scores for the linearity ratio, but also remedies and obtains almost perfect scores for the axis-alignment ratio too, as compared to the standard β-TVAE. Furthermore, these results show that use of the normalised ground-truth distance $\text{s}_{\text{f-dist-scaled}}$ from Section 5.2.4 is usually much better than the standard ground-truth distance $\text{s}_{\text{f-dist}}$ from Section 5.2.3 for sampling triplets. Finally, the maximum disentanglement performance over training runs for each hyper-parameter schedule from Section 5.4.5 is similar. However, we note that the stability of the *strong* versions of the schedules (schedules 3. & 5. from Table 5.2) is reduced and performance decays towards the end of the run. We disregard these schedules from further analysis.

We further narrow down our choice of hyper-parameter schedule in Figure 5.5. These results show that the use of the absolute distance over the KL divergence improves the disentanglement scores on average, confirming our hypothesis from Section 5.4.1. We also compare our results with the averaged scores of the unsupervised β-VAE and weakly supervised Ada-GVAE from Figure 4.4; the Ada-TVAE greatly improves disentanglement performance over prior frameworks. We note that the threshold-only schedule (schedule 1. from Table 5.2) is unstable when examining the reconstruction loss. While this schedule is still stable in terms of disentanglement performance, the reconstruction loss spikes more frequently. Furthermore, the weak combined schedule (schedule 4.

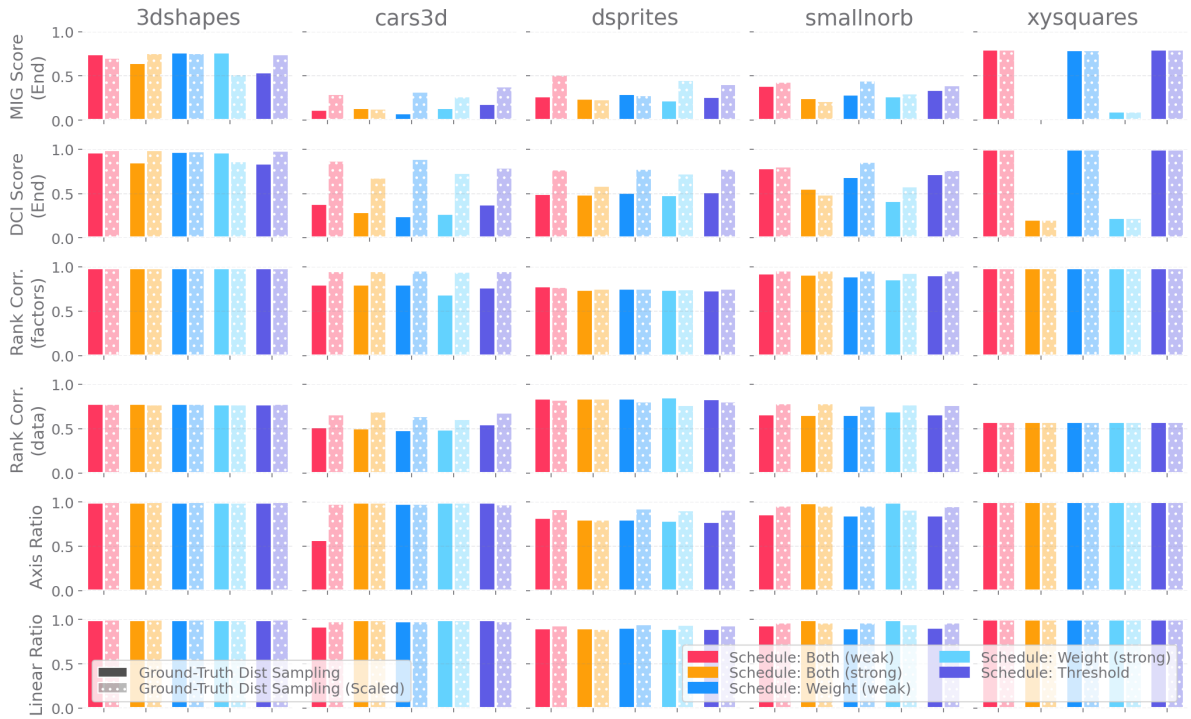


Figure 5.4: Disentanglement scores for the supervised Ada-TVAE framework over common disentanglement datasets versus our adversarial XYSquares dataset. This experiment verifies that using the normalised ground-truth distances $s_{\text{f-dist-scaled}}$ from Section 5.2.4 helps improves axis-alignment compared to the ground-truth distance $s_{\text{f-dist}}$ from Section 5.2.3. Furthermore, this experiment rules out the strong versions of the hyper-parameter schedules from Section 5.4.5 for the adaptive method. We compare our results against the Ada-GVAE and β-VAE in Figure 5.5.

from Table 5.2) takes much longer to produce disentangled representations on average. As such, we disregard both schedules from further consideration and choose the weak schedule for the shared weight ω (schedule 2. from Table 5.2) as the final choice.

We visually confirm the axis-alignment and disentanglement of the supervised Ada-TVAE models by inspecting their latent units through latent traversals in Figure 5.6. Each ground-truth factor of the XYSquares dataset is controlled by a single latent variable and thus the model is successfully disentangled through the introduction of our adaptive triplet approach from Section 5.4.3.

5.5 Unsupervised Adaptive Triplet

In fixing the issue of axis-alignment from Item 3, we have introduced supervision through the use of triplet loss and our proposed Adaptive Triplet Loss from Section 5.4.3. The supervision is because we sample triplets using distance between ground-truth fac-

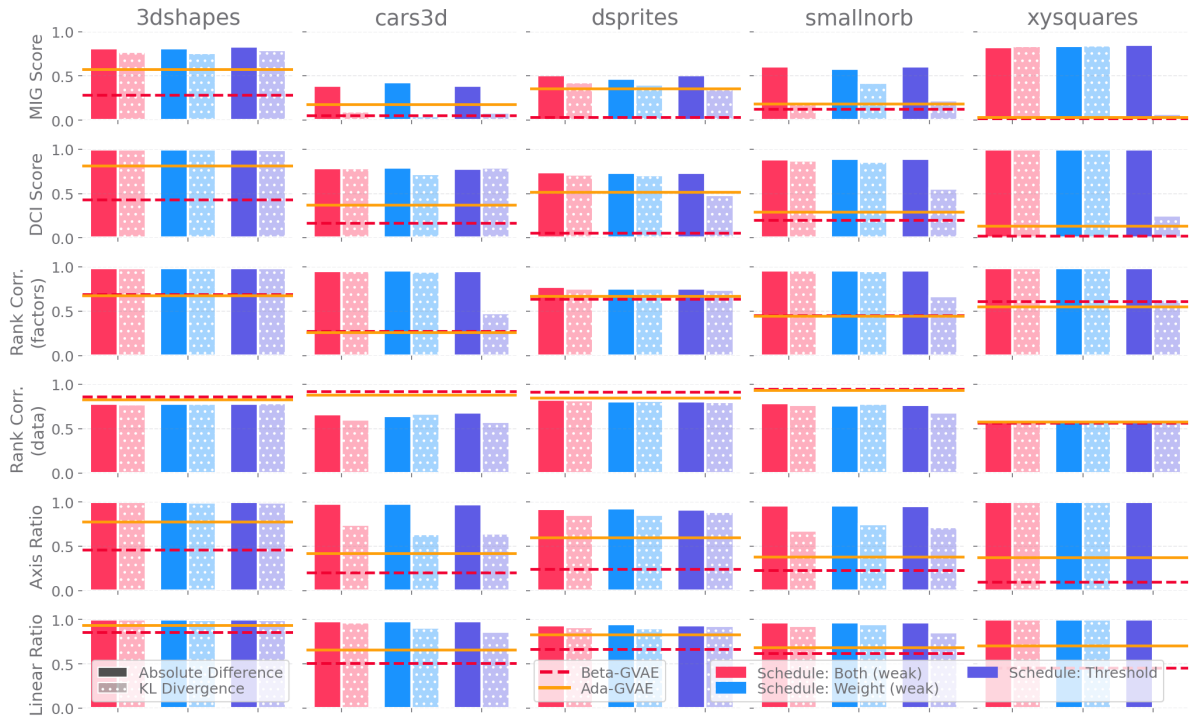


Figure 5.5: Disentanglement scores for the supervised Ada-TVAE framework over common disentanglement datasets versus our adversarial XYSquares dataset. We compare our results with those of Figure 4.4 by overlying the averaged scores for the β-VAE and Ada-GVAE frameworks as the horizontal dashed and solid lines, respectively. The Ada-TVAE greatly improves the disentanglement performance. This experiment also indicates that replacing the KL divergence with the absolute distance between latent units improves the estimation of shared latent units (see Section 5.4.1). Furthermore, the weak schedule for the shared weight ω performs the best on average.

tors as defined in Equation 5.9.

The only way to remedy this issue and revert to an unsupervised approach is by relying on the accidental correlation between ground-truth distances and perceived distances between data according to the reconstruction loss. We thus convert the supervised *Adaptive Triplet* loss into an unsupervised approach by sampling triplets from random mini-batches of observations during training using the perceived distances between observation according to the perceived distance function.

5.5.1 Sampling Based on Perceived Distance

Formally, we relate back to Section 5.2 and define this sampling distance function in Equation 5.24.

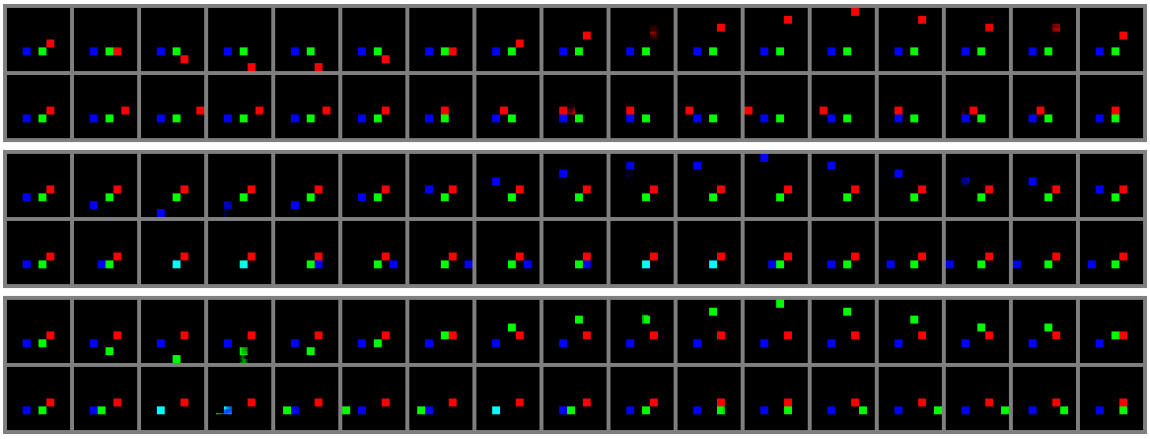


Figure 5.6: Traversals of different latent units of a supervised Ada-TVAE model with almost perfect scores for the axis-alignment ratio. The squares follow almost identical movement patterns compared to the ground-truth factors. Representations are successfully factored and disentangled. The first and second rows show y and x movement of the red square, the third and forth rows show movement of the blue square, the fifth and sixth rows show y and x movement of the green square.

$$\mathrm{s}_{\mathrm{rec}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) = \mathrm{d}_{\mathrm{pcv}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \tag{5.23}$$

$$= \mathcal{L}_{\mathrm{rec}}(\boldsymbol{x}^{(a)}, \boldsymbol{x}^{(b)}) \tag{5.24}$$

While this approach has an upper bound on how disentangled representations may be as argued in Section 4.3, careful choice of the perceived distance function by the researcher can result in improved disentanglement scores. We give an example of such a case in Section 3.4 and Section 4.2.

For improved efficiency during training, we construct triplets during training from dataset mini-batches. For each mini-batch sampled from our dataset $\mathcal{X}_{\mathrm{batch}} \subset \mathcal{X}$, we sample $n \in \mathbb{N}_1$ random triplets of observations from that mini-batch $\mathcal{X}_{\mathrm{anchor}}, \tilde{\mathcal{X}}_{\mathrm{pos}}, \tilde{\mathcal{X}}_{\mathrm{neg}} \subseteq \mathcal{X}_{\mathrm{batch}}$. We then proceed to swap the corresponding positive and negative observations in $\tilde{\mathcal{X}}_{\mathrm{pos}}$ and $\tilde{\mathcal{X}}_{\mathrm{neg}}$ if their anchor-positive and anchor-negative distances are wrong according to $\mathrm{s}_{\mathrm{rec}}$. We obtain the final positive $\mathcal{X}_{\mathrm{pos}}$ and negative $\mathcal{X}_{\mathrm{neg}}$ observations used for the triplets after this swapping procedure.

5.5.2 Results

We validate our unsupervised adaptive triplet approach for disentanglement by comparing against typical VAE based approaches. Our results in Figure 5.7 show that while disentanglement scores are not perfect, as expected, disentanglement scores do improve over datasets where their data distances correlate more with ground-truth distances (see Table 4.1 and Table 4.2). Unlike the standard VAE approaches, stability of disentanglement when enforcing distances using triplet loss is improved.

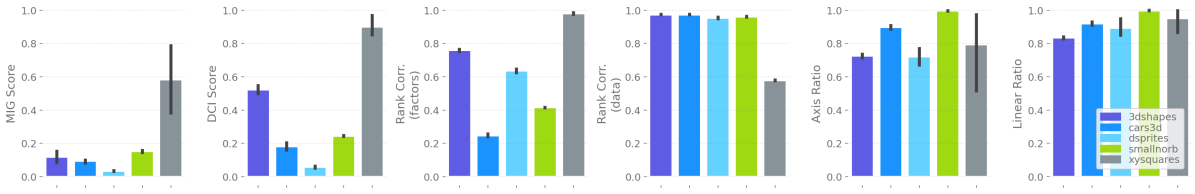


Figure 5.7: Unsupervised Ada-TVAE which constructs triplets using the reconstruction loss. Performance is visualised over differing datasets. The MSE loss is used for the standard datasets, while the augmented MSE-boxblur loss from Section 3.4 is used for the XYSquares dataset. Original unsupervised VAE disentanglement results are given in Figure 3.7 from Section 3.3.3, and Section 4.3 from Figure 4.4.

We suspect that the lower performance compared to the original unsupervised VAE methods can be attributed to the inconsistency of distances between data points compared to the ground-truth factors. These inconsistencies may confuse the adaptive triplet approach which mimics these distances globally over the representation. We hypothesise that only using the decoder combined with random sampling to learn local versions of these distances averages out these distance inconsistencies. Furthermore, the structure of the decoder may bias learning such that neurons activate for nearby units or local features in the data. Using the reconstruction loss as a global distance function may override these local changes that may be more important for disentanglement. We leave an investigation of these biases to future work.

5.6 Summary

In this chapter we show the relationship between supervised metric learning and disentanglement. We then progressively introduce techniques to solve disentanglement using metric learning, according the three components of disentangled factored representations identified in the previous Chapter 4: ground-truth distance correlation, linearity and axis-alignment.

To solve the first two disentanglement metrics, the ground-truth distance correlation and linearity ratio, we first described the metric learning problem Section 5.1 and introduced sampling strategies to construct the observation triplets from ground-truth factors in Section 5.2. We then combine these sampling strategies in Section 5.3 with triplet loss and VAEs to show that this successfully encourages linearity and ground-truth correlation of the latent space, but there is no pressure for axis alignment.

Finally, we introduce axis-alignment by developing a adaptive version of triplet loss that estimates which factors have changed between observations in Section 5.4. Our adaptive triplet loss takes inspiration from the methods of Locatello *et al.* [2020], however, we identify issues with their approach and generalise the approach.

Our supervised adaptive triplet loss successfully solves all three disentanglement crite-

ria from Chapter 4, showing that disentanglement can be seen as a distance learning problem. This aligns well with the ideas from Chapter 3 that suggest that disentanglement in VAEs arises due to the fact that VAEs learn latent distances that correlate with distances between data in terms of the reconstruction loss; however, these data distances accidentally correlate with ground-truth distances. Our supervised approach simply formalises this notion, instead of it being accidental.

Chapter 6

Learning Overlap

Our work identifies the overlooked mechanism for disentanglement in VAEs, which inherently mimic distance learning according to the reconstruction loss. The hand-crafted adversarial XYSquares dataset and naive solution using an augmented loss function are purely examples which break the state of the art (see Sections 3.3 and 3.4). In this chapter we outline methods of using supervision to learn similar adversarial datasets or solutions in terms of the loss function.

6.1 Learning To Disentangle

In Section 3.4 we drastically improve disentanglement by adjusting the loss function to capture spatial differences over the data. This simple, hand-crafted adjustment is intended purely as an example that emphasises the distance learning mechanism within VAEs that gives rise to disentanglement. The problem with adjusting the loss function to improve disentanglement is that it would need to be adjusted per dataset and per choice of ground-truth factors; often this process may be unintuitive.

In this section we seek to automate this process, and learn loss functions that are capable of disentangling. We take inspiration from the fact that augmenting the reconstruction loss improves the correlation between perceived distances in the data space and ground-truth distances. We design our approach to directly optimise our loss function to improve this correlation. We use the fast differentiable sorting algorithm proposed by Blondel *et al.* [2020] to implement a differentiable version of Spearman's rank correlation coefficient. We use this implementation to optimise the loss function to improve correlation between its perceived distances over raw samples from the dataset and corresponding ground-truth distances.

6.1.1 Learning Loss Functions

We seek to learn a loss function $\mathcal{L}^{\theta}_{\text{Learnt}}$, with learnable parameters θ, that should simultaneously enabling disentanglement while still encouraging the decoder to reconstruct the data reliably when used as the reconstruction loss. When we use this learnt loss function for a downstream task, we freeze its parameters.

We train the loss function $\mathcal{L}^{\theta}_{\text{Learnt}}$ using standard machine learning techniques. For each

minibatch sampled from our dataset $\mathcal{X}_{\text{batch}} \subset \mathcal{X}$, we sample $n \in \mathbb{N}_1$ random pairs of observations from that minibatch $\ddot{\mathcal{X}}_{\text{pairs}} \subset \mathcal{X}_{\text{batch}} \times \mathcal{X}_{\text{batch}}$. Each batch has corresponding pairs of ground-truth factors $\ddot{\mathcal{Y}}_{\text{pairs}}$. We use these pairs to compute the differentiable rank correlation between normalised ground-truth distances (see Section 5.2.4) and the perceived distances (see Equation 3.3).[1][2] This correlation becomes the loss function in Equation 6.1 that we wish to optimise using the objective in Equation 6.2.

$$\mathcal{L}^{\theta}_{\text{Learn-Loss}}(\ddot{\mathcal{Y}}_{\text{pairs}},\ \ddot{\mathcal{X}}_{\text{pairs}}) = -\rho_{\text{Spearman}}\left(\mathrm{d}_{\text{gt-scaled}}(\ddot{\mathcal{Y}}_{\text{pairs}}),\ \mathcal{L}^{\theta}_{\text{Learnt}}(\ddot{\mathcal{X}}_{\text{pairs}})\right) \tag{6.1}$$

$$\min_{\theta}\left[\mathcal{L}^{\theta}_{\text{Learn-Loss}}(\ddot{\mathcal{Y}}_{\text{pairs}},\ \ddot{\mathcal{X}}_{\text{pairs}})\right] \tag{6.2}$$

We note the similarity to metric learning approaches in Chapter 5, since both techniques attempt to induce an ordering over distances. While metric learning could be used for this same purpose, we find it is more limiting and more difficult to implement for the objective of learning the loss function.

Furthermore, by simply modifying the ground-truth factor distances or by removing the negation from the rank correlation term in Equation 6.1, we may instead encourage learning of adversarial behaviours. These adversarial behaviours would discourage disentanglement and learning of correct distances. A researcher may want to do this to generate adversarial data or fool disentanglement frameworks. In this work we focus on the synergistic case where we improve disentanglement and leave the adversarial case to future research. However, this ease of switching the optimisation procedure to generating adversarial results is advantageous.

6.1.2 Augmented Loss Function

While our formulation of $\mathcal{L}^{\theta}_{\text{Learnt}}$ can use any differentiable function or machine learning model with appropriate inputs and outputs, the loss function still needs to be able to train a model to reconstruct the data appropriately. We thus construct the function using a similar formulation as that of the hand-crafted example in Section 3.4. We keep the original MSE loss term to enable pixel-wise reconstruction, while introducing overlap and modifying perceived distances using a secondary MSE term that makes use of augmentations. We write our loss function using a learnable data augmentation $h_\theta : \mathbb{R}^{\mathrm{N}} \to \mathbb{R}^{\mathrm{N}}$ in Equation 6.3, and weight the secondary term with $\alpha > 0$.

[1] The normalised ground-truth distance from Section 5.2.4 is the ℓ_1 distance between ground-truth factors where each element is scaled by the total size of the factor itself. For any two ground-truth vectors $\boldsymbol{y}^{(a)}, \boldsymbol{y}^{(b)} \in \mathcal{Y}$ the distance between them is $\mathrm{d}_{\text{gt-scaled}}(\boldsymbol{y}^{(a)}, \boldsymbol{y}^{(b)}) = \sum_{i=1}^{\mathrm{F}} \frac{1}{f_i - 1} |y_i^{(a)} - y_i^{(b)}|$ where f_i is the size of the factor $i \in [\mathrm{F}]$.

[2] We use the overloaded distance notation from Section 4.1.1. For some distance function $d(\cdot, \cdot)$, the result of $d(\ddot{\mathcal{X}})$ is the vector of distances between all the pairs in $\ddot{\mathcal{X}}$.

$$\mathcal{L}^{\theta}_{\text{Learnt}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \mathcal{L}_{\text{rec}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \alpha \mathcal{L}_{\text{rec}}(h_\theta(\boldsymbol{x}),\, h_\theta(\hat{\boldsymbol{x}})) \tag{6.3}$$

With this formulation, the augmentation h_θ can be any learnable model that produces outputs that are the same size as the input. For both simplicity and consistency with Section 3.4, we let h_θ be a single convolutional kernel that we apply to each separate channel of the image. This will work well for the XYSquares dataset by introducing spatial awareness into the loss function, however, more complex datasets will require more complex models. A good choice in practice for these more complex datasets may be a standard Auto-Encoder.

Another significant reason we choose this formulation, is that if we discard the original pixel-wise term, and only keep the second augmented loss term, then we can rephrase the problem as learning augments for data that re-introduce overlap. By re-saving the datasets with these augmentations applied, we enable direct modification of the data to encourage synergistic or adversarial disentanglement behaviour using the original, unmodified loss function. However, we leave this direct learning of augmentations for future work.

6.1.3 Experimental Results

We provisionally test our framework by learning optimised kernels for disentangling varying sizes of the XYSquares dataset. We visualise these kernels in Figure 6.1. We note the distinct patterns that arise corresponding to the locations of the squares in the images. In practice we find that it is beneficial to add symmetric regularisation when learning the kernel and reparameterise the weights so that negative values cannot be learnt. This reparameterisation is similar to how a VAE encoder parameterises the variance (see Section 2.1.3), however, we use absolute value instead of $\ln$. Further experiment details can be found in Appendix A.

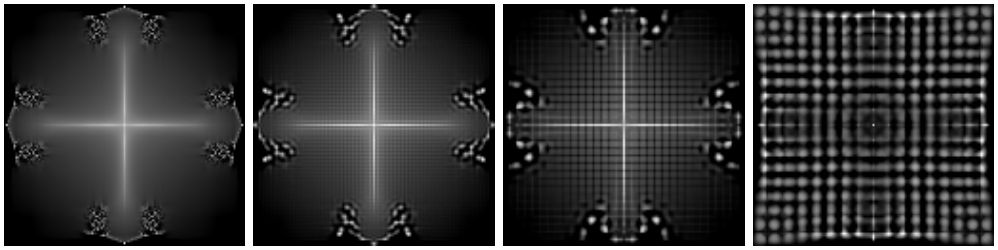


Figure 6.1: Example kernels of radius 63 optimised for variants of the XYSquares dataset. Kernels are trained to maximise the rank correlation between ground-truth distances between observations and distances in the data space according to the reconstruction loss. Kernel outputs are scaled for visual clarity. From left to right kernels are trained on versions of XYSquares with decreasing square sizes: 8x8, 4x4, 2x2 and 1x1.

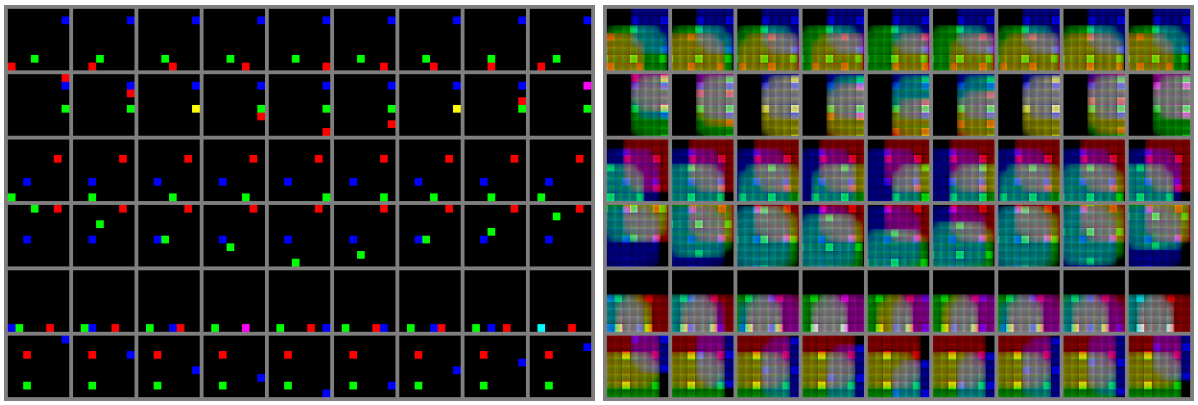
We validate our learnt loss function for the 8x8 XYSquares case by comparing against hand-crafted versions of the loss functions. We visualise the learnt kernel against the

hand-crafted box-blur kernel from Figure 6.2 and a Gaussian blur kernel. We observe the interesting, unintuitive features that arise within the learnt kernel. For interest purposes, we also plot augmented version of the data using the learnt kernel in Figure 6.3. We note that the kernel imitates a blurring operation.



(a) Gaussian Blur **(b)** Box Blur **(c)** Learnt Kernel

Figure 6.2: Hand-crafted and learnt kernels of radius 31 corresponding to the results from Table 6.1. Each kernel is specifically intended to help disentangle the 8x8 XYSquares dataset by introducing perceived overlap. We note the interesting and unintuitive features of the learnt kernel.



(a) Original Data Traversals **(b)** Augmented Data Traversals

Figure 6.3: For interest, we compare raw traversals from the original 8x8 XYSquares dataset against the augmented traversals that we obtain by applying the learnt kernel of radius 31 to the data. We observe that this operation resembles a blurring function which introduces spatial awareness into the learnt loss function by adding overlap between observations.

Finally, we compute the rank correlation between ground-truth distances and the perceived distances between data-points in terms of the differing loss functions according to Chapter 4. We present these results in Table 6.1. We observe that the rank correlation for our learnt loss function is a significant improvement over the hand-crafted baselines. Before running further experiments, we used this method of computing the rank correlation over data to perform a basic hyper-parameter sweep over the weights and sizes of the Gaussian and box-blur kernels. Interestingly, the smaller box-blur kernel of radius 31 performed the best out of the hand-crafted kernels, while the larger learnt kernels of radius 63 obtained the best correlation scores overall. However, for the

sake of fair comparison, we analyse learnt kernels of the same size as the hand-crafted versions.

Loss	Kernel	Radius	Linear Corr. (factor)	Rank Corr. (factor)	Linear Corr. (random)	Rank Corr. (random)
MSE	—	—	0.52	0.58	0.55	0.36
MSE	Gaussian Blur	31	0.84	0.88	0.73	0.61
MSE	Box Blur	31	0.96	0.97	0.95	0.94
MSE	Learnt XY8	31	**0.97**	**0.98**	**0.96**	**0.96**
MSE	Learnt XY8	63	1.00	0.99	1.00	1.00

Table 6.1: Average correlation for varying loss functions between ground-truth distances and perceived distances within the original 8x8 adversarial XYSquares dataset. The hand crafted loss functions score worse compared to optimised loss over the XYSquares dataset. This learnt reconstruction loss obtains much better perceived distance and ground-truth distance correlation. Results can be compared to the original hand-crafted losses from Table 4.2, we additionally use the same experimental setup. Visualisations of these kernels are given in Figure 6.2. Larger learnt kernels perform better, however, we keep the radius fixed at 31 for fair comparisons over VAE experiments.

As we have shown in previous chapters, using this new learnt loss function to improve the ground-truth distance and data distance correlation can improve disentanglement. We validate this claim by comparing the learnt loss function against the results from Section 3.4.1 using a similar experimental setup where we train β-VAE and Ada-GVAE frameworks using the different loss functions. We observe that the disentanglement performance improves in Figure 6.4 for the learnt function compared to the hand-crafted versions.

Our results show that learning the loss functions can improve disentanglement performance by improving the correlation between the ground-truth distances and the perceived distances. This correlation is improved for the XYSquares dataset because spatial awareness is introduced into the model by changing the loss function. This spatial awareness is due to the blur-like operation that the kernels perform in the loss. It should be noted, however, that the results from this section are preliminary in the sense that they are only intended to validate the construction of this framework. We leave disentanglement of other datasets to future work. However, as previously mentioned, more complex models may be needed to capture the correct distances. The results in their current form are still beneficial; even though losses may be trained on synthetic data, once learnt, a researcher may still apply the loss to other datasets for improved results.

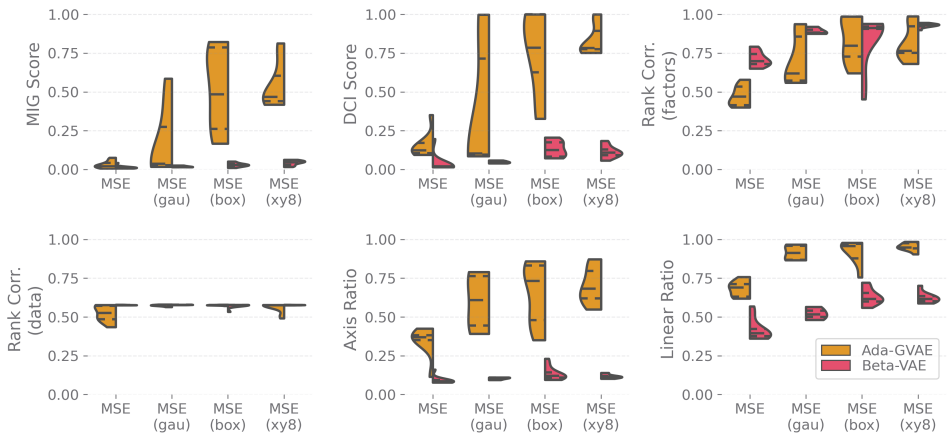


Figure 6.4: Disentanglement scores over the original 8×8 XYSquares dataset for different reconstruction losses. We compare standard MSE loss, the hand-crafted losses, and the learnt loss intended to disentangle the dataset. We train both β-VAEs and Ada-GVAEs using the differing loss functions. Performance is improved for the learnt loss targeted towards this dataset compared to the original MSE loss or hand-crafted versions.

6.2 Summary

In this chapter we have outlined methods for learning loss functions that can improve the correlation between ground-truth distances and perceived distances in data. Improving this correlation improves disentanglement as we have shown in previous chapters. Our learnt loss functions resemble perceptual loss functions [Hou *et al.* 2017], but targeted towards specific features in the data. Unfortunately, learning these loss functions requires supervision; however, such loss functions may be reused between similar tasks. Synthetic data could be used to learn such a function that may then be applied to unlabelled data. We recognise that learning such a loss function for each dataset to improve disentanglement is not feasible, this suggests that truly general disentanglement needs further research or entirely new approaches.

Chapter 7

Considerations for Disentanglement Research

In this chapter, we point out various considerations for disentanglement research based on our results from the previous chapters. In Section 7.1 we emphasise that disentanglement is inherently subjective by highlighting various related pitfalls. We then construct new datasets in Sections 7.2 and 7.3 to reinforce these issues.

7.1 Problems With Disentanglement

We have shown that VAEs reorganise the latent space to minimise the perceived reconstruction error due to random sampling. Furthermore, synthetic datasets typically used in disentanglement research complement this mechanism because their ground-truth distances correlate with data distances. We argue that this mechanism should not be considered as disentanglement. Rather, as Figures 3.5 and 3.11 illustrate, it is closer to distance learning [Yang and Jin 2006; Hoffer and Ailon 2015] and only approximates disentanglement.

We first examine the problems that arise from the correspondence of distances between data and that of distances between ground-truth factors:

1. We observe that adjusting the reconstruction loss changes the representations learnt by affecting overlap, and thus disentanglement. In practice, careful choice and tuning of the loss function may be needed to capture different ground-truth factors. Changing the loss to capture one factor may conflict with the model's ability to learn another factor. See Sections 3.4 and 6.1.
2. Care should be taken in the construction of datasets that are to be used for disentanglement. Injecting more information through overlap is often desirable. For example, invalid intermediate positions on a chess board can be given to help disentangle positional data, as in XYSquares. Similarly, alignment of objects in images may be desirable, which is why disentanglement over face datasets may perform well. If no intermediate examples of factor changes are given, it may be difficult for a model to learn these changes as controllable concepts. See Section 3.3.4
3. Current datasets used for disentanglement generally do not include noise or imperfections that may mask the ground-truth factors. Real-world data is known to

harm performance of disentanglement frameworks [Gondal *et al.* 2019]. Similarly, models may disentangle noisy data, but metrics may not be able to capture this amongst the noise in the representation space.

Furthermore, learning ground-truth factors is an inherently subjective process. There are often multiple ways to represent these factors within datasets and these choices may change depending on the task at hand or the requirements of the researcher:

4. Factors may be decomposed into multiple sub factors or merged into single parent factors. Disentanglement metrics that cannot capture these relationships may incorrectly attribute worse scores. We illustrate this in Figure 7.1, where VAEs often prefer to learn the "azimuth" or "object type" factors using multiple latent variables when trained on Cars3D. However, these representations may still be considered intuitive.
5. Ground-truth factors may not be linear. For example, rainbow-like colours or rotation are usually cyclic in nature. Frameworks and metrics that do not provision for this fact may struggle to learn meaningful representations over these factors or evaluate these representations. A good example of this is HSV versus RGB representations of colours.

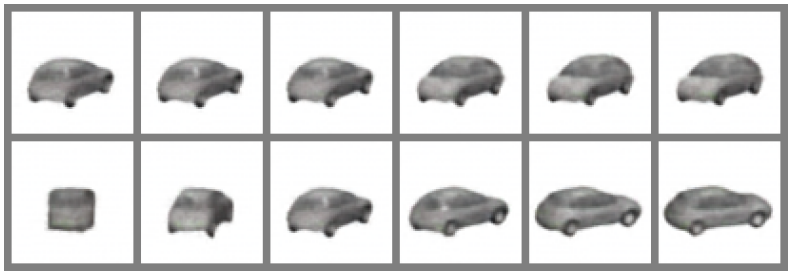
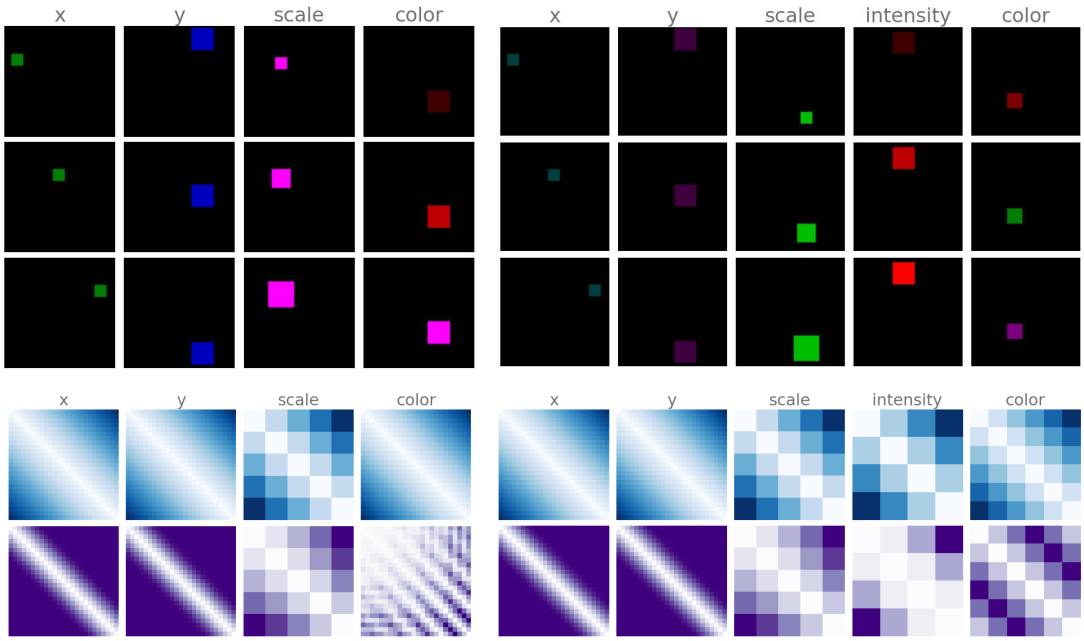


Figure 7.1: Unsupervised models often prefer to learn the "azimuth" factor of the Cars3D dataset with two latent variables: One latent variable that controls a partial rotation (bottom row) and one latent variable that flips the car around (top row). This corresponds to our distance plots in Figure 3.4, which are clearly segmented. Models additionally prefer to learn the large "object type" factor as multiple sub-factors, which may include colour of the car, shape of the car, height of the car, etc.

Disentanglement, in its current state, using VAE frameworks ultimately depends on the researcher's requirements. Care should be taken when using VAEs for disentanglement without considering the problem at hand and the structure of the data. A researcher cannot expect to obtain disentangled results without correct overlap in the data. Furthermore, if the distances present within the data do not align with the factors that the researcher expects to obtain, then VAE disentanglement under their goals is not possible. Furthermore, care should be taken when evaluating disentanglement using existing datasets; these datasets are structured in a way that is already conducive to learning disentangled representations. As such, the results obtained can be disentangled by accident irrespective of special algorithmic choices.

7.2 Choice Of Ground-Truth Factors

We experimentally validate Problems 4 and 5 from Section 7.1 by constructing two equivalent datasets with differing representations of their ground-truth factors. We base this dataset called *XYObject* on dSprites from Matthey *et al.* [2017], with factors for x and y position, scale of the object, and colour of the object. The second version of the dataset called *XYObject (Shades)* represents the colour of the object instead with two separate ground-truth factors being colour and intensity. The observations contained within these datasets and the orderings of these observations are equivalent in every way; the only difference is the representations of their ground-truth factors. We visualise these ground-truth factor traversals for each dataset in Figure 7.2. We also plot the average perceived distances between observations along these factor traversals using the methodology from Section 3.2.4.



(a) XYObject **(b)** XYObject (Shades)

Figure 7.2: The same *XYObject* dataset with different choices for the ground-truth factors. (a) All the colours and shades are mixed into one ground-truth factor. (b) The colours and shades are split into separate ground-truth factors, this representation correlates better with distances in the data space. Raw data traversals are given in the top row of figures, while average distances along these factor traversals are given in the bottom row of figures. The average traversal distances that are shaded blue are the ground-truth distances given for reference, while the purple row at the bottom gives the actual perceived distances between observations along these traversals (see Section 3.2).

We train models over these datasets and examine their disentanglement scores in Figure 7.3. We observe that *XYObject (Shades)* obtains better disentanglement scores.

This is because the ground-truth factors are a more natural representation of the data, and also correspond better with perceived distances between data points. We visually verify this in Figure 7.2. Further experiment details can be found in Appendix A.

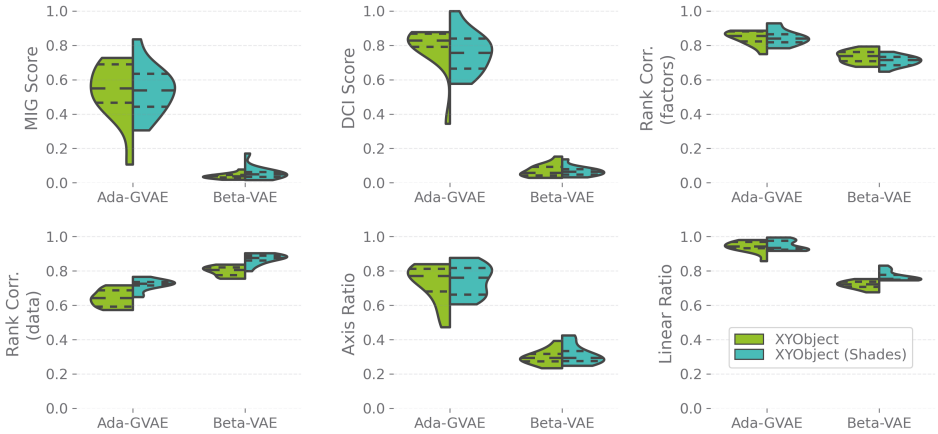


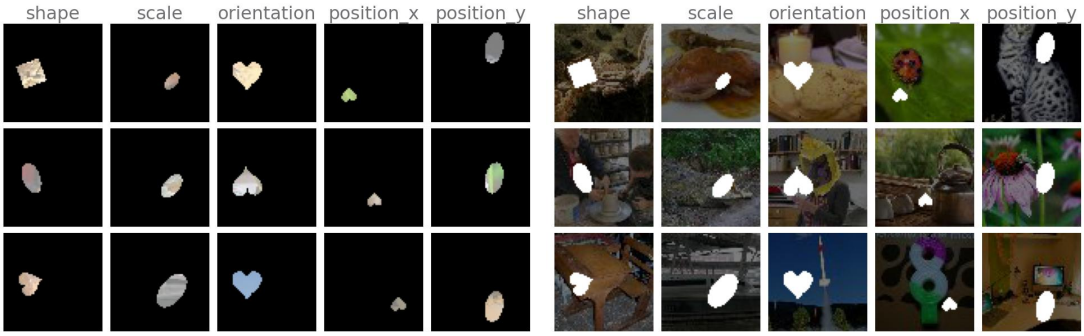
Figure 7.3: Choices for ground-truth factors affect disentanglement scores within datasets. If data distances correlate more with the ground-truth factors, then disentanglement is easier for VAE models. The frameworks perform better with the shaded version of XYObject as seen in Figure 7.2.b.

There results emphasise that the choice of how ground-truth factors are represented can have an impact on disentanglement. The effect is noticeable even though the change is small. Now, consider the Cars3D dataset with its "object type" factor which controls most of the variance within the dataset according to Table 3.1. A model's ability to disentangle may already be better than disentanglement metrics portray over the Cars3D dataset because of this non-optimal representation. Furthermore, this suggests that alternative disentanglement metrics may be needed that are not sensitive to decomposing factors into multiple latent units (see Figure 7.1).

7.3 Random External Factors

We experimentally evaluate problem Item 3 from Section 7.1 by constructing a dataset with meaningful noise that intends to mask the ground-truth factors. We modify the dSprites dataset which consists of binary images [Matthey *et al.* 2017]. The binary nature of the dSprites data makes it easy to extract foreground from background content. Images can directly be used as a mask by pixel-wise multiplying similarly shaped images. We use the *Tiny ImageNet* dataset from Standford's Tiny ImageNet Visual Recognition Challenge, which consists of 100000 training images of size 64×64. We pseudo-randomly attach an image from Tiny ImageNet to each image in dSprites using a fixed random seed. We construct two versions of the dataset, *dSprites-ImageNet-FG* which replaces the foreground content with the ImageNet data, and *dSprites-ImageNet-BG* which replaced the background content with ImageNet data. We additionally allow control of the visibility of the masked ImageNet regions, by linearly interpolating the regions

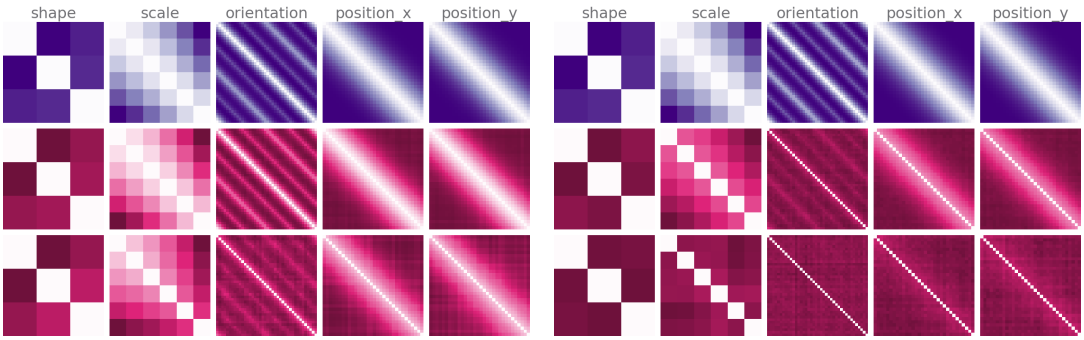
to black or white depending if the content replaces the foreground or background. We visualise these datasets in Figure 7.4.



(a) *dSprites-ImageNet* Foreground **(b)** *dSprites-ImageNet* Background

Figure 7.4: Different versions of the *dSprites-ImageNet* dataset plotted at 50% visibility. (a) The foreground objects are masked out and replaced with noisy content from ImageNet. Foreground pixels are linearly interpolated towards white based on the visibility percentage. (b) The background is replaced with noisy content from ImageNet. Background pixels are linearly interpolated towards black based on the visibility percentage.

The intent of introducing the unrelated content into the dSprites dataset is to change the correlation between ground-truth and perceived distances within the data. We observe these changing distances in Figure 7.5 for differing visibilities of the ImageNet data. On average, these distances are still present, but it is much more difficult for a model to discover them.

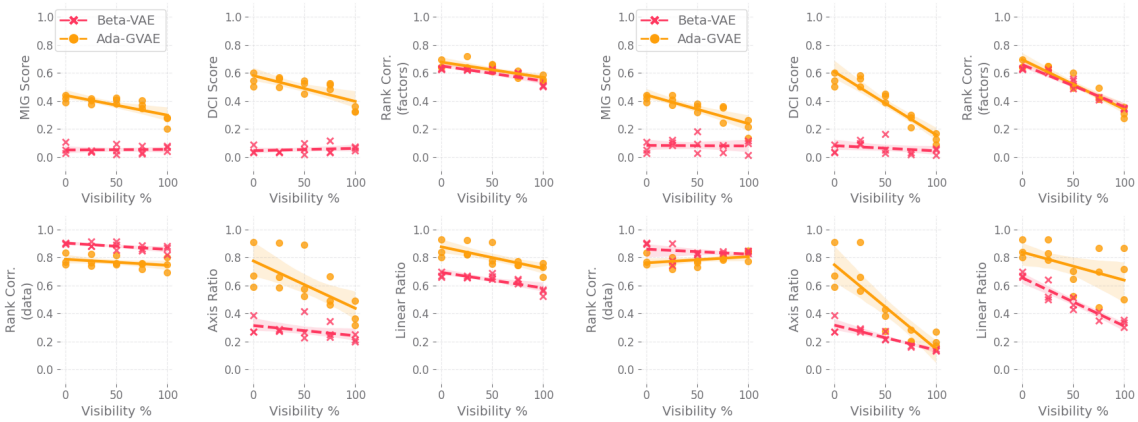


(a) *dSprites-ImageNet* Foreground **(b)** *dSprites-ImageNet* Background

Figure 7.5: Average perceived distances according to MSE loss within *dSprites-ImageNet* with differing visibility percentages. The top row has an ImageNet visibility of 0% and is equivalent to standard dSprites, the middle row has a visibility of 50%, and the bottom row has a visibility of 100%. Increasing the visibility introduces noise which reduces the natural correspondence of perceived distances with ground-truth distances.

We run a sweep of experiments over differing visibilities of the foreground and background versions of the dataset, against the β-VAE and Ada-GVAE frameworks. Further

experiment details can be found in Appendix A. In Figure 7.6 we observe that as the visibility of the ImageNet data increases, the disentanglement performance of the models suffer. The reasoning for this may be two-fold. It may be more beneficial for the models to reduce the reconstruction loss by encoding ImageNet information; these random factors may thus be prioritised over or mixed in with the original ground-truth factors. Secondly, as ImageNet information is encoded via the representations, disentanglement metrics may not be able to discard this information as irrelevant to the actual disentanglement problem. Performance may thus be unintentionally impacted, even if ground-truth factors have been learnt. This further suggests that new disentanglement metrics may be needed.



(a) *dSprites-ImageNet* Foreground **(b)** *dSprites-ImageNet* Background

Figure 7.6: Performance of the β-VAE and Ada-GVAE frameworks over foreground and background versions of *dSprites-ImageNet* with differing visibility. Disentanglement performance decreases as there is more external noise within the data, masking the original ground-truth factors and reducing the correlation between ground-truth and perceived distances.

7.4 Summary

In this chapter, we identified various problems with VAE disentanglement and the current formulation of disentanglement using ground-truth factors. Simply, changing the ground-truth factors or introducing new factors can hinder or encourage disentanglement performance, similarly, disentanglement metrics may not capture or take these factors into account. We contribute two new datasets and their variants that highlight these problems; we suggest that future research use these datasets when evaluating disentanglement.

Chapter 8

Conclusion & Future Work

In the previous chapter, we highlighted various issues that exist with VAE disentanglement research. In this chapter, we propose future research avenues in Section 8.1 and conclude our work by summarising its important aspects and their implications in Section 8.2.

8.1 Future Work

Our work suggests that VAEs are not general disentanglement frameworks that are capable of uncovering ground-truth factors within the data through special algorithmic choices; rather, these frameworks rely on the distances between data points. Although biases may be introduced by regularisation or algorithmic choices within these unsupervised and weakly-supervised models, the distances still remain. Future work should investigate alternative mechanisms for learning disentangled representations that do not depend on distances between data points or a decoder neural network. A promising new direction may be the use of contrastive learning that also includes regularisation to improve disentanglement [Burns *et al.* 2021].

Auto-Encoder and VAE based approaches for disentanglement rely on representations that are fixed in size. These fixed representations cannot adapt to changing environments or datasets; as soon as a new object is added to a scene or attributes are modified that may not have appeared in the training data, the resulting representations may not be useful or disentangled. A better approach may be to design systems that have varying representation sizes that adjust to the amount of information in the scene. These varying representations could have repeated sections that correspond to repeated objects, which, in turn, are all separately disentangled. Furthermore, disentanglement metrics would need to be able to capture such properties.

Our work has also shown that disentanglement is inherently subjective. Researchers should design methods in such a way that disentanglement can be adjusted to the needs of the researcher. As humans, we can adapt our own mental model of a scene to the task at hand; one could design systems that are capable of this behaviour using prior knowledge and a fine-tuning process. Ideas may be taken from the field of natural language processing, including [Devlin *et al.* 2018] who introduce unlabelled pretraining of models, or [Radford *et al.* 2021] who learn visual concepts from natural language.

Our work may have applications in other fields, such as reinforcement learning. If sequential observations are obtained by an agent interacting with an environment, we could use our supervised disentanglement framework from Chapter 5 to learn disentangled concepts. One could make the assumption that observations further in the past differ by more ground-truth factors. This time-based assumption could be used as the supervision signal to generate triplets.

We also show in Section 7.3 that disentanglement using VAEs is affected by noise and additional information present within the data that can conflict with the required ground-truth factors. Frameworks could be designed that separate out these desired ground-truth factors by identifying the noise.

8.2 Conclusion

Throughout this work, we investigated this under-recognised mechanic that VAEs learn distances over the data according to the chosen reconstruction loss. We stress that special algorithmic choices and regularisation within VAEs are insufficient processes for disentanglement, and that learning disentangled representations is largely accidental. Without intuitive perceived distances between data points, disentanglement cannot take place. These distances must correspond to the distances between ground-truth factors, the learnt distances must be linear along some arbitrarily rotated axis, and finally these distances must be axis-aligned such that individual ground-truth factors correspond to individual latent units. With these conditions for learnt distances to result in disentangled and factored representations, we introduced three new disentanglement metrics to quantify each of these separate properties.

Based on this observation that VAEs learn distances over the data, we demonstrated that there are fundamental characteristics of existing benchmark datasets that encourage VAEs to learn disentangled representations. We then exploited this mechanism to construct a simple adversarial dataset for which state-of-the-art models were unable to disentangle. We then proposed multiple solutions to the problem of disentangling the adversarial dataset. The first method involved modifying the reconstruction loss to improve the correspondence between ground-truth factors and perceived distances over the data. The second solution introduced biases through supervised metric learning, where latent distances were adjusted to match ground-truth distances; our approach was modified to encourage disentanglement where usual metric learning approaches otherwise fail. Both techniques can be adapted for use across any dataset to improve disentanglement as long as ground-truth factors are available as a supervision signal.

With solutions available, we then discussed the various problems related to the inherent subjectiveness of VAE disentanglement research. Simply changing the underlying construction of ground-truth factors with otherwise unmodified data can affect disentanglement performance. Furthermore, the introduction of external factors into the data can mask the desired ground-truth factors for disentanglement. Even if existing VAE models appear to disentangle, learning distances over data does not mean that

these models have learnt human concepts about the data. For example, if we seek to use disentanglement to train fair models that make fair predictions, then we need to know that these models are robust and that it is not some accidental property of the data that leads to these seemingly fair predictions; current methods are unsuitable for such a case and future research should investigate new avenues.

If future disentanglement research insists on using unsupervised or weakly supervised VAEs for disentanglement, where disentanglement itself is evaluated using datasets constructed from ground-truth factors, then the researchers should also evaluate their frameworks on the datasets we have proposed in this work. The adversarial XYSquares dataset highlights the importance of choosing the correct reconstruction loss within VAE frameworks. XYObject and its variant highlight the choice of ground-truth factors when constructing datasets. Finally, dSprites-ImageNet and its variants highlight the issues of noise or external factors that may mask the desired ground-truth factors. These datasets can be solved by the supervised metric learning approach we have introduced, however, supervision is not always feasible, and new disentanglement approaches are needed.

Our results throughout this work highlight the issues with current representation learning frameworks. If we ever hope to apply these models to real-world data, then perceived overlap in datasets and the distance learning mechanism within VAEs *cannot* be prerequisites for disentanglement. More advanced methods are required that can uncover true meaning within the data.

Bibliography

[Aggarwal *et al.* 2001] Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*, pages 420–434. Springer, 2001.

[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.

[Blondel *et al.* 2020] Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, pages 950–959. PMLR, 2020.

[Bouchacourt *et al.* 2018] Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[Burgess and Kim 2018] Christopher Burgess and Hyunjik Kim. *3D Shapes Dataset*. https://github.com/deepmind/3dshapes-dataset/, 2018.

[Burgess *et al.* 2017] Christopher Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-VAE. In *Workshop on Learning Disentangled Representations at the 31st Conference on Neural Information Processing Systems,*, 2017.

[Burns *et al.* 2021] Andrea Burns, Aaron Sarna, Dilip Krishnan, and Aaron Maschinot. Unsupervised disentanglement without autoencoding: Pitfalls and future directions. *arXiv preprint arXiv:2108.06613*, 2021.

[Chen *et al.* 2018] Ricky Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pages 2615–2625, 2018.

[Devlin *et al.* 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Dittadi *et al.* 2020] Andrea Dittadi, Frederik Träuble, Francesco Locatello, Manuel Wüthrich, Vaibhav Agrawal, Ole Winther, Stefan Bauer, and Bernhard Schölkopf. On the transfer of disentangled representations in realistic settings. *arXiv preprint arXiv:2010.14407*, 2020.

[Dupont 2018] Emilien Dupont. Joint-vae: Learning disentangled joint continuous and discrete representations. *arXiv preprint arXiv:1804.00104*, 2018.

[Eastwood and Williams 2018] Cian Eastwood and Christopher Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

[Gondal *et al.* 2019] Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. *Advances in Neural Information Processing Systems*, 32:15740–15751, 2019.

[Hermans *et al.* 2017] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[Higgins *et al.* 2017] 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. In *International Conference on Learning Representations*, 2017.

[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.

[Hosoya 2019] Haruo Hosoya. Group-based learning of disentangled representations with generalizability for novel contents. In *International joint conference on artificial intelligence*, 2019.

[Hotelling 1933] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[Hou *et al.* 2017] Xianxu Hou, Linlin Shen, Ke Sun, and Guoping Qiu. Deep feature consistent variational autoencoder. In *2017 IEEE Winter Conference on Applications of Computer Vision*, pages 1133–1141. IEEE, 2017.

[Hsu *et al.* 2012] Daniel J Hsu, Sham M Kakade, and Percy S Liang. Identifiability and unmixing of latent parse trees. In *Advances in neural information processing systems*, pages 1511–1519, 2012.

[Ishfaq *et al.* 2018] Haque Ishfaq, Assaf Hoogi, and Daniel Rubin. Tvae: Triplet-based variational autoencoder using metric learning. *arXiv preprint arXiv:1802.04403*, 2018.

[Karaletsos *et al.* 2015] Theofanis Karaletsos, Serge Belongie, and Gunnar Rätsch. Bayesian representation learning with oracle constraints. *arXiv preprint arXiv:1506.05011*, 2015.

[Kim and Mnih 2018] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, pages 2649–2658. PMLR, 2018.

[Kingma and Ba 2015] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[Kingma and Welling 2014] Diederik Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[Kingma and Welling 2019] Diederik Kingma and Max Welling. *An Introduction to Variational Autoencoders*. Now Foundations and Trends, 2019.

[Krizhevsky *et al.* 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural*

Information Processing Systems, 25:1097–1105, 2012.

[Kumar *et al.* 2018] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. In *International Conference on Learning Representations*, 2018.

[LeCun *et al.* 2004] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[Liu *et al.* 2015] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[Locatello *et al.* 2019a] Francesco Locatello, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem. On the fairness of disentangled representations. In *Advances in Neural Information Processing Systems*, pages 14584–14597, 2019.

[Locatello *et al.* 2019b] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, pages 4114–4124. PMLR, 2019.

[Locatello *et al.* 2020] Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pages 6348–6359. PMLR, 2020.

[Mathieu *et al.* 2019] Emile Mathieu, Tom Rainforth, Nana Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In *International Conference on Machine Learning*, pages 4402–4412. PMLR, 2019.

[Matthey *et al.* 2017] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. *dSprites: Disentanglement testing Sprites dataset*. https://github.com/deepmind/dsprites-dataset/, 2017.

[Mnih *et al.* 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Paszke *et al.* 2017] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. *Advances in Neural Information Processing Systems*, 2017.

[Pearson 1901] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[Radford *et al.* 2021] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language su-

pervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[Reed *et al.* 2015] Scott Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. *Advances in Neural Information Processing Systems*, 28:1252–1260, 2015.

[Ridgeway and Mozer 2018] Karl Ridgeway and Michael C Mozer. Learning deep disentangled embeddings with the f-statistic loss. In *Advances in Neural Information Processing Systems*, pages 185–194, 2018.

[Rolinek *et al.* 2019] Michal Rolinek, Dominik Zietlow, and Georg Martius. Variational autoencoders pursue PCA directions (by accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12406–12415, 2019.

[Sepliarskaia *et al.* 2019] Anna Sepliarskaia, Julia Kiseleva, Maarten de Rijke, et al. Evaluating disentangled representations. *arXiv preprint arXiv:1910.05587*, 2019.

[Suter *et al.* 2018] Raphael Suter, Đorđe Miladinović, Bernhard Schölkopf, and Stefan Bauer. Robustly disentangled causal mechanisms: Validating deep representations for interventional robustness. *arXiv preprint arXiv:1811.00007*, 2018.

[Wang *et al.* 2014] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1386–1393, 2014.

[Xuan *et al.* 2020] Hong Xuan, Abby Stylianou, and Robert Pless. Improved embeddings with easy positive triplet mining. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2474–2482, 2020.

[Yang and Jin 2006] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 2(2):4, 2006.

[Zaidi *et al.* 2020] Julian Zaidi, Jonathan Boilard, Ghyslain Gagnon, and Marc-André Carbonneau. Measuring disentanglement: A review of metrics. *arXiv preprint arXiv:2012.09276*, 2020.

[Zhao *et al.* 2017] Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*, 2017.

[Zietlow *et al.* 2021] Dominik Zietlow, Michal Rolinek, and Georg Martius. Demystifying inductive biases for β-VAE based architectures. *arXiv preprint arXiv:2102.06822*, 2021.

Appendix A

Supplementary Details

A.1 Normalised Axis-Orientation Ratio

We provide the *normalised axis-orientation ratio* purely for interest — we do not consider this score to be one of the *factored component scores* from Chapter 4 rather it is computed from these scores. We also have not used this metric for evaluation purposes throughout the dissertation, however, this metric may have interesting properties to researchers.

The axis-alignment ratio $s_{\text{axis},i}$ in Equation 4.6 is always bounded above by the linearity ratio $s_{\text{linear},i}$ in Equation 4.9, such that $s_{\text{axis},i} \leq s_{\text{linear},i}$, since both metrics are computed using the same formulas over different bases.

Using this result we can compute the axis-orientation ratio given in Equation A.1 as a measure of well points align with an axis independent of their linearity. In other words, information as to the general orientation of points in the latent space towards an axis irrespective of how well points are clustered together along that same line.

$$s_{\text{orientation}} = \frac{s_{\text{axis}}}{s_{\text{linear}}} \tag{A.1}$$

A.2 VAE Implementation Details

In this section, we describe our various implementation details of the Beta-VAE [Higgins *et al.* 2017] and Ada-GVAE [Locatello *et al.* 2020] frameworks.

A.2.1 Beta Normalisation

For general consistency across datasets with different numbers of channels and models with different numbers of latent units, we implement beta normalisation as described by Higgins *et al.* [2017].

Instead of taking the sum over the KL divergence in the regularisation term and the sum over elements in the reconstruction term of the VAE loss, we instead compute the means over elements in both terms and adjust the β value accordingly.

A.2.2 Symmetric KL

The original Ada-GVAE implementation uses the asymmetric KL divergence $D_{\mathrm{KL}}(p \parallel q)$ as the distance function between the corresponding latent units of observation pairs. The Ada-GVAE uses this distance to estimate which of these latent units should be averaged together.

We instead follow the approach of Dittadi *et al.* [2020] and use the symmetric KL divergence to compute these distances between latent units, improving the averaging procedure and computation of the threshold. The symmetric KL divergence is defined in Equation A.2.

$$\tilde{D}_{\mathrm{KL}}(p \parallel q) = \frac{1}{2} D_{\mathrm{KL}}(p \parallel q) + \frac{1}{2} D_{\mathrm{KL}}(q \parallel p) \tag{A.2}$$

A.2.3 Sampling Ada-GVAE Pairs

The Ada-GVAE [Locatello *et al.* 2020] framework introduces weak supervision by sampling pairs of observations such that there are always $k \in [1, \mathrm{F}]$ differing factors between them. We use the weaker but more realistic case where k is sampled uniform randomly for each pair as described in the original paper.

A.3 VAE Experiment Details

In this section, we give further details on the experiments conducted throughout the dissertation and their chosen hyper-parameters. For easier comparison with prior work, we use similar hyper-parameters, optimiser and model choices to Higgins *et al.* [2017], Kim and Mnih [2018] and Locatello *et al.* [2019b]. We also discuss the handling and standardisation of the different datasets.

A.3.1 Optimiser and Batch Size

Models are trained using the Adam [Kingma and Ba 2015] optimiser with a learning rate of 10^{-3}. A batch size of 256 is used in the case of the Beta-VAE [Higgins *et al.* 2017]. In the case of the Ada-GVAE [Locatello *et al.* 2020], 256 observation pairs are used per batch.

A.3.2 Model Architecture

We use similar convolutional encoder and decoder models as Higgins *et al.* [2017], given in Table A.1. The Gaussian encoder parameterises the mean and log variance of each latent distribution. The decoder uses the Gaussian derived Mean Squared Error

(MSE) as the loss function. The number of input channels the encoder receives and the number of output channels the decoder produces depends on the dataset the model is trained on, which is either 1 or 3 channels.

Encoder	**Decoder**
Input {1 **or** 3}x64x64	**Input** {9 **or** 25}
Conv. 32x4x4 (*stride 2*, ReLU)	Linear 256 (ReLU)
Conv. 32x4x4 (*stride 2*, ReLU)	Linear 1024 (*reshape 64x4x4*, ReLU)
Conv. 64x4x4 (*stride 2*, ReLU)	Upconv. 64x4x4 (*stride 2*, ReLU)
Conv. 64x4x4 (*stride 2*, ReLU)	Upconv. 32x4x4 (*stride 2*, ReLU)
Linear 256 (ReLU)	Upconv. 32x4x4 (*stride 2*, ReLU)
2x Linear {9 **or** 25}	Upconv. {1 **or** 3}x4x4 (*stride 2*)

Table A.1: Encoder and decoder architectures based on Higgins *et al.* [2017]. Model inputs and outputs change based on the number of channels in the dataset. Model latent units depend on the experiment hyper-parameters.

A.3.3 Dataset Standardisation

For improved consistency and training performance, dataset observations are standardised. We first resize the observations to a width and height of 64×64 pixels using bilinear filtering. Then the observations are normalised such that on average each channel of the image has a mean of 0 and a standard deviation of 1. Normalisation constants for each channel are precomputed across the entire dataset and are given in Table A.2.

A.3.4 Experiment Sweeps

Experiment figures and tables in each chapter of our work are produced from hyper-parameter sweeps. The grid search values used for these sweeps are given in Tables A.3 to A.7. These hyper-parameter sweeps make use of the default values given above, unless overridden or specified.

Dataset	Mean	Std
Cars3D [Reed *et al.* 2015]	**R**: 0.897667614997663 **G**: 0.889165802006751 **B**: 0.885147515814868	0.225031955315030 0.239946127898126 0.247921063196844
3D Shapes [Burgess and Kim 2018]	**R**: 0.502584966788819 **G**: 0.578759756608967 **B**: 0.603449973185958	0.294081404355556 0.344397908751721 0.366168598152475
Small NORB [LeCun *et al.* 2004]	0.752091840108860	0.095638790168273
dSprites [Matthey *et al.* 2017]	0.042494423521890	0.195166458806261
XYSquares	**R**: 0.015625 **G**: 0.015625 **B**: 0.015625	0.124034734589209 0.124034734589209 0.124034734589209
XYObject XYObject (Shaded)	**R**: 0.009818761549013 **G**: 0.009818761549013 **B**: 0.009818761549013	0.052632363725246 0.052632363725246 0.052632363725246

Table A.2: Precomputed channel-wise normalisation constants for datasets, assuming the input data is in the range $[0, 1]$.

Experiment	Total	Hyper-Parameters
3.3.3 Adversarial Experiments (Figure 3.7)	$8 \times 2 \times 2 \times 5$ $= 160$ $\times 1$ repeats $= 160$	train steps $= 115200$ beta $(\beta) \in \{0.000316, 0.001, 0.00316, 0.01, 0.0316, 0.1, 0.316, 1.0\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents (D) $\in \{9, 25\}$ dataset $\in$ {dSprites, 3D Shapes, Cars3D, Small NORB, XYSquares}
3.3.4 Varying Levels of Overlap (Figure 3.10)	$2 \times 2 \times 8$ $= 32$ $\times 5$ repeats $= 160$	train steps $= 57600$ beta $(\beta) \in \{0.001, 0.00316\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents (D) $= 9$ dataset $=$ XYSquares grid spacing $\in \{8, 7, 6, 5, 4, 3, 2, 1\}$
3.4.1 Augmented Loss Experiments (Figure 3.12)	$2 \times 2 \times 2$ $= 8$ $\times 5$ repeats $= 40$	train steps $= 57600$ beta $(\beta) \in \{0.0001, 0.0316\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents (D) $= 25$ dataset $=$ XYSquares recon. loss $\in$ {MSE, BoxBlurMSE} box blur radius $= 31$ (63×63in size) box blur weight $= 63^2 = 3969$

Table A.3: Grid search hyper-parameters for experiments from Chapter 3.

Experiment	Total	Hyper-Parameters
4.3 VAE Frameworks Learn Perceived Distances (Figure 4.4)	$2 \times 9 \times 2 \times 5$ $= 180$ $\times 1$ repeats $= 180$	train steps $= 57600$ lr $\in \{1e^{-3}, 1e^{-4}\}$ beta $(\beta) \in \{0.0001, 0.000316, 0.001, 0.00316, 0.01, 0.0316, 0.1, 0.316, 1.0\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents (D) $= 25$ dataset $\in$ {dSprites, 3D Shapes, Cars3D, Small NORB, XYSquares}

Table A.4: Grid search hyper-parameters for experiments from Chapter 4.

Experiment	Total	Hyper-Parameters
5.3 Triplet Auto-Encoders (Figures 5.1 and 5.3)	$2 \times 2 \times 2 \times 2$ $\times 2 \times 2 \times 5$ $= 320$ $\times 1$ repeats $= 320$	train steps $= 57600$ margin $(\psi) \in \{1, 10\}$ scale $(\alpha) \in \{1, 10\}$ distance $(\mathrm{d}) \in \{\ell_1, \ell_2\}$ triplet type $\in$ {soft-margin, hard-margin} decoder gradient $\in$ {True, False} dataset $\in$ {dSprites, 3D Shapes, Cars3D, Small NORB, XYSquares}
5.4 Axis-Aligned Metric Learning (Figures 5.4 and 5.5)	$2 \times 2 \times 5 \times 5$ $= 100$ $\times 1$ repeats $= 100$	train steps $= 86400$ sampler $(\mathrm{s}) \in \{\mathrm{s}_{\text{f-dist}}, \mathrm{s}_{\text{f-dist-scaled}}\}$ thresh. dists.$(\delta_i) \in$ {Symmetric KL, Absolute Difference} schedule $\in$ $\{(\omega = 1.0,\ \eta\text{: } 0.0 \rightarrow 0.5)$ $(\omega\text{: } 1.0 \rightarrow 0.5,\ \eta = 0.5)$ $(\omega\text{: } 1.0 \rightarrow 0.0,\ \eta = 0.5)$ $(\omega\text{: } 1.0 \rightarrow 0.5,\ \eta\text{: } 0.0 \rightarrow 0.5)$ $(\omega\text{: } 1.0 \rightarrow 0.0,\ \eta\text{: } 0.0 \rightarrow 0.5)\}$ dataset $\in$ {dSprites, 3D Shapes, Cars3D, Small NORB, XYSquares}
5.5 Unsupervised Adaptive Triplet (Figure 5.7)	2×5 $= 10$ $\times 2$ repeats $= 20$	train steps $= 86400$ sampler $(\mathrm{s}) = \mathrm{s}_{\text{rec}}$ schedule $= (\omega\text{: } 1.0 \rightarrow 0.5,\ \eta = 0.5)$ num triplets $(n) \in \{512, 1024\}$ dataset $\in$ {dSprites, 3D Shapes, Cars3D, Small NORB, XYSquares}

Table A.5: Grid search hyper-parameters for experiments from Chapter 5. The Ada-TVAE framework is used with the following default parameters unless otherwise specified: $\mathrm{D} = 25$ latent units, regularisation scale set to $\beta = 0.01$, *hard-margin* triplet loss using the $\mathrm{d} = \ell_1$ distance with a margin of $\psi = 10$ and scale of $\alpha = 1$, and adaptive distances computed as the absolute difference between latent units.

Experiment	Total	Hyper-Parameters
6.1 Learning To Disentangle (Figures 6.1 and 6.2)	$2 \times 4 \times 4$ $= 32$ $\times 1$ repeats $= 32$	train steps $= 28800$ batch size $= 512$ sampled pairs $(n) = 4096$ lr $\in \{1e^{-3}, 5e^{-4}\}$ kernel radius $\in \{15, 31, 47, 64\}$ dataset $\in$ {XYSquares (8×8), XYSquares (4×4), XYSquares (2×2), XYSquares (1×1)}
We note that the higher learning rate, with larger batch sizes and more sampled pairs often performs better. We do not visualise all these kernels in Section 6.1, however, we used these kernels to perform hyper-parameter sweeps when finding optimal kernels for Table 6.1.		
6.1 Learning To Disentangle (Figure 6.4)	$6 \times 2 \times 4$ $= 48$ $\times 5$ repeats $= 240$	train steps $= 57600$ beta $(\beta) \in \{0.0316, 0.01, 0.00316, 0.001, 0.000316, 0.0001\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents (D) $= 25$ dataset $=$ XYSquares recon. loss $\in$ {MSE, MSE-Gau, MSE-Box, MSE-XY8} kernel radius $= 31$
We only plot runs in Figure 6.4 that obtain low values for the reconstruction loss. This indicates that the models have successfully learnt the data and are not in a local-minima; often this occurs when incorrect values for β are chosen, or the weighting of the kernels is incorrect. We find that for MSE $\beta \in \{0.0001, 0.000316\}$, for MSE-Gau $\beta \in \{0.001, 0.00316\}$, for MSE-Box and MSE-XY8 $\beta \in \{0.01, 0.0316\}$.		

Table A.6: Grid search hyper-parameters for experiments from Chapter 6.

Experiment	Total	Hyper-Parameters
7.2 Choice Of Ground-Truth Factors (Figure 7.3)	$4 \times 2 \times 2$ $= 16$ $\times 3$ repeats $= 48$	train steps $= 57600$ beta $(\beta) \in \{0.0316, 0.01, 0.00316, 0.001\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents $(\mathrm{D}) = 9$ dataset $\in \{$XYObject, XYObject (Shaded)$\}$
7.3 Random External Factors (Figure 7.6)	$3 \times 2 \times 2 \times 5$ $= 60$ $\times 1$ repeats $= 60$	train steps $= 57600$ beta $(\beta) \in \{0.0316, 0.01, 0.00316\}$ framework $\in \{\beta$-VAE, Ada-GVAE$\}$ latents $(\mathrm{D}) = 9$ dataset $=$ dSprites-ImageNet dataset type $\in \{$Foreground, Background$\}$ dataset visibility $\in \{0, 25, 50, 75, 100\}$

Table A.7: Grid search hyper-parameters for experiments from Chapter 8.