Robustness, Structure and Hierarchy in Deep Generative Models

Matthew Willetts

Magdalen College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary 2021

For Olivia

Acknowledgements

This thesis would not have been possible without wonderful people and institutions. First I thank my supervisors, Chris Holmes and Steve Roberts. Both have been peerless advisors. Key parts of this thesis were prompted by simple one-sentence questions asked in our supervisions. Without their support and guidance, none of this would have been possible. I cannot overstate my gratitude.

I thank Yeh Whye Teh and José Miguel Hernández-Lobato for examining this thesis and giving helpful feedback.

Thanks also to Brooks Paige and Tom Rainforth for mentoring me. The clarity of thought and deep insights that they bring to any problem put in front of them has been inspiring, as has their endless creativity.

Next, I must thank my collaborator Alexander Camuto. At times, down in London at the Alan Turing Institute, it seemed that the two of us were a mini research group of our own. The marathon days of chalk on blackboard; of coding; of writing, were some of the happiest of my life.

I also thank Xenia Miscouridou. Though we were both students at the Stats Department, it was only when we were both in London at the Alan Turing Institute that we got to know each other and ended up collaborating on a project together, for which I am grateful.

In Oxford, at the Machine Learning Research Group, I was lucky to be surrounded by wonderfully hardworking and insightful graduate students. I thank Arno Blaas, Adam Cobb, Jonathan Downing, Richard Everett, Logan Graham and Kyriakos Polymenakos for the great conversations over lunch and over endless cups of coffee.

The work in this thesis also needed institutional backing; many of the projects required quite a large amount of compute, mostly using GPUs. In this I was extremely lucky to have the help and support of the research engineering team at the Alan Turing Institute. I thank Oscar Giles, Jim Madge and Tomas Lazauskas for the time and care they put into making sure our experiments were able to run.

It is always a pleasure to discuss ideas with my old friends Eric Hambro, Raza Habib, and Toby Smithe, who all, like me, got the bug for Bayesian statistics, deep learning, and their combination. Those conversations have, without fail, given me fresh insights, for which I am always grateful.

I'd also like to thank my dear friends at Oxford, in London and elsewhere for abiding me; James, Michael, Edward, Elli, James, Maddy, Jane Darby, Will, Hugh, Rose, Paul, Kath, Tom, Maxine, Leaf, Stephen, Xavier, Nico, Felix, Pal, Philip and many others, thank you.

Thanks to my family, my parents David and Sarah and my sister Imogen. You always had my back, and always encouraged me. Finally, I thank my wife Olivia for her kindness and support as I carried out this research.

Abstract

Deep learning provides us with ever-more-sophisticated neural networks that can be tuned via gradient ascent to maximise some objective. Bayesian statistics provides us with a principled and unified approach to specify statistical models and to perform inference. One productive way to pair these two methodologies results in *Deep Generative Models* (DGMs), where the mappings between the statistical parameters in a probabilistic model are themselves parameterised using neural networks. In this thesis we study both the ways in which this approach can be brought to bear on various problems in machine learning and the properties of the resulting models. There are three recurrent themes in this thesis, *robustness*, *structure* and *hierarchy*, that reemerge throughout.

We begin by studying how we can structure a deep generative model to learn in a novel learning regime we propose called *semi*-un*supervised* learning. It is an extreme case of semi-supervised learning, where for some classes of data there are no labelled examples given whatsoever. In learning to partition the data into distinct components, different ground truth classes, the model must be able to cluster over the unlabelled classes and perform semi-supervised learning over the classes for which some labelled examples are given. We show how this can be done for a range of standard datasets.

From handling one discrete latent, cluster assignment, we then study models with a hierarchy of discrete latent variables. We propose a novel way to parameterise latent variables in models of this type, *relaxed responsibility vector quantisation* that enables training of very deep hierarchies of layer of latent variables. This method achieves state-of-the-art results at maximising a lower bound on the evidence of the data (train and test set) for hierarchical discrete DGMs trained end-to-end, on a range of standard datasets. In doing so, these models help to close the gap between hierarchical DGMs with discrete latents and those with continuous latents, as well as providing extremely stable training.

We then swap to a different problem, how to structure a model to effectively learn statistically independent latent representations from high dimensional data. We propose a hierarchical approach, where we use a bijective function, a flow, to produce an intermediate representation that a highly-constrained linear Independent

Component Analysis (ICA) model then acts on. This leads to superior performance on various toy and real datasets compared to other approaches.

We then study the hitherto-unconsidered problem of how to render DGMs robust to adversarial attack. We demonstrate that regularising the latent space of these models can reliably induce robustness, and obtain even more robust models by applying this regularisation to hierarchical DGMs. Finally, we then study the problem of DGM robustness from a theoretical standpoint. We define r-robustness, a novel criterion for DGM robustness, and then derive a margin on that criterion within which a model can be said to be robust. Combined with new theory about the optimal model for a variety of DGM where the latent space is regularised, the form of this margin sheds insight into how this regularisation increases robustness.

The work presented in this thesis shows how productive the combination of deep learning and Bayesian statistics can be, as well as providing insights into the nature of the models made by their combination. This opens up new research in both of these directions—for new models that build on the work presented and also new avenues for theoretical work studying deep generative models.

Contents

List of Figures

This thesis is submitted in integrated format and consists of seven chapters, the middle five of which are each based on a published paper or recent pre-print.

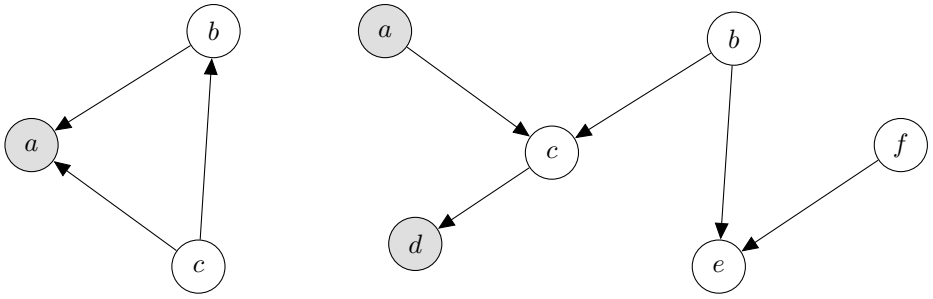
Nothing takes place in the world whose meaning is not that of some maximum or minimum.

— Leonhard Euler

1 Introduction

Contents

The long-term year-on-year increases in the quantity of compute power one can buy for a dollar leads to increases in the sophistication of the computer programs we can feasibly run (Nordhaus, 2007; Hinton, 2018; Sutton, 2019). In tandem, we have also seen increases in the amount of data stored digitally, making it available to us for high-speed analysis. Two modelling paradigms have been infamous for the amount of computation they require: Bayesian statistics & deep learning.



(a) $p(a|b,c)p(b|c)p(c)$ **(b)** $p(a)p(b)p(c|a,b)p(d|c)p(e|b,f)p(f)$

Figure 1.1: Two Bayesian Network each corresponding to a particular joint distribution. In a) a is observed, in b) d and e are observed.

Bayesian statistics As David MacKay said, "you cannot do inference without making assumptions" (MacKay, 2005, §2.2). In Bayesian statistics our aim is to use the information we have, measurements we have recorded and assumptions we make about the structure of the world. We want to use these to calculate probability distributions over quantities that are, in some sense, explanatory of the data we have but that we did not observe. We wish to do this following the rules of logic and probability. Bayesian statistics is thus a continuation of probability theory, using it to build statistical models that we then perform inference on (Jaynes, 2002).

The distributions we wish to find, probability distributions over unobserved quantities given observations, are called *posterior distributions* (historically known as *inverse probabilities*). Obtaining posterior distributions can require evaluating very challenging sums or integrals, or a large number of them, or both. These computational challenges necessitate approximate methods of Bayesian inference for any but a lucky subset of problems.

The fundamental object in probabilistic modelling is the joint probability. Informally, it is the the function that returns the probability that the variables in the model take given values. When students of probability are enjoined by Stephen Gull to "Always write down the probability of everything", it is the joint probability being referred to (MacKay, 2005, §3.6). In general any probability distribution over any number of variables can be mechanically expanded into a product of distributions, by repeated application of the Product rule (also known as the chain

rule). This decomposes any joint distribution into an *auto-regressive* form. For three variables one such decomposition is

$$p(a, b, c) = p(a|b, c)p(b, c) = p(a|b, c)p(b|c)p(c).$$

We could have chosen a different order in which to decompose the joint into conditional and prior distributions. When specifying a probabilistic model over variables we are in effect writing down a particular factorisation of the joint over our variables for which the various constituent distributions take an amenable form. Further, some variables may not depend on some other variables, simplifying the the overall structure. The factorisation of our joint distribution, together with functional forms of the constituent terms, defines a *generative* model, so called as it corresponds to a claim about how the data was formed (Pearl, 1988).

We can represent a joint probability distributions as a *directed acyclic graph* (DAG). Each node with its edges corresponds to a distribution in the joint. It is the *absence* of edges connecting nodes that conveys information here, exactly the same information we get from the removal of conditioning in the factorisation of the joint. In our model we will measure some variables as data; other variables, however, will remain unobserved or *latent*. It is customary in visual representations to shade in observed variables, leaving latent variables unshaded. See Fig 1.1 for two examples of a such factorisations. The first is a factorisation over three variables (as given above) with a observed and all dependencies maintained; the second is more interesting: it is over more variables, some dependencies have been removed, and two variables a & d are observed. Observed variables commonly are leaf nodes in these models, as we are interested in specifying a model that explains how the data we have came to be, but this is not a concrete rule.

Bayesian statistics, in its plain and exact form, reduces to performing certain required integrals and sums. But what are these calculations, and what are the rules for obtaining them? The sums and integrals we have to calculate are given to us by mechanical manipulation of the joint distribution using the rules for the

manipulation of probability distributions. If these integrals and sums were easy to carry out, Bayesian statistics would be a very simple discipline. This link, between the simplest approaches to exact inference and the calculation of integrals (or sums) is seen from Bayes' Rule itself.

Bayes' Rule can be obtained from the two different ways we can factorise the joint distribution of two variables,

$$p(a, b) = p(a|b)p(b) = p(b|a)p(a) \Rightarrow p(b|a) = \frac{p(a|b)p(b)}{p(a)}.$$

This means we can swap from probabilities of a given b to probabilities of b given a. Say our generative model is for observed data a and latent b with $p(a, b) = p(a|b)p(b)$, and we know the functional form of $p(a|b)$ and $p(b)$. We can now calculate the posterior $p(b|a)$ using Bayes' Rule. But what is $p(a)$ here? We can obtain it by *marginalising out* b (an application of the Sum rule),[1]

$$p(a) = \int \mathrm{d}b\, p(a, b).$$

That covers the simplest possible case of one latent and one observed variable. In general, performing inference, obtaining posteriors for the latent variables given the observed variables, for even simple models can be computationally challenging, in two senses. First, if the distributions that define the model are not sufficiently simple then closed form solutions may not exist. This problem can emerge in models with even the simplest graphical structure if the constituent distributions in the generative model have a complicated non-linear form. Secondly, even if the model is analytically tractable, exact methods may be inapplicable for large models or large number of observations as their run-time may be exponentially-scaling in the number of variables (MacKay, 2005, §26.1).

[1] We have assumed b is continuous, that $\mathrm{d}\mu(b) \to \mathrm{d}b$ exists such that we have a measure on the domain of b. If it is was discrete this would be a sum instead.

Deep Learning The discipline of deep learning provides us with *neural networks*. These are a rich class of parametric models made by stacking simple parametric functions (multiplication of an input vector by a learnt matrix, say) interspersed with non-linearities (a rectifier, say). Training these models corresponds to tuning their parameters to maximise some objective (or minimise a loss, if one is feeling pessimistic). This optimisation is commonly done using differential calculus, and when we apply the chain rule to parametric functions of this sort we call it *backpropagation*.

These models can be extremely large. Cutting-edge language models can have hundreds of billions of parameters (Brown et al., 2020). However, deep learning finds a neat match in computer hardware that makes training these models easier than it would otherwise be. Deep learning models are almost always trained using gradient-based optimisation on *mini-batches* (commonly just called *batches*) of data at a time. This makes training inherently parallel within each batch. Graphical processing units (GPUs), developed to solved the massively-parallel arithmetic that emerges in rendering computer-generated scenes, are also well suited to the training of deep learning, as it can be cast in similar mathematical form. This interlocking of model and computer hardware has lead to a renaissance of neural network approaches in the last eight or so years (Hinton, 2018).

Deep Generative Models These two modelling paradigms can be combined: we can embed neural networks inside (often quite simple) Bayesian networks. The statistical parameters of the conditional probabilities within the graphs, a Gaussian distribution, say, with mean and co-variance conditioned on the values of its parents, are now themselves the output of neural networks that take in those parental values as inputs. We have *parameterised the parameters* of our generative model. The resulting structures we get from combining deep learning with generative models are called Deep Generative Models (DGMs), and are the principle object of study in this thesis.

Given such a model we can perform ancestral sampling to produce realistic-looking 'synthetic data'. We will, though, have to find good parameter settings for the neural networks inside our DGM for that to be effective. Relatedly, we will wish to find posterior distributions for the unobserved nodes within our graph, using some appropriate, likely approximate, method of inference. In this thesis we make use of a type of approximate inference called amortised stochastic variational inference, which we briefly review later in this introductory chapter. In that approach, the (approximate) posterior is implemented using another Bayesian network, specifying the chosen posterior dependence between variables. We embed neural networks in that graph, much as we do in the generative model, again parameterising the parameters of the probability distributions present.

This method of inference has the effect of changing the problem of inference from being one of calculating integrals and sums to being one of performing optimisation. Informally, our aim when training our model is to tune the two halves of the model, the generative model and the approximate posterior, to make the data as likely as possible under the model—we will define this more rigorously later this chapter as we introduce these ideas in more detail.

Broad swathes of machine learning can be cast in this framework, combining the principled and consistent nature of probabilistic modelling with the rich parametric functions we obtain from deep learning. The general recipe for a deep generative model is thus: firstly, specify a graph that corresponds to a factorisation and probabilistic form of the generative (i.e., forward) joint distribution over observed and latent variables; secondly, chose how to implement the conditional probabilities in that graph using neural networks; then do the same for the approximate posterior, choosing a factorisation and neural network implementation; and finally train both halves of the model using the data we have.

The simplest deep generative model is a Variational Autoencoder (VAE), which we will introduce in detail in this chapter. Roughly speaking, we have a vector-valued continuous latent variable that is mapped to some observation model for our data

by a neural network. Our approximate posterior similarly maps each datapoint to the parameters of its posterior distribution in the latent space, again performing this mapping using a neural network. We can consider this model as having an auto-encoder like structure. For a well-trained model of this type, we can map a given datapoint to a posterior that is concentrated in a region of the latent space; if we push samples from that posterior through the generative model we'd hope to get samples that look like the original input. If the latent variable is of a lower dimensionality that the data, we are in effect bottle-necking our data through the latent representation. Thus we have a way of learning parsimonious representations our data, which can be useful for many tasks such as, for example, text classification (Xu et al., 2017), drug discovery (Kusner et al., 2017), image compression (Gregor et al., 2016; Theis et al., 2017; Townsend et al., 2019), and for perception within reinforcement learning systems (Ha & Schmidhuber, 2018; Higgins et al., 2017b).

1.1 Thesis Outline

In this thesis we attack numerous modelling problems using the framework of deep generative models and provide new experimental and theoretical insights into their behaviour. There are three recurring themes in this thesis, *Robustness*, *Structure* and *Hierarchy*.

In Chapter 2 we are interested in learning discrete structure from our data, class membership. In particular we are interested in a novel learning regime, an extreme variety of semi-supervised learning, where some ground truth classes of data are found only in the unlabelled dataset—not a single labelled exemplar is given for some classes, only unlabelled examples. Can we structure a model that is robust to this extreme sparsity in the training data? The model must learn to discover classes of data in the unlabelled data while also learning the semi-supervised classes. We call this learning regime semi-*un*supervised learning, and show that a deep Gaussian mixture model can be used to successfully learn in this regime for a range of standard machine learning datasets.

In that chapter we have just one discrete variable, class membership. Learning deep generative models with numerous discrete variables, arranged in a hierarchy of layers, remains an open research problem (Liévin et al., 2019). In Chapter 3, taking inspiration both from recent work in the space of deep generative models and from classical methods of inference, we propose a novel variety of hierarchical discrete variational autoencoder. We build on ideas from how one performs inference in Gaussian mixture models to propose *relaxed-responsibility vector quantisation*, which forms the probabilistic building-block for our model. This structure enables stable training of very deep hierarchies of discrete latent variables, up to 32 in our experiments, with good resulting performance. These models make significant progress in closing the gap between deep generative models with discrete latents and those with continuous latent variables.

From a model with a very large number of latent variables, in the following chapter, Chapter 4, we change gear to considering a model with a single highly compressed latent variable, but one that is highly structured in numerous ways. Our aim is to learn a small number of statistically independent latent variables that explain our data. This is the problem of non-linear non-square Independent Component Analysis (ICA) (Hyvärinen et al., 2001). We propose a novel approach, using an invertible function (implemented using flows (Papamakarios et al., 2019)) to map the given data into a space where performing linear ICA on the resulting representations is easier. Further, the part of the linear ICA model that acts on this learnt representation, the unmixing matrix, is fixed, *not* learnt. We find that this approach performs better on natural image data than VAEs with ICA-appropriate prior distributions on their latents, and offers a new set of approaches for non-linear non-square ICA.

In the final two substantive chapters we study in detail the robustness of deep generative models to adversarial attack. In these attacks an agent adds some (ideally small) distortion to the input image with the aim of fooling the model into reconstructing the distorted input to a chosen target image. In Chapter 5 we show that methods that tune the noisiness of the latent representations of VAEs can

reliably induce robustness to a range of adversarial attacks. Further, we propose a hierarchical VAE with tuned latent representations that is more robust still. In Chapter 6, having established these empirical aspects of VAE robustness, we look at this problem through a more theoretical lens. We give a mathematical framework for understanding VAE robustness, and derive a bound on a margin in data-space within which a VAE can be said to be robust. This margin connects back to methods for tuning the noise in the latent space. We show how manipulating the latent space via the simplest tuning method available changes the constituent terms in the derived form of the margin so as to increase the value of the margin. Further, the derived margin is strongly correlated with our empirical evaluation of the true margin, making it potentially useful as an effective metric for VAE robustness.

Before we launch into these ideas in depth, we will briefly introduce latent variable models and outline the challenges of Bayesian inference for them, how Variational Inference (VI) is a useful method for approximate inference in these models, and then describe how deep generative models emerge naturally from these considerations.

1.2 Latent Variable Models

1.2.1 A Simple Class of Latent Variable Models

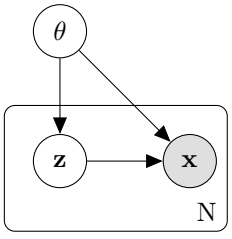


Figure 1.2: A simple class of latent variable models, $p(\mathbf{x}, \mathbf{z}, \theta) = p(\theta)p(\mathbf{z}|\theta)p(\mathbf{x}|\mathbf{z}, \theta)$, represented as a DAG. As discussed, the absence of an arrow linking two nodes indicates conditional independence. In these plate diagrams, the plates represent repetition: we have N pairs of $\mathbf{x}, \mathbf{z}$ variables, one per observation, but only one θ variable as it is off-plate.

We have unlabelled data $\mathcal{D} = \{\mathbf{x}_i\}, i \in 1, ..., N$. We model $\mathbf{x} \in \mathcal{X}$ ($\mathbb{R}^{d_\mathbf{x}}$, say) as having been drawn from a generative model that depends on latent variables. We split the latent variables into those parts that effect the entire model, the

global hidden variables $\theta \in \Theta$; and the remaining *local* hidden variables $\mathbf{Z} = \{\mathbf{z}_i\}, i \in 1, ..., N$, where $\mathbf{z} \in \mathcal{Z}$ ($\mathbb{R}^{d_\mathbf{z}}$, say). Each $\mathbf{z}_i$ contains the local latent variables for the i^{th} datapoint.

A generative model is defined by our choice of the factorisation of the joint $p(\mathbf{x}_i, \mathbf{z}_i, \theta)$ and our choice of the functional form of the individual terms. For this class of models, the joint probability for one data point is

$$p(\mathbf{x}_i, \mathbf{z}_i, \theta) = p(\theta)p(\mathbf{x}_i, \mathbf{z}_i|\theta) \tag{1.1}$$

$$p(\mathbf{x}_i, \mathbf{z}_i|\theta) = p(\mathbf{x}_i|\mathbf{z}_i, \theta)p(\mathbf{z}_i|\theta) \tag{1.2}$$

and for all our data

$$p(\mathcal{D}, \mathbf{Z}, \theta) = p(\theta) \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{z}_i, \theta)p(\mathbf{z}_i|\theta). \tag{1.3}$$

See Figure 1.2 for a graphical representation of the above generative model as a directed acyclic graph, a Bayesian network. To get the posterior for our latent variables given our data, we can apply Bayes' Rule. This means that we would have to find the evidence under our model [2],

$$p(\mathcal{D}) = \int \mathrm{d}\theta \prod_{i=1}^{N} \int \mathrm{d}\mathbf{z}_i \; p(\mathbf{x}_i, \mathbf{z}_i, \theta). \tag{1.4}$$

From this we can then calculate the posterior distributions over our latent variables,

$$p(\mathbf{Z}, \theta|\mathcal{D}) = \frac{p(\theta) \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{z}_i, \theta)p(\mathbf{z}_i)}{p(\mathcal{D})}. \tag{1.5}$$

1.2.2 Treating θ as a Parameter

Sometimes it is not practical to marginalise over θ: the integrals can be too computationally challenging (MacKay, 2005, §29.1). For many cases in the class of models we will consider in this thesis, deep generative models, it is not currently feasible to give a fully Bayesian treatment to θ, though this remains an active area of research. Instead one might aim to find the maximum likelihood (ML) estimate for θ,

[2] For the sake of simplicity of notation we are going to treat θ and $\mathbf{z}$ as continuous, so as before we will be taking integrals rather than sums.

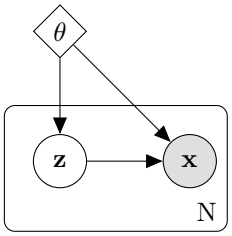


Figure 1.3: Not giving θ a full Bayesian treatment in our model but instead treating it as a deterministic parameter, as indicated by the diamond around θ. $p_\theta(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}, \mathbf{z}; \theta) = p_\theta(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z})$.

maximising the likelihood of our data $\mathcal{D}$ under our generative model. Thus we view θ as simply some parameters to optimise. To indicate when we are treating variables as parameters, we will place them in the subscript of probability distributions, for example $p_\theta(\mathcal{D}|\mathbf{Z})$ is $p(\mathcal{D}|\mathbf{Z}; \theta)$. For most of our following discussion and in our models we treat θ in this way. Thus we have a family of joint distributions for data and our local latent variables:

$$p_\theta(\mathcal{D}, \mathbf{Z}) = p_\theta(\mathcal{D}|\mathbf{Z})p_\theta(\mathbf{Z}),\ \theta \in \Theta. \tag{1.6}$$

Under models of this form, we have two questions to ask:

1. What is the maximum likelihood estimate for θ for our model given data $\mathcal{D}$?

$$\theta^* = \arg\max_{\theta \in \Theta} p_\theta(\mathcal{D}) = \arg\max_{\theta \in \Theta} \int \mathrm{d}\mathbf{Z}\, p_\theta(\mathcal{D}|\mathbf{Z})p_\theta(\mathbf{Z}) \tag{1.7}$$

2. With a particular model and setting of θ, what is the posterior distribution over the latent variables given our data?

Both of these steps seem to require calculating the evidence, for which exact evaluation is often intractable: either analytically intractable or because the calculation becomes computationally infeasible as the amount of data increases, or both. This can be true even for simple models (Blei et al., 2017).

1.2.3 Separating the Model from the Learning Paradigm

Before we go further, there is an important separation to highlight: the separation between model and learning paradigm. Inference in latent variable models can be

attacked with a wide range of approximate methods — Laplace approximations, Approximate Bayesian Computation, Variational Inference, Bayesian Quadrature, Monte Carlo methods, and others (Bishop, 2006; Murphy, 2012) — without changing the generative model. Some approaches suit some models better than others. It is only when we combine a generative model with a learning principle that we get a complete learning algorithm. In the next section we introduce variational methods, as they are a productive and natural pairing to the generative models we are investigating, though of course inference in the generative models we discuss could be attacked using other approaches. For the rest of this thesis, with one exception where a model is in part trainable using pure maximum likelihood, all inference will be done using variational methods.

1.3 Variational Inference

In variational inference (VI) we turn our problem of inference, often requiring us to evaluate challenging sums and integrals, to one purely of optimisation.

There are two views available for deriving the equations of VI. The first is associated with finding the evidence of our data via a lower bound proxy. The second comes from attempting to find an approximation to the true posterior. We will view these in turn, but both lead to the same key equations.

After that, we will introduce mean-field VI, the reparameterisation trick and then amortised inference, which between them form the bedrock of the methods and models we will propose and study in subsequent chapters.

1.3.1 Bounding the Evidence

As discussed above, in order to calculate the evidence exactly for our model over our dataset $\mathcal{D}$, for a given setting of θ, we have to perform a challenging integral, marginalising out $\mathbf{Z}$:

$$p_\theta(\mathcal{D}) = \mathbb{E}_{p_\theta(\mathbf{Z})}\, p_\theta(\mathcal{D}|\mathbf{Z}) = \int \mathrm{d}\mathbf{Z}\, p_\theta(\mathcal{D}, \mathbf{Z}). \tag{1.8}$$

If we wanted to approximate this integral using Monte Carlo (MC) techniques, the simplest approach would be to sample M times from the prior $p_\theta(\mathbf{Z})$ and take a sum:

$$p_\theta(\mathcal{D}) \approx \frac{1}{M} \sum_{m=1}^{M} p_\theta(\mathcal{D}|\mathbf{Z}^m),\ \mathbf{Z}^m \sim p_\theta(\mathbf{Z}). \tag{1.9}$$

However, this sampling procedure is not efficient. To get an accurate and efficient estimator for $p_\theta(\mathcal{D})$ we want to target the regions in $\mathcal{X} \times \mathcal{Z}$ that are associated with large values of our joint density; these are the regions that strongly effect the value of our integral.

So, we can perform importance sampling (Bishop, 2006), re-writing our evidence and multiplying by 1 in disguise,

$$\begin{aligned} p_\theta(\mathcal{D}) = \mathbb{E}_{p_\theta(\mathbf{Z})}\, p_\theta(\mathcal{D}|\mathbf{Z}) &= \int \mathrm{d}\mathbf{Z}\, p_\theta(\mathcal{D}|\mathbf{Z}) p_\theta(\mathbf{Z}) && (1.10)\\ &= \int \mathrm{d}\mathbf{Z}\, p_\theta(\mathcal{D}|\mathbf{Z}) p_\theta(\mathbf{Z}) \cdot \frac{q_\Phi(\mathbf{Z})}{q_\Phi(\mathbf{Z})} && (1.11)\\ &= \mathbb{E}_{q_\Phi(\mathbf{Z})} \frac{p_\theta(\mathcal{D}, \mathbf{Z})}{q_\Phi(\mathbf{Z})}. && (1.12) \end{aligned}$$

We have introduced a *proposal distribution* $q_\Phi(\mathbf{Z})$. Our aim is for it to place more probability mass over likely settings of $\mathbf{Z}$ given our data. $q_\Phi(\mathbf{Z})$ is a member of a family of distributions $\mathscr{Q}$. For now we will not be specific about the functional form of $q_\Phi(\mathbf{Z})$ – it is just some parameterised probability distribution for $\mathbf{Z}$ with parameters Φ. The optimal proposal distribution $q_\Phi(\mathbf{Z})$ would be proportional to the true posterior $p_\theta(\mathbf{Z}|\mathcal{D})$ (Owen, 2013).

We can expand the log evidence and apply Jensen's Inequality:

$$\begin{aligned} \log p_\theta(\mathcal{D}) &= \log\Big(\mathbb{E}_{q_\Phi(\mathbf{Z})}\Big[\frac{p_\theta(\mathcal{D}|\mathbf{Z}) p_\theta(\mathbf{Z})}{q_\Phi(\mathbf{Z})}\Big]\Big) && (1.13)\\ &\geq \mathbb{E}_{q_\Phi(\mathbf{Z})}\Big[\log\Big(\frac{p_\theta(\mathcal{D}|\mathbf{Z}) p_\theta(\mathbf{Z})}{q_\Phi(\mathbf{Z})}\Big)\Big] && (1.14)\\ = \mathcal{L}(\mathcal{D}; \theta, \Phi) = \mathbb{E}_{q_\Phi(\mathbf{Z})}\Big[\log p_\theta(\mathcal{D}|\mathbf{Z})\Big] &- D_{\mathrm{KL}}(q_\Phi(\mathbf{Z})||p_\theta(\mathbf{Z})) && (1.15) \end{aligned}$$

Where $D_{\mathrm{KL}}(\cdot||\cdot)$ is the Kullback–Leibler divergence (Kullback & Leibler, 1951). This gives us $\mathcal{L}(\mathcal{D}; \theta, \Phi)$, the Evidence Lower Bound or ELBO.

The size of the *variational gap* between the model evidence $\log p_\theta(\mathcal{D})$ and $\mathcal{L}$ depends on the choice of $q_\Phi(\mathbf{Z})$, both its functional form, determined by $\mathscr{Q}$, and the particular setting of Φ. To answer the first question of Section 1.2.2, we would now aim to make the variational gap as small as possible by tuning Φ. We can do this jointly with tuning θ to find its ML estimate. Thus we aim to maximise Eq (1.15) w.r.t. $\{\theta, \Phi\}$, and so we have turned our problem of inference into one of optimisation.

1.3.2 Minimising a KL divergence

So an alternate approach is to try to find an approximate posterior for $\mathbf{Z}$ directly. Trying to obtain the true posterior $p_\theta(\mathbf{Z}|\mathcal{D})$ is exactly as challenging as calculating the evidence: the latter is the denominator we need in Bayes' Rule.

As above, we pick some family of distributions $\mathscr{Q}$, the members of which $q_\Phi(\mathbf{Z})$ have parameters Φ. We then wish to find the member of that family which most closely approximates the true posterior, given our data and our generative model. If we choose to minimise the Kullback-Leibler (KL) divergence between $q_\Phi(\mathbf{Z})$ and $p_\theta(\mathbf{Z}|\mathcal{D})$, we will be doing variational inference.

We aim to find

$$q^*(\mathbf{Z}) = \underset{q_\Phi(\mathbf{Z})\in\mathscr{Q}}{\arg\min}\, D_{\mathrm{KL}}(q_\Phi(\mathbf{Z})||p_\theta(\mathbf{Z}|\mathcal{D})). \tag{1.16}$$

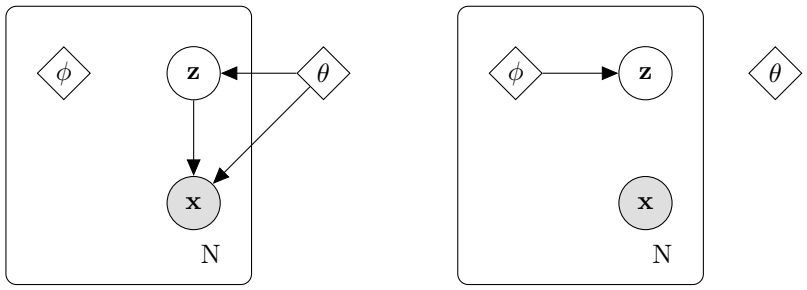
We cannot directly optimise the D_{KL} divergence as it requires us to have access to $p_\theta(\mathbf{Z}|\mathcal{D})$. So, let us expand it out (Mackay, 1995; Bishop, 2006):

$$\begin{aligned} D_{\mathrm{KL}}(q_\Phi(\mathbf{Z})||p_\theta(\mathbf{Z}|\mathcal{D})) &= \mathbb{E}_{q_\Phi(\mathbf{Z})}\left[\log q_\Phi(\mathbf{Z})\right] - \mathbb{E}_{q_\Phi(\mathbf{Z})}\left[\log p_\theta(\mathbf{Z}|\mathcal{D})\right] && (1.17)\\ &= \mathbb{E}_{q_\Phi(\mathbf{Z})}\left[\log q_\Phi(\mathbf{Z})\right] - \mathbb{E}_{q_\Phi(\mathbf{Z})}\left[\log p_\theta(\mathbf{Z},\mathcal{D})\right] + \log p_\theta(\mathcal{D}) && (1.18)\\ &= -\mathcal{L}(\mathcal{D};\theta,\Phi) + \log p_\theta(\mathcal{D}) && (1.19) \end{aligned}$$

And now we have found the discrepancy from applying Jensen's inequality in the previous section; as we might expect the ELBO matches the log evidence for a given θ when we perfectly minimise the KL in Eq (1.17), when $q_\Phi(\mathbf{Z}) = p_\theta(\mathbf{Z}|\mathcal{D})$.

As $\log p_\theta(\mathcal{D})$ does not depend on Φ, minimising $D_{\mathrm{KL}}(q_\Phi(\mathbf{Z})||p_\theta(\mathbf{Z}|\mathcal{D}))$ w.r.t. Φ is the same as maximising $\mathcal{L}(\mathcal{D};\theta,\Phi)$ w.r.t. Φ. So by maximising $\mathcal{L}$ w.r.t. $\{\theta,\Phi\}$ we are jointly doing ML maximisation of the evidence while finding the best approximate posterior with parameters Φ, given the current ML estimate for θ.

1.3.3 Mean-Field and Stochastic Variational Inference:



(a) Generative Model **(b)** Approximate Posterior

Figure 1.4: Representation of our model and its approximate posterior for mean-field VI for $\mathbf{z}$ (applying MLE for θ) as a probabilistic graphical model. Figure (a) shows the generative model $p_\theta(\mathcal{D},\mathbf{Z}) = \prod_{i=1}^N p_\theta(\mathbf{x}_i|\mathbf{z}_i)p_\theta(\mathbf{z}_i)$. Figure (b) shows the classical approximate posterior $q_\Phi(\mathbf{Z}) = \prod_{i=1}^N q_{\phi_i}(\mathbf{z}_i)$. When training we jointly learn θ and ϕ.

As our generative model is identically and independently distributed (i.i.d.) over our datapoints, we can decompose the total log evidence as a sum of those for our individual datapoints $\mathbf{x}$:

$$\log p_\theta(\mathcal{D}) = \sum_{i=1}^N \log \int \mathrm{d}\mathbf{z}_i \, p_\theta(\mathbf{x}_i, \mathbf{z}_i) \tag{1.20}$$

We will tend to focus on the per-datapoint formulation going forward, dropping subscripts,

$$p_\theta(\mathbf{x}) = \int \mathrm{d}\mathbf{z} \, p_\theta(\mathbf{x}, \mathbf{z}). \tag{1.21}$$

As we have an i.i.d. model, we can repeat the above analysis at the datapoint level. Commonly use the per-datapoint i.i.d. formulation of statistical quantities. When we are interested in whole-dataset values, we can take MC samples over the dataset to give us an unbiased estimator. This is minibatch sampling, as mentioned in the context of deep learning, and is particularly useful when we have large datasets.

In general, in classical approaches to VI we define $q_\Phi(\mathbf{Z})$ by imposing a particular factorisation over latent variables. If we pick an individual $q_\phi(\mathbf{z})$ for each datapoint $\mathbf{x}$, we have made the *mean-field approximation* (Mackay, 1995; Waterhouse et al., 1996; Choudrey, 2000; Beal, 2003): that our overall variational distribution for $\mathbf{Z}$ factorises as

$$q_\Phi(\mathbf{Z}) = \prod_{i=1}^{N} q_{\phi_i}(\mathbf{z}_i). \tag{1.22}$$

Where we have made explicit here that we have different parameters, ϕ_i, for each $\mathbf{x}_i$: $\Phi = \{\phi_i\}$, $i \in 1, ..., N$.

We then obtain the per datapoint ELBO, dropping i subscripts,

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z})}\Big[\log p_\theta(\mathbf{x}|\mathbf{z})\Big] - D_{\mathrm{KL}}(q_\phi(\mathbf{z})||p_\theta(\mathbf{z})), \tag{1.23}$$

so the dataset ELBO is the sum of each $\mathcal{L}(\mathbf{x}_i; \theta, \phi_i)$:

$$\mathcal{L}(\mathcal{D}; \theta, \Phi) = \sum_{i=1}^{N} \mathcal{L}(\mathbf{x}_i; \theta, \phi_i) \tag{1.24}$$

See Figure 1.4 for a plate-diagram showing the mean-field approximate posterior.

For mean-field VI applied to conjugate exponential family models, we can obtain both the optimal functional form of our variational distributions and closed-form updates for $\Phi = \{\phi_i\}$ given a forward model (Bishop, 2006). We can also treat θ on an equal Bayesian footing by introducing for it a variational distribution $q_\lambda(\theta)$ if we so wish. Mean-field VI requires a full pass over the dataset for one EM-like update (Bishop, 2006). Stochastic variational inference (Hoffman et al., 2012) remedies this by performing batched stochastic gradient descent on $\mathcal{L}$, but still requires our models to be conjugate exponential.

1.3.4 The Reparameterisation Trick

The Reparameterisation Trick (Kingma & Welling, 2014) [also known as the pathwise derivative estimator (Fu, 2006), stochastic backpropagation (Rezende et al., 2014) and doubly stochastic estimation (Titsias & Lázaro-Gredilla, 2014)] renders the

differentiation of the integrals inside $\mathcal{L}$ w.r.t. the model parameters θ, ϕ tractable by performing a change of variables inside $\mathcal{L}$. This approach does not require us to use the functional form for the posterior as given by mean-field VI, as vanilla SVI does, but we do have to fulfill certain requirements about differentiability.

The idea is that we find a deterministic, differentiable function $\mathbf{z} = g(\boldsymbol{\epsilon}; \phi)$ such that $\mathbf{z} \sim q_\phi(\mathbf{z})$, where $\boldsymbol{\epsilon}$ is a draw from some base distribution $p(\boldsymbol{\epsilon})$. Thus, by taking gradients of the function g, our samples from $q_\phi(\mathbf{z})$ are differentiable w.r.t. to the parameters of the distribution ϕ. Consider $q_\phi(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. If $p(\boldsymbol{\epsilon})$ is an isotropic unit-variance Gaussian distribution, then

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|0, \mathbb{I}), \tag{1.25}$$

$$\mathbf{z} = g(\boldsymbol{\epsilon}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{1/2}\boldsymbol{\epsilon}, \tag{1.26}$$

$$\mathbf{z} \sim \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{1.27}$$

Performing this change of variables we get

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{\boldsymbol{\epsilon}\sim p(\boldsymbol{\epsilon})}\left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z} = g(\boldsymbol{\epsilon}; \phi))}{q_\phi(\mathbf{z} = g(\boldsymbol{\epsilon}; \phi))}\right]. \tag{1.28}$$

All our ϕ dependence when sampling has been moved into our reparameterisation of $\mathbf{z} = g(\boldsymbol{\epsilon}; \phi)$. Differentiating, using the chain rule for partial derivatives for ϕ dependence through $g(\boldsymbol{\epsilon}; \phi)$,

$$\nabla_\phi \mathcal{L}(\mathbf{x}; \theta, \phi) = \int \mathrm{d}\boldsymbol{\epsilon}\, p(\boldsymbol{\epsilon})\Big(\nabla_{\mathbf{z}} \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z})} \nabla_\phi g(\boldsymbol{\epsilon}; \phi)\Big) \tag{1.29}$$

$$= \int \mathrm{d}\boldsymbol{\epsilon}\, p(\boldsymbol{\epsilon}) \nabla_{\mathbf{z}}[\log p_\theta(\mathbf{x}, \mathbf{z}) + \log q_\phi(\mathbf{z}|\mathbf{x})] \nabla_\phi g(\boldsymbol{\epsilon}; \phi). \tag{1.30}$$

Similarly for θ,

$$\nabla_\theta \mathcal{L}(\mathbf{x}; \theta, \phi) = \int \mathrm{d}\boldsymbol{\epsilon}\, p(\boldsymbol{\epsilon})\Big(\nabla_\theta \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z})}\Big) = \int \mathrm{d}\boldsymbol{\epsilon}\, p(\boldsymbol{\epsilon})\Big(\nabla_\theta \log p_\theta(\mathbf{x}, g(\boldsymbol{\epsilon}; \phi))\Big). \tag{1.31}$$

As $p(\boldsymbol{\epsilon})$ is easy to sample from, it is simple to get Monte Carlo estimates of the gradients. Often just one sample from $p(\boldsymbol{\epsilon})$ per datapoint per batch is sufficient (Doersch, 2016). In the language of computational graphs, in the graph representing our model we have now removed the stochastic node from the path between the input and objective nodes, so now we can differentiate end-to-end. To be able to use the reparameterisation trick we must be able to:

- evaluate $\log q_\phi(\mathbf{z}|\mathbf{x})$ and $\log p_\theta(\mathbf{x}, \mathbf{z})$
- compute the gradient of $\log p_\theta(\mathbf{x}, \mathbf{z})$ w.r.t. parameters θ and w.r.t. $\mathbf{z}$
- compute the gradient of $\log q_\phi(\mathbf{z})$ w.r.t. parameters ϕ and w.r.t. $\mathbf{z}$
- find a function $g(\cdot; \phi)$ and base distribution $p(\boldsymbol{\epsilon})$ so $\mathbf{z} = g(\boldsymbol{\epsilon}; \phi), \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}) \leftrightarrow \mathbf{z} \sim q_\phi(\mathbf{z})$ where we can differentiate g w.r.t. ϕ.

1.3.5 Amortised Inference

When we perform our updates for $\Phi = \{\phi_i\}$ we are trying to find the setting of Φ that minimises the variational gap, to minimise the difference between our lower bound and the value of $p_\theta(\mathcal{D})$ for our current θ setting.

However, there are two downsides to the mean-field approach. Firstly, we have to keep track of and update separately as many approximate posteriors as we have data points. This is potentially a problem for large datasets. And secondly, and more importantly, if we come along with a new data point and ask for its posterior, we have to repeat the process from scratch.

An alternative is to cast the learning of our set of latent parameters $\Phi = \{\phi_i\}$ as a regression problem: we wish to predict ϕ as a function of $\mathbf{x}$. This is *amortising* our inference: we attempt to learn a mapping directly from points in $\mathcal{X}$ to settings of ϕ. We need this mapping to be rich — it must have the capacity to given a good posterior for each of our different $\mathbf{x} \sim \mathcal{D}$ in our dataset — as well as (we hope) generalising to subsequent datapoints $\mathbf{x}^* \sim p(\mathbf{x})$. We can represent this by saying each posterior is now conditioned on its $\mathbf{x}$ value. By changing to have a functional mapping from the space of data to the space of parameters for our variational posterior,

$$q_{\phi_i}(\mathbf{z}_i) \rightarrow q_\phi(\mathbf{z}_i|\mathbf{x}_i), \tag{1.32}$$

we have changed how we treat the parameters ϕ of our variational distribution. In mean-field VI the functional form of $q_{\phi_i}(\mathbf{z}_i)$ and thus of its parameters ϕ_i comes performing calculus of variations given the factorisation chosen (Bishop, 2006). $\{\phi_i\}$ were simply the parameters of those distributions. Now ϕ are the parameters of

our mapping. So all q distribution are now $q(\mathbf{z}_i|f(\mathbf{x}_i;\phi))$ for some shared, rich function $f(\cdot;\phi)$, the outputs of which are the parameters required for a per-datapoint posterior distribution (such as the mean and covariance for a Gaussian distribution). So all per-datapoint variational posteriors use the same function, parameterised by shared, that is, *amortised*, parameters ϕ.

These shared parameters are the driving force of our q distribution, though there has been this subtle change in meaning, so we continue to use ϕ to represent them: $q_\phi(\mathbf{z}_i|\mathbf{x}_i)$ really means $q(\mathbf{z}_i|f(\mathbf{x}_i;\phi))$.

Our ELBO now for one datapoint is

$$\mathcal{L}(\mathbf{x};\theta,\phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\Big[\log p_\theta(\mathbf{x}|\mathbf{z})\Big] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})). \tag{1.33}$$

Our overall dataset $\mathcal{L}(\mathcal{D};\theta,\phi)$ is still the sum of the per-datapoint $\mathcal{L}(\mathbf{x};\theta,\phi,)$, as our overall q still factorises simply: $q_\phi(\mathbf{Z}|\mathcal{D}) = \prod_{i=1}^N q_\phi(\mathbf{z}_i|\mathbf{x}_i)$. Going forward we will continue to treat the i subscript as implicit. $q_\phi(\mathbf{z}|\mathbf{x})$ is often known as the *recognition model*. Even though our latent structure is now simpler, having $\Phi = \phi$, we have the freedom to make $q_\phi(\mathbf{z}|\mathbf{x})$ highly expressive through our freedom in selecting $f(\cdot;\phi)$. As before we can jointly optimise $\mathcal{L}$ w.r.t. θ and ϕ. When we maximise $\mathcal{L}$ using stochastic gradient descent we are requiring that $\mathbf{x}$ is likely under $\log p_\theta(\mathbf{x}|\mathbf{z}=\mathbf{z}^*)$, $\mathbf{z}^* \sim q_\phi(\mathbf{z}|\mathbf{x})$ and that $q_\phi(\mathbf{z}|\mathbf{x})$ is close (in D_{KL} divergence) to our prior for $\mathbf{z}$. This is the ever present combination of reconstruction and regularisation in variational inference.

1.4 Deep Generative Models

Deep learning offers a natural pairing to variational inference. Deep learning models are state-of-the-art for supervised machine learning tasks, can be trained at massive scale thanks to stochastic gradient ascent. Further, because they are trained using gradient ascent they can be combined with other methods also trained via gradient-based optimisation. As VI turns inference into optimisation it can be

paired with deep learning, allowing us to extend approximate Bayesian inference beyond the conjugate-exponential family.

Models made by combining deep learning with latent variable models are known as Deep Generative Models (DGMs) (Rezende et al., 2014). Variational inference is often a highly effective way of learning these models. Although we are going to focus on variational methods, it is important to remember the separation between models and choice of paradigm to training them.

Deep generative models are parametric, the parameters θ in the data likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ in a deep generative model are commonly the parameters of a highly expressive mapping from a latent variable to the parameters of an appropriate distribution given the form of the data. We could say $p_\theta(\mathbf{x}|\mathbf{z}) = p(\mathbf{x}|g(\mathbf{z};\theta))$.

In most of the models we will study or propose in this thesis, the functions $f_\phi(\cdot)$ inside our variational posteriors and those inside our generative models $g_\theta(\cdot)$ are deep neural networks.

With such a wide range of deep learning techniques, architectures and optimisers, our generative models and approximate posteriors can be highly non-linear. Through appropriate choices they can enable avenues of attack for problems, like image analysis, previously beyond the scope of approximate Bayesian inference.

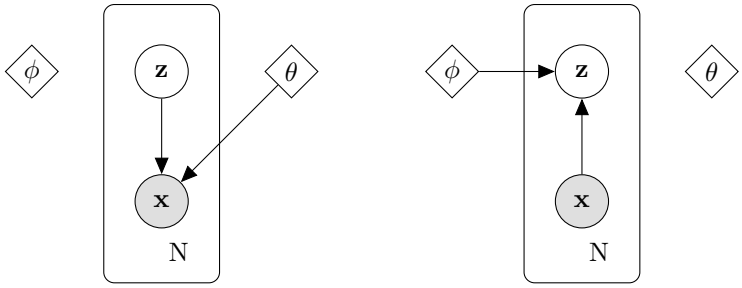
1.4.1 Variational Autoencoders

Now we have the framework in which to set up Variational Autoencoders (VAEs) (Kingma & Welling, 2014), also known as a Deep Latent Gaussian Models (Rezende et al., 2014). For a vanilla VAE, the generative model is:

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z}), \tag{1.34}$$

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbb{I}), \tag{1.35}$$

$$p_\theta(\mathbf{x}|\mathbf{z}) = p(\mathbf{x}|g_\theta(\mathbf{z})), \tag{1.36}$$



(a) Generative Model **(b)** Recognition Model

Figure 1.5: VAE as a probabilistic graphical model. Figure (a) shows the generative model $p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$. Figure (b) shows the variational approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$. When training we jointly learn θ and ϕ. Note that $p(\mathbf{z})$ does not depend on θ at all, even as a parameter.

with $\mathbf{z} \in \mathbb{R}^{d_\mathbf{z}}$. $p(\mathbf{x}|g_\theta(\mathbf{z}))$ is an appropriate distribution given the form of the data, say $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_\theta(\mathbf{z}), \boldsymbol{\Sigma})$ for continuous data (with $\boldsymbol{\Sigma}$ diagonal and often a fixed hyper-parameter) or Bernoulli$(\mathbf{x}|\pi_\theta(\mathbf{z}))$ if our data is binary.

For inference, choose q to be a Gaussian distribution, amortising our inference as above. The parameters of this distribution are themselves parameterised by neural networks:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x})). \tag{1.37}$$

The covariance matrix $\boldsymbol{\Sigma}_\phi(\mathbf{x})$ is commonly chosen to be diagonal. This enables us to define a standard VAE:

Definition 1.1. *(Kingma & Welling, 2014) A standard VAE is a deep generative model with forward model as in Eqs (1.34-1.36) and amortised variational posterior as in Eq* (1.37)*, trained w.r.t. its parameters θ, ϕ to maximise the objective:*

$$\mathcal{L}^{\text{VAE}}(\mathcal{D}; \theta, \phi) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\right]. \tag{1.38}$$

This model fulfils the requirements for using the reparameterisation trick. See Figure 1.5 for a probabilistic graphical model representation. $p_\theta(\mathbf{x}|\mathbf{z})$ is often called the *decoder network* and $q_\phi(\mathbf{z}|\mathbf{x})$ the *encoder network*. The model can be seen as bottle-necking data into a latent space then expanding again.

As touched on in Section 1.3.5, one of the benefits of casting variational inference as a regression problem to be solved by optimisation is that after training we have a mapping from data to approximate mean-field posteriors. Thus the trained encoder and decoder can be used to map new data points into $\mathcal{Z}$ and to map arbitrary points in $\mathcal{Z}$ to their corresponding $p_\theta(\mathbf{x}|\mathbf{z})$, respectively. These useful objects we obtain after training, the networks that represent these mappings and hence their distributions, are examples of *inference artifacts*.

1.5 Contributions

DGMs, including VAEs and their extensions, have been useful to a broad range of tasks. In this thesis we present advances in deep generative models in numerous directions: we demonstrate the use of DGMs in novel learning regimes, we propose new DGMs that learn representations of various types, and we provide novel empirical and theoretical insights into the properties of these models under adversarial attack.

This thesis is submitted in integrated format and consists of seven chapters, the middle five of which are each based on a published paper or recent pre-print. As such, each substantive chapter contains its own literature review. As there is overlap in ideas between chapters, these background sections do tend to build on each other as we go through.

While this thesis covers a broad range of topics within deep generative models, it can be read as a series of variations on the themes of *Robustness*, *Structure* and *Hierarchy*. Here I outline for each chapter the paper it is based on, and for each describe my contribution.

In Chapter 2 I outline a novel learning regime, semi-unsupervised learning, and show how DGMs developed for semi-supervised learning fail on standard datasets—while a variety of DGM appropriate for clustering can be usefully brought to bear on this problem. I envisaged and executed this paper on my own, performing all experiments myself, under the aegis of my supervisors. This chapter is based on the paper:

Willetts, M., Roberts, S. and Holmes, C.,
Semi-Unsupervised Learning: Clustering and Classifying using Ultra-Sparse Labels, IEEE International Conference on Big Data 2020

which itself was an extension of a workshop paper:

Willetts, M., Doherty, A., Roberts, S. and Holmes, C.,
Semi-Unsupervised Learning using Deep Generative Models, NeurIPS Bayesian Deep Learning Workshop 2018

In Chapter 3 I will explore an extension of vector quantisation, relaxed-responsibility vector quantisation, that makes possible the specification and training of very deep hierarchies of discrete latent variables for VAEs. This was work done with Xenia Miscouridou and my supervisors. I contributed the theoretical results, the design of methods, and the formalisation of the mathematical framework, both relaxed-responsibility vector quantisation and its setting within VAEs. I conducted all experiments, including all baselines. This chapter is based on the pre-print:

Willetts, M., Miscouridou, X., Roberts, S. and Holmes, C.,
Relaxed-Responsibility Hierarchical Discrete VAEs, 2020 (arXiv:2007.07307, Under Review)

In Chapter 4 I will develop a new approach for non-linear non-square ICA using invertible feature maps and fixed unmixing matrices. This is joint work with Alexander Camuto, Brooks Paige, Chris Holmes and Stephen Roberts. For this work I am joint first author with Alexander Camuto. Together with Alexander Camuto, Brooks Paige and Steve Roberts I contributed to the design of methods and the formalisation of the mathematical framework. In particular, I contributed the idea of using flows to learn a non-linear feature map for linear ICA to act on, and the idea of using Johnson–Lindenstrauss transforms within the unmixing matrix in the linear-ICA component. Further, I contributed to the design of the implementation and experiments for our model and ran the experiments for all linear models and all the various baselines. This chapter is based on paper:

Camuto, A.*, **Willetts, M.***, Paige, B., Holmes, C., and Roberts, S.,
Learning Bijective Feature Maps for Linear ICA, *to appear in* AISTATS 2021

In Chapter 5 I will demonstrate how one can increase the noisiness of the learnt latent representations in VAEs to increase the robustness of these models to adversarial attack, and how this idea when applied to hierarchical VAEs increases robustness further still. This is joint work with Alexander Camuto, Tom Rainforth, Stephen Roberts and Chris Holmes. For this work I am joint first author with Alexander Camuto. Together with Alexander Camuto and Chris Holmes I contributed to the design of methods and the formalisation of the mathematical framework. In particular, I contributed the idea of using TC-penalisation to increase robustness and of using hierarchical VAEs with regularisation, including the derivations of the objective for our hierarchical model and the sampling method needed during training. Further, I contributed to the implementation of our model and the various empirical investigations. This chapter is based on the paper:

Willetts, M.*, Camuto, A.*, Rainforth, T., Roberts, S, and Holmes, C.,
Improving VAEs' Robustness to Adversarial Attack, *to appear in* ICLR 2021

In Chapter 6 I will demonstrate novel theoretical contributions in the formalisation of and understanding of robustness of VAEs to adversarial attack, backed up by empirical results. This is joint work with Alexander Camuto, Tom Rainforth, Stephen Roberts and Chris Holmes. Together with Alexander Camuto and Tom Rainforth I contributed to the design of methods and the formalisation of the mathematical framework. In particular, I contributed the idea of re-purposing the approach of Cohen et al. (2019) to understand VAE robustness and derived the theoretical results for the optimal posterior of the β-VAE, as well as running various experiments to help elucidate our ideas. This chapter is based on the paper:

Camuto, A., **Willetts, M.**, Roberts, S, Holmes, C., and Rainforth, T.,
Towards a Theoretical Understanding of the Robustness of Variational Autoencoders, *to appear in* AISTATS 2021 *(Oral)*

Additional Projects Over the course of my doctoral research I have had the pleasure to work on other papers and projects in addition to the ones presented here. They are:

Willetts, M., Hollowell, S., Holmes, C., and Doherty, A., *Statistical machine learning of sleep and physical activity phenotypes from sensor data in 96,220 UK Biobank participants*, 2018, Nature Scientific Reports 8-7961

Willetts, M., Doherty, A., Roberts, S. and Holmes, C., *Semi-Unsupervised Learning of Human Activity using Deep Generative Models*, NeurIPS Bayesian ML4Health Workshop 2018

Willetts, M., Roberts, S. and Holmes, C., *Disentangling to Cluster: Gaussian Mixture Variational Ladder Autoencoders*, NeurIPS Bayesian Deep Learning Workshop 2019

Camuto, A., **Willetts, M.**, Şimşekli, U., Roberts, S. and Holmes, C., *Explicit Regularisation in Gaussian Noise Injections*, NeurIPS 2020

Barrett, B., Camuto, A., **Willetts, M.**, Rainforth, T., *Certifiably Robust Variational Autoencoders*, 2021 (arXiv:2102.07559, Under Review)

There are more things in heaven and earth, Horatio,
Than are dreamt of in your philosophy.

— William Shakespeare, *Hamlet*

2

Semi-*Un*supervised Learning

Contents

2.1 Introduction

In the previous chapter we have introduced the broad concepts of probabilistic deep generative models: we embed neural networks inside Bayesian networks and use an appropriate method of inference to obtain posteriors over the latent variables. In this

chapter we explore what happens when we have both a continuous latent variable and a discrete latent (that we hope will become associated with the ground-truth class). We will explore two simple deep generative models of this type, within the context of various learning regimes with differing amounts of label data.

What are the standard learning regimes when handling classes? When learning to classify we can either perform supervised learning, if we have fully labelled data, or semi-supervised learning, if that labelling is sparse. Or we can perform clustering via unsupervised learning if there is no labelled data at all.

Semi-supervised learning is useful as in many problem domains we have only a relatively small amount of labelled data compared to the amount of unlabelled data. Gathering labels often requires expert annotation, which is expensive, whereas unsupervised data can often be obtained by automated methods. Thus, it is common for only a subset of data gathered to be (expensively) labelled.

Within this sparsely-labelled dataset, however, it is possible that there may well be ground truth classes of data for which we have *no labelled examples at all*: some ground-truth classes of data are found only in the unlabelled dataset. It is this variety of data that we consider here, focusing on image data. A plausible cause of this would be selection bias, where the labelled data is from a biased sample of the overall data distribution.

First, we outline the pitfalls associated with trying to apply deep generative model (DGM)-based semi-supervised learning algorithms to datasets of this type. We then show how a combination of clustering and semi-supervised learning, using DGMs, can be brought to bear on this problem. We study several different datasets, showing how one can still learn effectively when half of the ground truth classes are entirely unlabelled and the other half are sparsely labelled.

A hypothetical example of a dataset of this type would be a set of medical images, scans say. As labelling is expensive, we obtain expert labelling—the variety of pathology present—for only some small proportion of all the scans we have. It is plausible that we might not happen to capture all distinct types of pathology in

this smaller labelled dataset. This could occur if we gathered all of our data to be labelled from only one hospital or ward that had a non-representative sample of the population.

Datashift could also cause a variety of biasing in the data collection. If new varieties or behaviours emerge over time, and unlabelled data continues to be gathered cheaply, while the expensive process of data labelling is not continued, then newly-emergent classes of data will not be found in the labelled dataset, only the unlabelled.

In a dataset of this type, an unlabelled image could be from one of the varieties that is captured in the labelled dataset, or it could be of another, unseen, variety. We would not be in the semi-supervised regime. Nor do we want to treat the problem as unsupervised, discarding our limited yet still information-rich labelled data.

We call this *semi-unsupervised learning*: we wish to jointly perform semi-supervised learning on sparsely-labelled classes and unsupervised learning on completely unlabelled classes. This requires a model that can learn successfully in both unsupervised and semi-supervised regimes.

We study how learning on sparsely-labelled data of this type introduces challenges, and describe how those challenges can be answered. First we discuss potential issues from using semi-supervised methods when the training data does not contain labelled examples from all ground truth classes—i.e. when it is semi-*un*supervised data. We demonstrate this using deep generative models used for semi-supervised learning: one can end up attributing all data merely to the classes represented in the labelled dataset. Then we show how some obvious approaches one might take to make use of semi-supervised learning algorithms do not work, even when one knows that the training data is, in fact, semi-unsupervised and tries to make allowances. Finally, we show how certain varieties of deep generative model-derived clustering algorithms can be used to handle this learning regime, over a range of datasets. In the case that half of the ground truth classes are masked out entirely, and the remaining half of classes are sparsely-labelled, we show we can still learn a classifier that performs well on the test set containing all ground-truth classes.

2.2 Background

2.2.1 Semi-Supervised Learning

If we had a fully labelled dataset we could easily train a discriminative model under maximum likelihood. We have input data and (one-hot) labels drawn from a dataset $(\mathbf{x}, y) \sim (\mathbf{X}_\ell, \mathbf{Y}_\ell) = \mathcal{D}_\ell$. $\mathbf{x} \in \mathbb{R}^{d_x}$ and $y \in \{0,1\}^K, \sum_{i=1}^K y_i = 1$. We could specify a parametric model for $p_\lambda(y|\mathbf{x})$, say a deep neural network, and aim to find the optimal parameters

$$\lambda^* = \arg\max_\lambda \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \log p_\lambda(y|\mathbf{x}) \tag{2.1}$$

via optimisation, say via stochastic gradient ascent on minibatches of data. But in semi-supervised learning we have an additional dataset $\mathcal{D}_\text{u}$, which contains unlabelled data $\mathbf{x}$ with no corresponding label, and further $|\mathcal{D}_\text{u}| \gg |\mathcal{D}_\ell|$.

Semi-supervised models for classification are designed to be able to learn from both datasets—to learn from sparsely-labelled data. This is of particular importance when $\mathcal{D}_\ell$ is too small to train a good enough classifier on its own. The hope would be that by somehow extracting information from $\mathcal{D}_\text{u}$ we can make a better classifier, say in terms of accuracy, when applied to new data, than if we had trained on $\mathcal{D}_\ell$ only (Chapelle et al., 2010).

While there are numerous approaches one could take to try to extract information from the unlabelled data to train a better classifier, here we will focus on the use of probabilistic generative models. As discussed in the introduction, in deep generative models the parameters of the distributions within the model are themselves parameterised by neural networks. Due to the coherency of probabilistic modelling, these models can handle missing observations in a principled way. Within this framework, we can do partial conditioning to obtain distributions of importance to us, here the (approximate) posterior distributions for unobserved labels.

A simple generative model based approach would be to fit a parametric joint distribution over data $(\mathbf{X}_\ell, \mathbf{Y}_\ell) = \mathcal{D}_\ell$, and $\mathbf{X}_\text{u} = \mathcal{D}_\text{u}$, then obtain a posterior for

the unknown labels conditioned on all the data one has. So we would first aim to find the maximum likelihood setting of the model parameters,

$$\theta^* = \arg\max_\theta p_\theta(\mathbf{X}_\ell, \mathbf{Y}_\ell, \mathbf{X}_\mathrm{u}) = \arg\max_\theta \sum_{\mathbf{Y}'_u} p_\theta(\mathbf{X}_\mathrm{u}, \mathbf{Y}'_u, \mathbf{X}_\ell, \mathbf{Y}_\ell). \tag{2.2}$$

This requires summing over all $K^{|\mathcal{D}_\mathrm{u}|}$ possible arrangements of $\mathbf{Y}_\mathrm{u}$. Then find the posterior for $\mathbf{Y}_\mathrm{u}$,

$$p_{\theta^*}(\mathbf{Y}_\mathrm{u}|\mathbf{X}_\ell, \mathbf{Y}_\ell, \mathbf{X}_\mathrm{u}) = \frac{p_{\theta^*}(\mathbf{X}_\mathrm{u}, \mathbf{Y}_\mathrm{u}, \mathbf{X}_\ell, \mathbf{Y}_\ell)}{\sum_{\mathbf{Y}'_u} p_{\theta^*}(\mathbf{X}_\mathrm{u}, \mathbf{Y}'_u, \mathbf{X}_\ell, \mathbf{Y}_\ell)}, \tag{2.3}$$

where again we have to perform a sum with a geometrically-scaling number of terms. Reminiscent of Blei et al. (2017), the integrals required to compute this posterior are often intractable for large datasets and large label spaces, so we must make approximations so that we can learn efficiently. And it is desirable that our learning algorithm results in a classifier as a *inference artifact*, that can be called on new data just as a normal neural network classifier can be.

2.2.2 Semi-Supervised Variational Autoencoders

We attack this problem using Variational auto-encoders (VAEs) (Kingma & Welling, 2014; Rezende et al., 2014) as they offer a modelling paradigm that can give us both of these properties. The Semi-Supervised VAE (SSVAE) proposed in Kingma et al. (2014) is a simple extension for semi-supervised learning. It has a continuous latent variable $\mathbf{z} \in \mathbb{R}^{d_z}$ and a partially-observed class variable y. For $\mathcal{D}_\mathrm{u}$ we only have input data $\mathbf{x}$, so for each $\mathbf{x}$ there is a corresponding latent variable y. For $\mathcal{D}_\ell$ we have observed y, so y is not a latent variable. In Kingma et al. (2014) the joint distribution is:

$$p_\theta(\mathbf{x}, y, \mathbf{z}) = p_\theta(\mathbf{x}|y, \mathbf{z})p(y)p(\mathbf{z}) \tag{2.4}$$

where $p(y) = \mathrm{Cat}(y|\pi)$ and $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, \mathbb{I})$. $p_\theta(\mathbf{x}|y, \mathbf{z})$ is an appropriate distribution given the form of the data, as discussed in § 1.4.1.

As exact inference is intractable, we perform stochastic amortised variational inference. We aim to optimise a lower bound on the evidence for all our data, the

two datasets $\mathcal{D}_{\mathrm{u}}, \mathcal{D}_{\ell}$. In each case, we need a variational posterior for the unobserved variables. $\mathbf{z}$ is always latent, and y is sometimes-latent, sometimes-observed. So for semi-supervised data the evidence lower bound consists of two terms. First, for unlabelled data (y is a latent variable to be inferred),

$$\mathcal{L}_{\mathrm{u}}(\mathbf{x}) = \mathbb{E}_{\mathbf{z},y\sim q_\phi(\mathbf{z},y|\mathbf{x})} \log \frac{p_\theta(\mathbf{x},y,\mathbf{z})}{q_\phi(\mathbf{z},y|\mathbf{x})}. \tag{2.5}$$

Secondly, for labelled data (y is observed),

$$\mathcal{L}_{\ell}(\mathbf{x},y) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x},y)} \log \frac{p_\theta(\mathbf{x},y,\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x},y)}. \tag{2.6}$$

Choosing a particular form for the posterior:

$$q_\phi(\mathbf{z},y|\mathbf{x}) = q_\phi(\mathbf{z}|\mathbf{x},y)q_\phi(y|\mathbf{x}) \tag{2.7}$$

$$q_\phi(y|\mathbf{x}) = \mathrm{Cat}(y|\pi_\phi(\mathbf{x})) \tag{2.8}$$

$$q_\phi(\mathbf{z}|\mathbf{x},y) = \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x},y),\Sigma_\phi(\mathbf{x},y)) \tag{2.9}$$

where $\mu_\phi(\mathbf{x},y), \Sigma_\phi(\mathbf{x},y), \pi_\phi(\mathbf{x})$ are neural networks.

$q_\phi(y|\mathbf{x})$ is an inference artifact, a classifier. But note that it only appears in $\mathcal{L}_{\mathrm{u}}(\mathbf{x})$, so it would only be trained on unlabelled data, clearly an undesirable property for a classifier. To remedy this, add to $\mathcal{L}_{\ell}(\mathbf{x})$ the cross entropy classifier loss, the same as inside Eq (2.1), weighted by a factor α (Kingma et al., 2014). The overall objective with unlabelled data $\mathcal{D}_{\mathrm{u}}$ and labelled data $\mathcal{D}_{\ell}$ is the sum of the evidence lower bounds for all data with this classification loss. This gives us the definition of a Semi-Supervised VAE:

Definition 2.1. *(Kingma et al., 2014) A Semi-Supervised VAE is a deep generative model with forward model as in Eq* (2.4) *and amortised variational posterior as in Eqs (2.7-2.9), trained w.r.t. its parameters θ, ϕ to maximise the objective:*

$$\mathcal{L}^{\mathrm{SSVAE}}(\mathcal{D}_{\mathrm{u}},\mathcal{D}_{\ell};\theta,\phi) := \mathbb{E}_{(\mathbf{x}_\ell,y_\ell)\sim\mathcal{D}_\ell}\left[\mathcal{L}_\ell(\mathbf{x}_\ell,y_\ell) + \alpha(\log q_\phi(y_\ell|\mathbf{x}_\ell))\right] + \mathbb{E}_{\mathbf{x}_{\mathrm{u}}\sim\mathcal{D}_{\mathrm{u}}}\,\mathcal{L}_{\mathrm{u}}(\mathbf{x}_{\mathrm{u}}). \tag{2.10}$$

Through joint optimisation over $\{\theta,\phi\}$ using stochastic gradient ascent we aim to find point-estimates of those parameters that maximises the evidence lower bound over both our datasets $\mathcal{D}_{\ell}$ and $\mathcal{D}_{\mathrm{u}}$.

For the expectations over $\mathbf{z}$ and y in the objective, we take Monte Carlo (MC) samples from the variational posteriors. To take derivatives through these samples w.r.t. θ, ϕ use *reparameterisation tricks*. For $\mathbf{z}$ this means rewriting the sample as a deterministic function given a sample from $\mathcal{N}(0, \mathbb{I})$, as discussed in § 1.3.4:

$$\mathbf{z} \sim \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \iff \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|0, \mathbb{I}), \mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{\frac{1}{2}} \cdot \boldsymbol{\epsilon}.$$

But what about our discrete variable y?

2.2.3 Gumbel-Softmax Trick / CONCRETE Sampling

Of course the simplest way to handle the required marginalisation of y, say in Eq (2.5), is to marginalise it. Doing so, however, requires K evaluations of $\mathcal{L}_\ell$ for each unlabelled datapoint in a batch. For relatively small numbers of classes this is acceptable, but as K increases even this linear computational complexity becomes challenging.

So we would like to have a reparameterisation trick for y, to take differentiable samples from $q_\phi(y|\mathbf{x})$. How can we do this? One approach to this is the Gumbel-Softmax trick/CONCRETE sampling (Jang et al., 2017; Maddison et al., 2017). Here we take a previous approach for sampling discrete variables, the Gumbel-Max trick (Yellott, 1977), that itself is not differentiable in term of the probabilities of the different possible outcomes, and then write a relaxation of that method that does allow us to differentiate w.r.t. those probabilities. This does come at the cost, however, of introducing bias into the procedure.

The Gumbel-Max trick is that if we take the vector of logits of the discrete distribution we wish to sample from, add independent Gumbel noise to each logit, and then take the arg max of the resulting vector, the resulting value is distributed according to discrete distribution we started with (Yellott, 1977).

This is not a differentiable process as we can not differentiate through arg max w.r.t. the logits of the distribution. The Gumbel-Softmax Trick replaces the arg max with a softmax function, with temperature τ. Intuitively, in the limit $\tau \to 0$

the softmax function becomes an argmax, returning a one-hot vector, and so the Gumbel-Softmax approximation becomes exact. At $\tau \to \infty$ the Gumbel-Softmax distribution becomes uniform. Variance is high for small τ and small for large τ. For a full derivation see Appendix B of (Jang et al., 2017).

Jang et al. (2017) provide advice on how to use this method. τ can be annealed according to a deterministic schedule (for which they give examples), or it can be a learned parameter of the model (where it corresponds to a form of entropy regularisation (Guo et al., 2017)).

If a method requires a one-hot sample exactly, rather than the approximations gained from Gumbel-Softmax, one can use Straight Through Gumbel-Softmax (ST-GS) sampling: Take the sample to be used directly from the original categorical distribution, and obtain gradients for that sample from the Gumbel-Softmax distribution with appropriate τ. And so this method provides one-hot samples regardless of temperature.

Here and in subsequent chapters we use the Gumbel-Softmax trick to take expectations over discrete latent variables, *not* using ST-GS but the raw relaxed samples during training, with temperature either a fixed or deterministically-annealed hyperparameter

2.3 Related Work

Semi-supervised and unsupervised learning each are large branches of machine learning. There are other methods that use a generative approach to these problems, including, for example, the Cluster-aware Generative Model (Maaløe et al., 2017) that can, like our approach, learn in both unsupervised and semi-supervised regimes. Auxiliary DGMs (Maaløe et al., 2016) could also be used within our approach. Generative Adversarial Networks (Goodfellow et al., 2014) have also been used to approach semi-supervised learning and clustering. Categorical Generative Adversarial Networks (Springenberg, 2016) and Adversarial autoencoders (Makhzani et al., 2016), the latter where the D_{KL} divergence in a VAE's ELBO is replaced

with a GAN-like discriminator, can each learn in both regimes, and thus could be appropriate for semi-unsupervised learning.

More recently, work has been done on a learning regime that is somewhat philosophically related to semi-unsupervised learning: new varieties of data are introduced in turn, entirely unsupervised, to a generative model as training progresses. The aim is to train a model that handles these new classes as they are introduced and does not catastrophically-forget previously-learnt clusters of data. The paper that first outlines this learning regime, Continuously Unsupervised Representation Learning (CURL) (Rao et al., 2019), uses the same clustering model that we make use of to tackle semi-unsupervised learning.

While all the above works take a generative approach to the problem, in recent years semi-supervised and unsupervised learning have been dominated by methods that learn a compressed representation of the input data using self-supervised and/or contrastive methods. These include Jigsaw (Noroozi & Favaro, 2016), AMDIM (Bachman et al., 2019), PIRL (Misra & van der Maaten, 2020), SvAV (Caron et al., 2020), CPC (van den Oord et al., 2018; Henaff et al., 2020), MoCo (He et al., 2020; Chen et al., 2020b), SimCLR (Chen et al., 2020a) and PCL (Li et al., 2020b). These kinds of methods simplify the problem of semi-supervised (and unsupervised) learning compared to a generative approach, as learning a cogent representation is generally easier than learning the underlying data distribution (Chapelle et al., 2010). Further, there are a range of effective methods that use psuedo-labels (the predictions of the model on unlabelled data) to learn in the semi-supervised regime; these include S^4L (Zhai et al., 2019), FixMatch (Sohn et al., 2020) and UDA (Xie et al., 2020).

The fact that different clustering algorithms can learn different partitions of the data has been discussed in the context of semi-supervised learning (Nigam et al., 2000; Corduneanu & Jaakkola, 2002), where it is called the 'indentifiability' of the clustering model. Incidentally, if the model is mis-matched in this sense, where the natural clustering of the model does not align with the labels being provided,

more unlabelled data can harm rather than improve performance (Chapelle et al., 2010). That the assumptions of semi-supervised learning, for example that all ground-truth classes are found in the labelled dataset, may not always hold is discussed briefly in Oliver et al. (2018).

Semi-unsupervised learning has similarities to some varieties of zero-shot learning (ZSL) (Weiss et al., 2016), though in zero-shot learning one generally has access to auxiliary 'attribute' information at training time, which we do not. Semi-unsupervised learning also has links to transfer learning, particularly methods that attempt to discover new classes of data (Hsu et al., 2018, 2019; Han et al., 2019).

2.4 Semi-Unsupervised Learning with Semi-Supervised Models

Can we use standard semi-supervised approaches with semi-*un*supervised data? In this section we demonstrate how SSVAEs perform. First, we cover the *by accident* case where we think we are performing semi-supervised learning, but our unlabelled data does contain extra classes. Then we cover the *on purpose* case, where we expect that there are extra classes of data and make reasonable allowances for them in our semi-supervised modelling.

2.4.1 By Accident

What happens if we train an SSVAE described above, on data that we believe conforms to our requirements for semi-supervised learning, but in fact contains additional ground-truth classes of data in the unlabelled dataset?

Experiments To mimic semi-unsupervised data using standard datasets we simply mask out values of the training data. We train an SSVAE on MNIST (LeCun & Cortes, 2010) and Fashion-MNIST (Xiao et al., 2017). For these experiments, we have to pick the overall dimensionalities of d_z. We use $d_z = 5$ for MNIST and $d_z = 10$ for Fashion-MNIST. For this and all subsequent experiments in this

Table 2.1: Accidental Semi-Unsupervised Learning. We want to see how performance changes when a semi-supervised model is trained with additional unlabelled classes in the unlabelled dataset. We show here the test set accuracy (Acc. $\% \pm$ SD) of SSVAE trained on MNIST and Fashion-MNIST (F-MNIST) over four runs when trained in two ways. SS (semi-supervised): trained with vanilla semi-supervised data but only for classes $\{0, .., 4\}$ with 20% of datapoints labelled. Accidental SUS (semi-unsupervised): the unlabelled training set also includes unlabelled examples of classes $\{5, ..., 9\}$. The test set is classes $\{0, .., 4\}$ only in both cases.

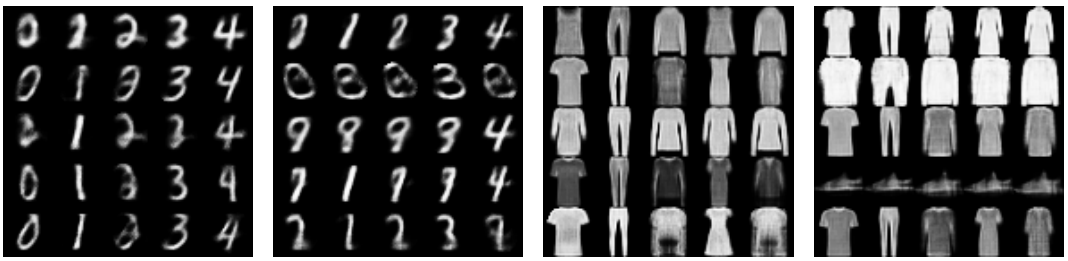
Dataset	SS	Accidental SUS
MNIST	97.0 ± 0.4	97.4 ± 0.3
F-MNIST	90.1 ± 0.3	89.8 ± 0.2

chapter we use small MLPs to represent the parameters for the distributions in the generative and recognition models. We train our models using ADAM (Kingma & Lei Ba, 2015). For full implementation details, see the appendix A.1.

The labelled training set $\mathcal{D}_\ell$ contains examples only of classes $\{0, .., 4\}$, 20% of the total examples in the standard training set, so ≈ 1000 labelled examples for each class in (Fashion-)MNIST. The unlabelled dataset $\mathcal{D}_\text{u}$ is thus all other training data, with labels dropped: all training data points for classes $\{5, ..9\}$ and the left-over 80% of training data points for classes $\{0, .., 4\}$.

We compare this to being trained in an equivalent semi-supervised manner, where the unlabelled data just contains classes $\{0, .., 4\}$. We find that for MNIST and Fashion-MNIST there is no significant change in test-set classification performance on classes $\{0, .., 4\}$ due to the additional presence of unexpected classes in the unlabelled data, see Table 2.1.

More apparent is the the effect on generated data, as semi-unsupervised data distorts the properties of the forward model. Under maximum-likelihood training, models with sufficient capacity will learn a way to have representations that correspond to the unlabelled-only classes. Put another way, we can expect a well-trained model (i.e. avoiding training pathologies like posterior collapse (Burda et al., 2016)) will learn to reconstruct its training data with high fidelity, meaning that there will be some setting of their latent representations that correspond closely to any

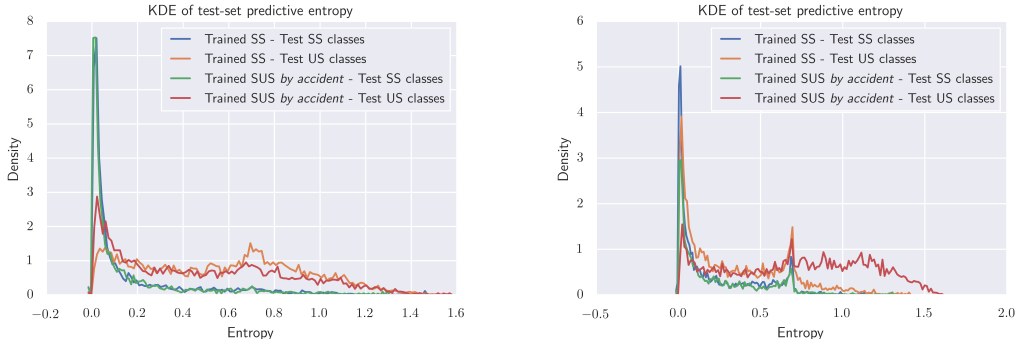


(a) MNIST - SS **(b)** MNIST - SUS accident **(c)** F-MNIST - SS **(d)** FMNIST - SUS accident

Figure 2.1: We show the effect of having unexpected additional classes in the training set for an SSVAE when trained on MNIST and Fashion-MNIST. We are plotting the mean of $p_\theta(\mathbf{x}|y, \mathbf{z})$ for a range of samples of $\mathbf{z}$ and for all values of y. In each plot, each row is generated using a shared draw $\mathbf{z}^* \sim p(\mathbf{z})$, with the columns indexing over y. The models were trained with a small labelled dataset of classes $\{0, ..., 4\}$. The semi-supervised (SS) model's unlabelled training set contained only those same classes. The semi-unsupervised by accident (SUS accident) model's unlabelled training set contained all classes $\{0, ..., 9\}$. The *semi-supervised* plots a) and c) shows successful controlled generation, as each column—each value of y—corresponds to a distinct class, with $\mathbf{z}$ encoding style. The *semi-unsupervised by accident* plots b) and d) show that for some settings of $\mathbf{z}$ we get generation of datapoints that look like the unlabelled-only classes, in some cases regardless of the value of y: in b) the 2^{nd} and 3^{rd} rows, in d) the 4^{th} row.

given training datapoint. Thus, when sampling the values of the model's latents during generation, those settings that correspond to unlabelled-only classes can be picked. We see this in Figure 2.1. We sample from the model for a set of samples $\mathbf{z}^* \sim p(\mathbf{z})$, plotting the mean of $p(\mathbf{x}|\mathbf{z}^*, y)$ for each value of y. Each row corresponds to the same $\mathbf{z}^*$ sample, each column corresponding to a value of y. For the models trained on semi-supervised data, generation is well controlled by these variables, with y controlling identity and $\mathbf{z}$ controlling style. For the models trained on semi-unsupervised data, we sometimes generate data that looks like one of the only-unlabelled classes $\{5, ..., 9\}$. By the design of the model, this cannot be done for a particular setting of y, so instead this is done through the information in $\mathbf{z}$. Thus we see that for some samples of $\mathbf{z}$, regardless of the value of y, the decoder output looks like an image from one of the unlabelled-only classes.

While it is perhaps desirable that a semi-supervised model's classification performance is robust to having been trained with semi-unsupervised data, it means that we cannot use that performance metric to detect if we have erred in our assumption



(a) MNIST **(b)** F-MNIST

Figure 2.2: KDE plots of entropy of test set $q_\phi(y|\mathbf{x})$ for MNIST and F-MNIST with the same training setup as Figure 2.1. The test set is either semi-supervised classes (SS) $\{0, ..., 4\}$ only or unsupervised classes (US) $\{5, ..., 9\}$ only. We see that for F-MNIST when trained SUS *by accident* there is higher entropy for the US classes, but for MNIST there is not a clear effect.

that we are in the semi-supervised regime. In Figure 2.2.b) we see that when trained semi-unsupervised *by accident* on Fashion-MNIST the classifier is perhaps more likely to output a high entropy predictive distribution for classes only observed as unlabelled data. However, this effect does not hold for MNIST. Overall, it is not at all clear that one could tell that extra unlabelled-only classes were present in the training data merely from quantitative measures.

To recap: the main problems with training a semi-supervised model on semi-unsupervised data *by accident* are first that a practitioner may struggle to know that there are extra classes hidden in their unlabelled dataset, and second, the model will necessarily attribute all incoming data point to only the classes in the limited labelled dataset.

2.4.2 On Purpose

If we have reason to think there are additional ground-truth classes present in the unlabelled dataset $\mathcal{D}_\text{u}$, a natural approach would be to have extra dimensions in y. For example, if we have n_ℓ classes each with some labelled examples and we think there is some number n_aug of additional classes present in the unlabelled data, we could reasonably choose $y \in \{0, ..., (n_\ell + n_\text{aug})\}$. We might then hope that our

semi-supervised learning algorithm would make use of these 'empty' dimensions in y, using them to encode the additional classes we hope it will discover.

Further, we could choose n_{aug} to be more than the number of extra ground-truth classes we think there are, giving more components in y over to them: a common practice in clustering (Yang et al., 2016; Makhzani et al., 2016; Kilinc & Uysal, 2018; Dilokthanakul et al., 2017). It gives us a degree of insurance against sub-optimal agglomeration of different kinds of data into the discrete latent variables in the model. For example, for MNIST we might think that we should give the model the potential to use multiple values of y for the same ground-truth class to capture the distinct varieties that exist in how people write certain digits. Sticking with the above, masked, dataset, from having access to labels for classes $\{0, .., 4\}$ we might have noticed that people write 4s in two different manners, open and closed. So an analyst may have reason to expect that any extra classes of digit they might discover in the unlabelled data could also vary in their manner of writing, requiring additional latent capacity.

Prior in y We need to specify a prior $p(y)$. We choose here to divide the probability mass between the known-classes and the expected-classes in proportion to our prior expectation of their proportions in the data. Then within the known, labelled classes we choose a prior proportional to their frequency; for the expected additional classes, we divide mass uniformly.

For the datasets we study, we know the ground truth classes are all of approximately equal frequency, and that the classes we have observed correspond to half of all the ground truth classes. So we divide the mass of our prior in half between the classes we have some labelled examples for and the expected classes. Thus

$$p_\theta(y) = \begin{cases} \frac{1}{2n_\ell} & y \in \{0, ..., n_\ell\} \\ \frac{1}{2n_{\text{aug}}} & y \in \{n_\ell + 1, ..., K\} \end{cases}$$

where $K = n_{\text{aug}} + n_\ell$. We use this prior for all experiments where we are trying to take account of the presence of additional unlabelled classes.

Evaluating performance on unlabelled classes In evaluating a model trained semi-unsupervised, we must choose a method for evaluation that naturally applies both to the semi-supervised and the unsupervised classes. In pure semi-supervised learning, we can simply measure accuracy on the test set (if the classes are balanced and the cost of misclassification is the same in all cases). A close analogy to accuracy, used to evaluate DGM-based clustering algorithms is cluster accuracy (ACC) (Jiang et al., 2017; Yang et al., 2016; Kilinc & Uysal, 2018). This is a test-time version of *cluster and label*.

$$\text{ACC} = \max_{P \in \mathcal{P}} \frac{\sum_{i=1}^{|\mathcal{D}|} \mathbb{I}[t_i = P y_i]}{|\mathcal{D}|} \tag{2.11}$$

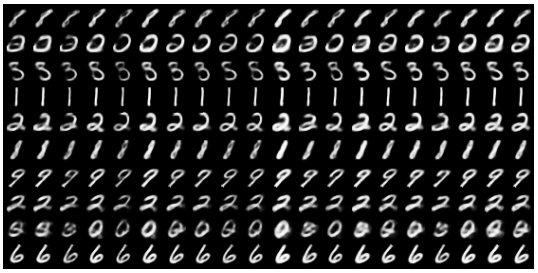
where P is a $T \times K$ rectangular permutation matrix that attributes each y to a ground truth class t.

We can think of this as solving an assignment problem. We need to link each of the learnt, unsupervised classes with a particular (never seen labelled during training) ground-truth class. So we attribute each learnt class to the ground truth class that is the most common ground truth class within it at test time. The assignment we get from this method will also enable us to plot confusion matrices in the semi-unsupervised case.

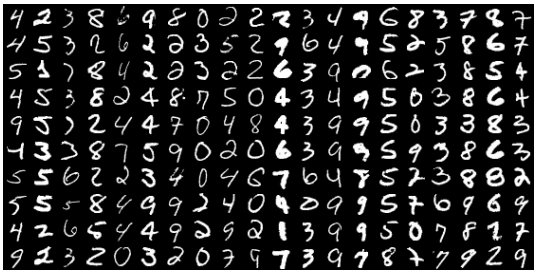
Experiments We use Fashion-MNIST, MNIST, and HAR (Stisen et al., 2015); masking the first half of classes partially and the second half fully in the same way as in the sub-section above. We augment $K = n_\ell + n_{\text{aug}} = n_{\text{gt}}/2 + n_{\text{aug}}$. We use $n_{\text{aug}} = 40$ in these experiments. We see from the example confusion matrices in Figure 2.5 and the overall results in Table 2.2 that no informative classifier is learnt over the unlabelled classes by an SSVAE.

2.4.3 Inductive Bias Matching

In an SSVAE, we are specifying that the latent space is composed of a continuous part and a discrete part. By training on semi-supervised data, we direct the discrete



(a) Samples after clustering with a SSVAE



(b) SSVAE's most confidently assigned datapoints, having been trained to cluster

Figure 2.3: We trained an SSVAE unsupervised with a 20 dimensional discrete latent y. In a), as in Figure 1, we are showing the mean of $p_\theta(\mathbf{x}|y,\mathbf{z})$ conditioned on all values of y for various samples of $\mathbf{z}$ drawn from the prior. Columns index over values of y, each row being a particular $\mathbf{z}$ samples. In b) we show the 10 most confidently-assigned data points in the test set for this model. From both a) and b) we see that different y classes, different columns, correspond to different stroke thicknesses, and from a) we see that digit identity is represented in $\mathbf{z}$.

part to correspond to the limited class labels that we have and that the continuous part should encode residual 'style' information.

By construction, generative models used for semi-supervised learning can be run using only unlabelled data: nothing is stopping us from attempting to use such a model for clustering. When trained on unlabelled information we can see that an SSVAE does not learn a clustering that matches the ground-truth classes we are interested in. On MNIST, the assignment of roles between the two latent variables is the opposite of what we desire: $\mathbf{z}$ encodes identity, with y encoding stroke thickness. We see this in Figure 2.3, where we show samples from an SSVAE trained unsupervised on MNIST with 20 cluster components. Each row corresponds to a sample from $p(\mathbf{z})$ and each column to a setting of y. This carries over to the unlabelled sub-problem in semi-unsupervised learning: we have poor performance

over the unlabelled classes as we can see in Figure 2.5 and Table 2.2. One might expect that the presence of some labelled data would induce the appropriate division of roles between $\mathbf{z}$ and y, with all (labelled and unlabelled) classes indexed in y, but this is not the case. Instead, we must turn to approaches with the appropriate clustering performance to solve this task.

2.5 Semi-Unsupervised Learning with Clustering Models

Attempting to learn using semi-unsupervised data with models developed for semi-supervised data has shown us to run into problems due to them having the wrong inductive biases. Even when expanding the dimensionality of the discrete label-space, we find that the SSVAE will not use these additional components to solve the task we care about. Classes of data found only in the unlabelled dataset are not separated. This is because these models cannot perform clustering in the terms we care about, including when that clustering is a sub-problem.

Thus, a reasonable way to wish to learn with semi-unsupervised datasets is to have a model that, in the absence of any label information, learns to cluster our data in a way that corresponds to the limited label information we do have. When we train our model with semi-unsupervised data, we expect those additional classes to have a similar nature to the partially-labelled classes. If our model clusters appropriately, we can expect it to assign any new classes of data (present only in the unlabelled dataset) to distinct components in y.

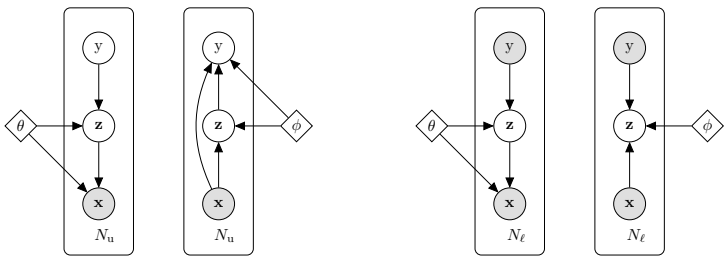
This paper has used SSVAEs as an anchoring example, and it demonstrates a general point around using semi-supervised models for, in effect, clustering. We cannot expect that generative models used for semi-supervised learning will necessarily have the right inductive biases to be run as appropriate clustering algorithms for the particular kind of label information we are interested in. For that reason, in order to capture semi-unsupervised learning, we have to have an appropriate clustering algorithm. We call this the *inductive bias requirement*: we want to have a clustering

algorithm that has the right inductive bias to produce clustering at the right level for our task—digit identity rather than stroke thickness, for MNIST. This property is not given by an SSVAE, for any of the datasets we have studied.

Further, if we are working with VAE-derived models, we are aided if our model has an amortised posterior for the clustering latent variable. For then we can train powerful neural network classifiers inside our models. We call this the *classifier requirement*. This property is given by an SSVAE.

Our experiments above tell us that we were unlucky in trying to capture semi-unsupervised learning with an SSVAE, as our task of encoding digit identity or garment-type into y is not the natural mode of behaviour for this model. Conversely, if we had been interested in semi-unsupervised learning of discrete degrees of stroke width, say, then an SSVAE would be an appropriate model. Moving forward with the standard labels given with the data sets under consideration, we show how a clustering algorithm that fulfills both the inductive-bias requirement and the classifier requirement can be used to perform semi-unsupervised learning.

2.5.1 Gaussian Mixture Deep Generative Models



(a) Gen. y latent **(b)** Post. y latent. **(c)** Gen. y obs. **(d)** Post. y obs.

Figure 2.4: Generative and Approximate Posterior models for GM-DGM, where N_u is the number of unlabelled points and N_ℓ the number of labelled points.

Gaussian Mixture Deep Generative Models (GM-DGMs) have been popular clustering algorithms. In various guises, they have been shown to produce clusters that correspond well to the ground-truth labels provided on various machine learning

datasets (Dilokthanakul et al., 2017; Jiang et al., 2017; Nalisnick et al., 2016; Shu, 2016). This means it fulfills the inductive bias requirement.

A simple version of a GM-DGM is to have a Gaussian mixture in $\mathbf{z}$, each component the result of conditioning on y:

$$p_\theta(\mathbf{x}, y, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z}|y)p(y) \tag{2.12}$$

$$p(y) = \text{Cat}(y|\pi) \tag{2.13}$$

$$p_\theta(\mathbf{z}|y) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\theta(y), \boldsymbol{\Sigma}_\theta(y)) \tag{2.14}$$

And use the same form of variational posterior as used for SSVAEs, Eqs (2.7 - 2.10). By keeping that posterior, we keep a parameterised $q_\phi(y|\mathbf{x})$, fulfilling the classifier requirement. The training objective remains Eq (2.10), using the new generative model. Thus the definition of a GM-DGM is:

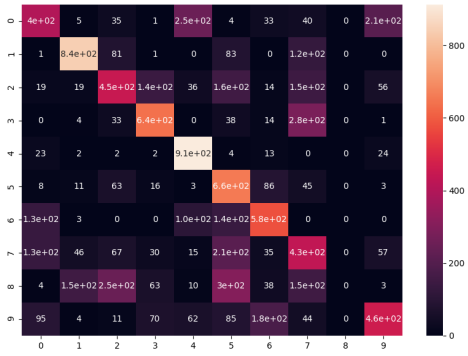
Definition 2.2. *A GM-DGM is a deep generative model with forward model as in Eqs (2.12-2.14) and amortised variational posterior as in Eqs (2.7-2.9), trained w.r.t. its parameters θ, ϕ to maximise the objective Eq* (2.10)*.*

See Figure 2.4 for a graphical representation of this model.

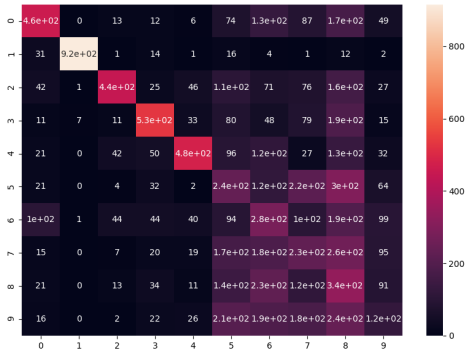
2.6 Experiments

We keep the experimental setup as in Section 2.4.2. And as in earlier experiments, we use small MLPs to parameterise the parameters of the distributions in the generative and recognition models, trained using ADAM (Kingma & Lei Ba, 2015). Again we use $d_z = 5$ for MNIST, $d_z = 10$ for Fashion-MNIST, and $d_z = 15$ for HAR. We describe our model implementation and data processing in detail in Appendix A.1.

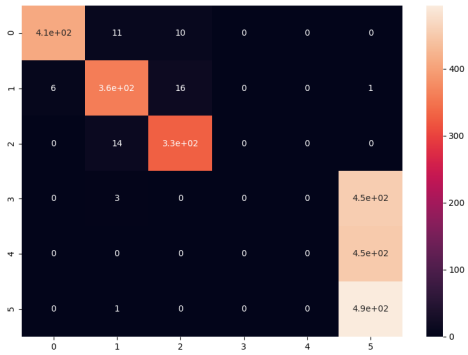
As before, in semi-supervised runs, we keep labels for 20% of the data. In the semi-unsupervised learning experiments, we keep 20% of label data for the first half of classes: $\{0, ..., 4\}$ for (Fashion-)MNIST, $\{0, 1, 2\}$ for HAR. We then mask out all label information for the $n_{\text{gt}}/2$ remaining classes.



(a) SSVAE Test Set Confusion Matrix on MNIST, accuracy: 0.537

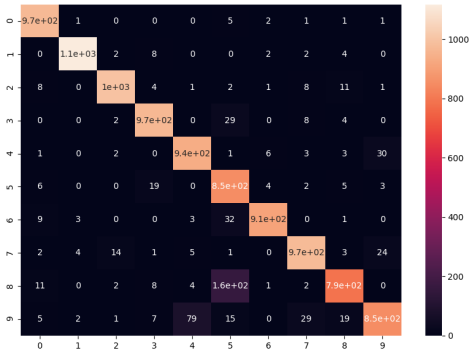


(b) SSVAE Test Set Confusion Matrix on F-MNIST, accuracy: 0.404

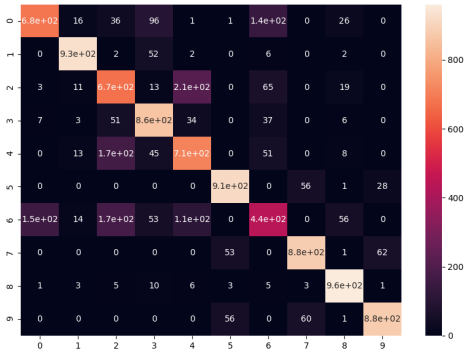


(c) SSVAE Test Set Confusion Matrix on HAR, accuracy: 0.618

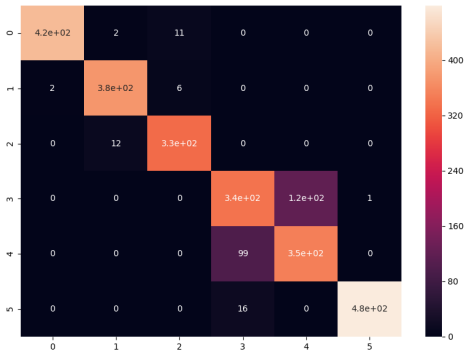
Figure 2.5: Example test set confusion matrices for SSVAEs trained semi-unsupervised on MNIST, Fashion-MNIST, and HAR. We discard all labels for classes $4-9$ for (F-)MNIST and $3-6$ for HAR, and keep 20% of labels for other classes. We assign each learnt cluster component to its most-commonly contained ground-truth class; using these assignments we can then plot the confusion matrix against ground truth classes.



(a) GM-DGM Test Set Confusion Matrix on MNIST, accuracy: 0.936



(b) GM-DGM Test Set Confusion Matrix on F-MNIST, accuracy: 0.792



(c) GM-DGM Test Set Confusion Matrix on HAR, accuracy: 0.893

Figure 2.6: Example test set confusion matrices for GM-DGMs trained semi-unsupervised on MNIST Fashion-MNIST and HAR. We discard all labels for classes $4-9$ for (F-)MNIST and $3-6$ for HAR, and keep 20% of labels for other classes. We assign each learnt cluster component to its most-commonly contained ground-truth class; using these assignments we can then plot the confusion matrix against ground truth classes.

Table 2.2: Test set accuracy/cluster purity of SSVAE and GM-DGM trained on MNIST, Fashion-MNIST (F-MNIST) and HAR, over four runs each for unsupervised learning (US), semi-supervised learning (SS) and semi-unsupervised learning (SUS).

Model	Dataset	US Acc. % ± SD	SS Acc. % ± SD	SUS Acc. % ± SD
SSVAE	MNIST	26.1 ± 2.8	97.6 ± 0.1	54.3 ± 0.9
GM-DGM	MNIST	$\mathbf{90.0 \pm 1.5}$	$\mathbf{97.7 \pm 0.1}$	$\mathbf{92.5 \pm 0.3}$
SSVAE	F-MNIST	18.2 ± 0.4	86.8 ± 0.2	38.7 ± 1.7
GM-DGM	F-MNIST	$\mathbf{75.8 \pm 0.3}$	$\mathbf{86.9 \pm 0.1}$	$\mathbf{78.22 \pm 1.1}$
SSVAE	HAR	29.0 ± 1.1	$\mathbf{97.7 \pm 0.1}$	49.7 ± 7.5
GM-DGM	HAR	$\mathbf{81.7 \pm 1.8}$	96.6 ± 0.3	$\mathbf{87.1 \pm 2.3}$

2.6.1 GM-DGM Results

In Figures 2.5, 2.6 we show example confusion matrices for SSVAEs and GM-DGMs trained semi-unsupervised on our studied datasets. To plot these confusion matrices, we are assigning each learnt cluster component to its most-commonly contained ground-truth class; the same asignment used in cluster accuracy,Eq (2.11).

While the SSVAE was unable to cluster the subspace of unsupervised classes in y, Figure 2.5, our GM-DGM model can learn a predictive classifier/clusterer for both the semi-supervised and unsupervised classes, Figure 2.6. See Figs A.1, A.2 for additional example confusion matrices for SSVAEs and GM-DGMs trained unsupervised and semi-supervised on these datasets.

We show the test-set accuracy after unsupervised, semi-supervised, and semi-unsupervised learning for SSVAEs and GM-DGMs in Table 2.2.

While SSVAEs demonstrably fail to cluster effectively over the ground truth classes when trained unsupervised, our GM-DGM can. As our GM-DGM can also do semi-supervised learning at the same overall level of performance as SSVAEs (as show in the middle column in Table 2.2) it can thus capture semi-unsupervised learning as well. As one might expect, the final performance in semi-unsupervised learning for GM-DGMs is above that for unsupervised learning and below vanilla semi-supervised learning. Our GM-DGM approach has been able to learn successfully to classify

and cluster for all ground truth classes of data in comparison to SSVAEs, for all data sets studied.

2.7 Conclusion

We introduced semi-unsupervised learning, an extreme limit case of semi-supervised learning where for some classes of data in our dataset there are no labelled examples at all. We show that a common approach to semi-supervised learning using DGMs does not work for data of this type, even when making allowances for the nature of the training data.

To capture semi-unsupervised learning, we propose that a model must be able to cluster data in a way that matches the structure given by the limited label information. We call this the *inductive bias requirement*. And for the sake of ease of use, it is desirable to have an explicit representation of the posterior predictions of classes for any labelled data we have to train on. Such an object can then be used directly as a classifier at test time. We call this the *classifier requirement*. We have demonstrated that a simple Gaussian mixture deep generative model, with appropriate amortised variational posterior, can fulfill these requirements, learning successfully in the semi-unsupervised regime for the datasets studied.

We hope for further study of this new learning regime, as it is potentially the true state of affairs when learning with limited or biased labelled data.

In dreams begin responsibilities

— W. B. Yeats

3

Relaxed-Responsibility Hierarchical Discrete VAEs

Contents

3.1 Motivation

Successfully training Variational Autoencoders (VAEs) with a hierarchy of discrete latent variables remains an area of active research. Vector-Quantised VAEs are a powerful approach to discrete VAEs, but naive hierarchical extensions can be unstable when training. Leveraging insights from classical methods of inference we introduce *Relaxed-Responsibility Vector-Quantisation*, a novel way to parameterise discrete latent variables, a refinement of relaxed Vector-Quantisation that gives better performance and more stable training. This enables a novel approach to hierarchical discrete variational autoencoders with numerous layers of latent variables (here up to 32) that we train end-to-end. Within hierarchical probabilistic deep generative models with discrete latent variables trained end-to-end, we achieve state-of-the-art bits-per-dim results for various standard datasets. Further, we observe different layers of our model become associated with different aspects of the data.

3.2 Introduction

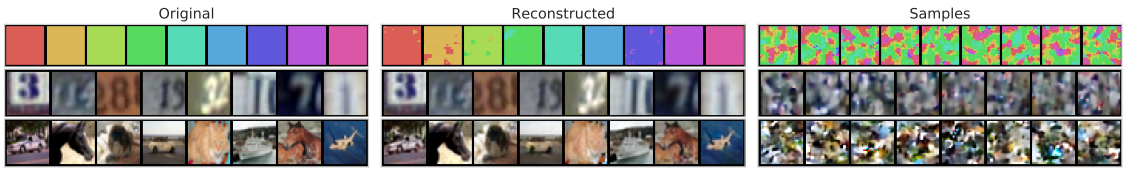
Probabilistic deep generative models, such as Variational Autoencoders (VAEs), have had significant and continuing success in learning continuous representations of data (Kingma & Welling, 2014; Rezende et al., 2014; Kingma et al., 2016; Vahdat & Kautz, 2020; Child, 2021). The learning of discrete representations has also flourished (Grathwohl et al., 2018; van den Oord et al., 2017; Razavi et al., 2019a; Fortuin et al., 2019; Pervez et al., 2020) and remains an active area of research. Discrete representations are useful as they are intrinsically compact, finding application in various tasks such as compression and clustering. Advances in differentiable relaxations of discrete probability distributions (Maddison et al., 2017; Jang et al., 2017) have contributed to training discrete latent variables models on high-dimensional data using gradient-based methods (Sønderby et al., 2017). However, training rich hierarchical models with discrete latent variables for high-dimensional data remains a problem in the field (Liévin et al., 2019; Williams et al., 2020; Pervez et al., 2020).

Here we propose an effective, scalable method for learning hierarchical discrete representations of image data within a unified probabilistic framework. This work builds on Vector-Quantised Variational Autoencoders (VQ–VAEs) (van den Oord et al., 2017) and their relaxation (Sønderby et al., 2017).

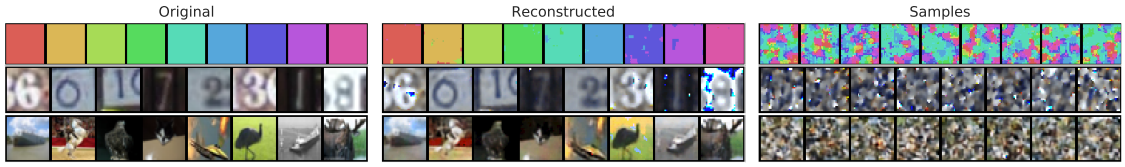
VQ–VAEs reach surprisingly poor raw bits-per-dim (bpd), a scaled form of the ELBO, on both the train and test sets. Thus to achieve good performance they require post-hoc training of density estimators on learnt embeddings. We begin by analysing how this happens. Perhaps one might think it is because of the probabilistic structure of these models—that having discrete latent variables as the prior leads to poor generations. We find that VAEs with the same neural structure as VQ–VAEs—convolutional neural networks with latents laid out spatially—but with Gaussian latents show the same pathologies. See Fig 3.1 for a demonstration.

This motivates us to develop a novel variety of hierarchical discrete VAEs. Previously developed hierarchical structures based around VQ building blocks have required various heuristics in model formulation and training (Williams et al., 2020) or highly restricted probabilistic structure (Pervez et al., 2020). We find that naive hierarchical extensions can be unstable during training. With a novel, richer formulation of probabilistic vector quantisation we train hierarchical discrete latent variable models end-to-end within a unified probabilistic framework. These models, which we call *Relaxed-Responsibility Vector-Quantised VAEs* or RRVQ–VAEs, have a hierarchical structure that means they achieve state of the art bits-per-dim for this class of models.

Our models show superior performance when compared against VQ–VAEs with their initial priors and naive hierarchical extensions, as well as various baselines. We find that performance increases as we increase the number of layers of latent variables in our model, with the deepest models we train having 32 layers. Further, we demonstrate that our model places information about different aspects of the images into different latent layers. We also demonstrate that our approach can be used to perform compression.



(a) rVQ–VAE



(b) $L = 1$ Spatial–VAE

Figure 3.1: Here we demonstrate that the poor quality draws when sampling from a VQ–VAE's prior $p(\mathbf{z})$ is not from having discrete latents, but from the spatial arrangement of latent variables. We train (a) rVQ–VAEs and (b) $L = 1$ Spatial–VAEs (a VAE with continuous latents, but arranged spatially like a VQ–VAE) on (top) a toy dataset composed of 9 colour swatches, (middle) SVHN, (bottom) CIFAR-10. For each dataset, both models give good reconstructions (middle column) but ancestral samples from the prior $p(\mathbf{z})$ (right column) are very dissimilar to datapoints in the training set, even for the toy dataset—for which we do not see uniformly-coloured images, instead we see regions of each the different colours of the dataset. This shows that it is the method used to parameterise the model's latent variables that leads to this sampling phenomena, not being discrete vs continuous.

RRVQ-VAEs help to close the performance gap between discrete VAEs and their continuous counterparts. This approach opens up new avenues for the building of hierarchical discrete VAEs and is a step towards a unified probabilistic framework for specifying and training models of this type.

3.3 Background

3.3.1 Vector Quantised Variational Autoencoders

The Vector-Quantised Variational Autoencoder (VQ–VAE) (van den Oord et al., 2017) is a density estimator for high dimensional data such as audio, images and video. Instead of having continuous latent variables, as in the vanilla VAE, the latents $\mathbf{z}$ are a set of M discrete variables $\mathbf{z} = \{z^1, \ldots, z^M\}$ each of dimensionality K. The joint $p_\theta(\mathbf{x}, \mathbf{z})$ factorises as for a vanilla VAE, but with

$$p(\mathbf{z}) = \prod_{m=1}^{M} \text{Cat}\left(z^m | \frac{1}{K}\right). \tag{3.1}$$

The likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ does not depend directly on samples of $\mathbf{z}$. Rather the discrete vector $\mathbf{z}$ is used to index over a dictionary of K embeddings, the codebook vectors $\mathbf{E} = \{\mathbf{E}^k\}$, each $\mathbf{E}^k \in \mathbb{R}^{d_e}$, d_e being the dimensionality of the embedding space. For stochastic amortised variational inference in VQ–VAEs, introduce a recognition network $\mathbf{e}_\phi(\mathbf{x}) \in \mathbb{R}^{M \times d_e}$ outputting M vectors in $\mathbb{R}^{d_e}$, the embedding space. The posterior $q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{m=1}^{M} q_\phi(z^m|\mathbf{x})$ is then defined via a nearest-neighbour vector-lookup. For each latent z^m,

$$q_\phi\left(z^m = k|\mathbf{x}\right) = \begin{cases} 1 & \text{if } k = \arg\min_j \left|\mathbf{e}_\phi^m\left(\mathbf{x}\right) - \mathbf{E}^j\right|_2^2 \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

This is a one-hot posterior: $q_\phi(\mathbf{z}|\mathbf{x})$ is deterministic. In a vanilla VAE we train the model by maximising the ELBO, $\mathcal{L}(\mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim q} \log p_\theta\left(\mathbf{x}|\mathbf{z}\right) - D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{z}|\mathbf{x}\right)||p(\mathbf{z})\right)$, over the dataset with respect to the generative and recognition model parameters. This standard training approach is not appropriate for this discrete model, for two reasons. Firstly, since it is not possible to differentiate through the vector lookup operation (due to the $\arg\min$) we cannot use differentiable samples to take gradients through Monte Carlo estimates of the expectations. Secondly, the one-hot posterior makes the D_{KL} term constant (equal to $M \log K$) so there is no regularisation on the posterior representations.

Thus, a VQ–VAE has two extra terms in its objective: a vector quantisation loss to train the embeddings; and a commitment loss to control the output of the embedding network, weighted by a chosen hyperparameter β (van den Oord et al., 2017).

3.3.1.1 rVQ–VAEs

Instead of the deterministic posterior found in a vanilla VQ–VAE, a Gumbel-Softmax distribution (Maddison et al., 2017; Jang et al., 2017) can be used to specify a posterior distribution from which we can take differentiable samples (Sønderby et al., 2017). This means that the posterior $q_\phi(\mathbf{z}|\mathbf{x})$ is no longer a one-hot, deterministic distribution and the D_{KL} in $\mathcal{L}$ is no longer a fixed constant. The model becomes probabilistic again. Thus the VQ-related loss terms are no longer needed and the

codebook $\mathbf{E}$ can be learnt via gradient descent. One can choose the logits of the posteriors to be proportional to the square distance between the given embedding vector and each codebook vector (Sønderby et al., 2017),

$$q_\phi(\mathbf{z}|\mathbf{x};\mathbf{E}) = \prod_{m=1}^{M} \text{Cat}\left(\mathbf{z}_m|\pi_\phi^m\left(\mathbf{x};\mathbf{E}\right)\right), \tag{3.3}$$

$$\pi_\phi^{m,k}(\mathbf{x};\mathbf{E}) \propto \exp\left(-\frac{1}{2}\left|\mathbf{e}_\phi^m(\mathbf{x}) - \mathbf{E}^k\right|_2^2\right). \tag{3.4}$$

This enables us to define *Relaxed*–VQ–VAEs:

Definition 3.1. *(Sønderby et al., 2017) A Relaxed–VQ–VAE is a deep generative model with forward model as in Eqs (1.34,3.1) and amortised variational posterior as in Eqs (3.3,3.4), trained w.r.t. its parameters $\theta, \phi, \mathbf{E}$ to maximise the objective:*

$$\mathcal{L}^{\text{rVQ-VAE}}(\mathcal{D};\theta,\phi,\mathbf{E}) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\mathbb{E}_{\mathbf{z}\sim q}\log p_\theta\left(\mathbf{x}|\mathbf{z};\mathbf{E}\right) - D_{\text{KL}}\left(q_\phi\left(\mathbf{z}|\mathbf{x};\mathbf{E}\right)||p(\mathbf{z})\right)\right] \tag{3.5}$$

To stochastically-estimate this objective in a way that enables us to differentiate w.r.t. model parameters we have to obtain differentiable samples from the discrete distribution $q_\phi(\mathbf{z}|\mathbf{x};\mathbf{E})$, which is done using the Gumbel-Softmax trick, § 2.2.3. These *Relaxed*–VQ–VAEs (henceforth rVQ–VAEs) have been shown to make better use of their latent variables than the deterministic base model, obtaining higher values of $\mathcal{L}$ (Sønderby et al., 2017) both at train and test time. Both Vanilla–VQ–VAEs and rVQ–VAEs train relatively stably. By definition Vanilla–VQ–VAEs avoid posterior collapse (Bowman et al., 2016; Razavi et al., 2019b; Dai & Wipf, 2019; Lucas et al., 2019) as the D_{KL} term in $\mathcal{L}$ is constant. For rVQ–VAEs matching the posterior to the prior in the latent space is not possible in the general case as it would require the posterior embedding to be equidistant from all codebook vectors.

In this work we build on rVQ–VAEs, not deterministic VQ–VAEs, both due to their demonstrated superior performance in maximising $\mathcal{L}$ and as they truly have probabilistic structure. In the rest of this paper, 'VQ–VAE' is used both to refer generically to either Vanilla (i.e., deterministic) VQ–VAEs and to rVQ–VAEs, as their properties and behaviours are broadly similar. When needing to refer to them distinctly, we do so.

3.3.2 Hierarchical VAEs

To make a hierarchical discrete VAE, introduce L layers of latent variables $\vec{\mathbf{z}} = \{\mathbf{z}_1, .., \mathbf{z}_L\}$ – $\mathbf{z}_\ell^m$ is the m^{th} latent variable in the ℓ^{th} layer. We can then impose some chosen structure in the generative and recognition models. Analogous to vanilla, single-layer VAEs, the conditional distributions in the generative and recognition models themselves have parameters parameterised by neural networks.

In the earliest hierarchical VAEs the generative and posterior models each factor as a Markov chain (Rezende et al., 2014) with the individual latent conditional distributions are Gaussian. As discussed in Sønderby et al. (2016); Zhao et al. (2017), this leads to a single path in the computational graph of such a model from $x \Rightarrow q_\phi(\mathbf{z}^1|\mathbf{x}) \Rightarrow \mathbf{z}^1 \Rightarrow \mathbb{E}_{q_\phi(\mathbf{z}^1|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}^1)$. Thus it is possible for a single-layer VAE to train in isolation within a hierarchical model of this type.

An advance came in Sønderby et al. (2016); Kingma et al. (2016), where it was proposed that the variational posterior should factorise as

$$q_\phi(\vec{\mathbf{z}}|\mathbf{x}) = q_\phi(\mathbf{z}_L|\mathbf{x}) \prod_{\ell=1}^{L-1} q_\phi(\mathbf{z}_\ell|\mathbf{z}_{>\ell}, \mathbf{x}). \tag{3.6}$$

Kingma et al. (2016), in proposing ResNet VAEs, also chose that the generative model's data likelihood should be conditioned on all layers of latent variables, with auto-regressive conditioning over the latents,

$$p_\theta\left(\mathbf{x}, \vec{\mathbf{z}}\right) = p_\theta\left(\mathbf{x}|\vec{\mathbf{z}}\right) p_\theta\left(\vec{\mathbf{z}}\right) = p_\theta(\mathbf{x}|\vec{\mathbf{z}}) p(\mathbf{z}_L) \prod_{\ell=1}^{L-1} p_\theta\left(\mathbf{z}_\ell|\mathbf{z}_{>\ell}\right). \tag{3.7}$$

Again, all latent conditional distributions are Gaussian in these models. These design choices stop the 'shortcut' dominating as it does in Markovian models. Subsequent to these works, numerous hierarchical VAEs have been proposed. Some, such as BIVA (Maaløe et al., 2019), have highly complex arrangements of latents in there generative and posterior networks. However, recent papers Child (2021); Vahdat & Kautz (2020) have latched onto using the same p and q factorisations as the ResNet VAEs in Kingma et al. (2016), Eqs (3.6,3.7). These papers all use fully convolutional neural parameterisations for the both the generative and recognition networks, so

like in a VQ-VAE the latents (within each layer) are laid out spatially. These recent works have shown that this choice of model factorisation with convolutional parameterisation can be used to give state of the art performance, outperforming pure autoregressive models (like PixelCNNs (van den Oord et al., 2016; Salimans et al., 2017; Chen et al., 2018b)) that previously were the highest-performance approach in image modelling using DGMs.

This choice of factorisation combined with convolutional parameterisation, while now ubiquitous in hierarchical VAEs – putting aside finer details implementations, for instance the exact choice of neural wiring, between different papers – does not have any particular name. For simplicity, and since the spatially-arranged convolutional latents are key to these models' performance, we will call hierarchical VAEs of this type *Spatial-VAEs*. This is essentially a ResNet VAE, but without inverse autoregressive flows in the posteriors (Kingma et al., 2016).

Definition 3.2. *(Kingma et al., 2016; Vahdat & Kautz, 2020; Child, 2021) A Spatial–VAE is a fully-convolutional hierarchical deep generative model with L layers of conditionally-Gaussian latent variables, with forward model as in Eq* (3.7) *and amortised variational posterior as in Eq* (3.6)*, trained w.r.t. its parameters* θ, ϕ *to maximise the objective:*

$$\mathcal{L}^{\text{Spatial}}(\mathcal{D}; \theta, \phi) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\mathbb{E}_{\vec{\mathbf{z}}\sim q} \log p_\theta\left(\mathbf{x}|\vec{\mathbf{z}}\right) - D_{\text{KL}}\left(q_\phi\left(\vec{\mathbf{z}}|\mathbf{x}\right)||p_\theta(\vec{\mathbf{z}})\right)\right]. \tag{3.8}$$

In this chapter in our discrete hierarchical models we use the same factorisation for our generative model and variational posterior, though of course unlike the models at hand our models will have discrete, not continuous, latent variables.

3.4 Related Work

VQ–VAEs have been extended to the two- and three-layer case (Razavi et al., 2019a), with large, powerful autoregressive models subsequently trained as priors to then sample from, producing draws competitive with the state of the art when combined with a classifier-based accept-reject algorithm.

Various recent papers have worked towards hierarchical discrete VAEs that eschew the training of priors as auxiliary models. One recent work trains layers of discrete latent variables in various hierarchical arrangements on MNIST and Fashion-MNIST (Liévin et al., 2019), building on variational memory addressing methods (Bornschein et al., 2017). In Hierarchical Quantised Autoencoders (Williams et al., 2020), much like in the original VQ–VAE paper, a sequential training pipeline is proposed. Here rVQ–VAEs are trained one at a time, with the first trained on the dataset and each subsequent sub-model trained on sampled values of the latents from the one below. This gives a Markovian structure both in the generative and inference networks. Pervez et al. (2020) perform inference over binary latents in single-latent-layer and hierarchical VAEs using a novel harmonic-analysis derived estimator; though in their results single-latent-layer VAEs trained this way have superior performance to those with a hierarchy of latents.

Methods have been developed to perform bits-back coding (Frey & Hinton, 1996) using the learnt representations of VAEs (Townsend et al., 2019), including for hierarchical VAEs (Townsend et al., 2020). In these methods the latents are continuous during training, with the space then subsequently bucketed. Recently flow-based models (Dinh et al., 2015, 2017; Papamakarios et al., 2019) have been extended to handle discrete variables (Hoogeboom et al., 2019; Tran et al., 2019).

As will discuss in §3.6.1, vector quantisation has close links to mixture models and mixtures of experts (Jacobs et al., 1991). Historically it has been known that stochastic relaxations of vector quantisation offer various benefits compared to deterministic assignment, and that they are equivalent to certain classes of mixture models (Hinton & Zemel, 1994). Vector quantisation can be thought of as inference on a Voronoi partition (Sack & Urrutia, 2000, §5). Our distributions are the responsibilities from a mixture model with learnt variances, so deterministic RRVQ would result in Mahalanobis-distance Voronoi partitions.

As discussed above, there is currently a renaissance in the usage of hierarchical VAEs for the modelling of high-dimensional data. Recently VAEs with hierarchies

of conditionally-Gaussian latents have enjoyed a resurgence. Precisely, VAEs with the same probabilistic 'wiring' as studied here (that of Kingma et al. (2016)) with deep hierarchies of latents obtain state-of-the-art performance (Vahdat & Kautz, 2020; Child, 2021).

3.5 Sampling and Reconstructing in VQ–VAEs

Here we focus on modelling square images, though the arguments we make can generalise to rectangular images, as well as to audio or video data. In VQ–VAEs, one uses convolutional neural networks to represent p and q, laying out $\mathbf{z}$ as a square of side $\sqrt{M}$, mirroring the spatial structure of pixels in an image (van den Oord et al., 2017). For audio one might choose a 1D structure, and 3D for video.

Interestingly, ancestral sampling from $p(\mathbf{z})$ in (relaxed or not) VQ–VAE models gives draws that do not resemble the training data. This indicates severe aggregate posterior–prior mismatch. Samples from this prior fail to capture the structure needed, i.e. the dependencies between the M latents that are necessary to produce realistic data when decoded. This is why in VQ–VAEs it is necessary to subsequently train a second density estimator, commonly a large, powerful autoregressive model such as a PixelCNN (van den Oord et al., 2016; Salimans et al., 2017) over the latent representations to then sample from. This is followed in the two-layer and three-layer extension of VQ–VAEs as well (Razavi et al., 2019a).

Conversely, in standard, non-hierarchical, VAEs with continuous latent variables the reconstructions are generally found to be somewhat blurry, while samples tend to have some coherent structure. In a standard VAE with $p(\mathbf{z}) = \prod_{i=1}^{M} \mathcal{N}\left(z^m|0, \mathbb{I}\right))$ the prior factorises over dimensions similar to how it does in a VQ–VAE, yet samples appear reasonable, which suggests that the reason is not only that.

We give an explanation for this phenomenon. It is not to do with discrete vs continuous latents at all, but rather with their neural parameterisation.

In VQ–VAEs, convolutional neural networks are used to represent p and q. With convolutionally-parameterised latents, each is tied spatially to be mostly concerned with a particular region of pixels in the input. That is, the latents are arranged spatially in a grid (of lower resolution than the input image) and the latents in, say, the top-left corner of the grid are concerned with the top-left corner of the input image. This comes directly from the use of convolutional neural networks to output these latents (in the encoder) or to map these latents to the data likelihood. The (often small) receptive field of convolutional layers means that the encoder output in a particular part of the grid of latents is a function of only a roughly-similarly-located patch of the input. The convolutional net that implements the data likelihood, the decoder, similarly will tie each region of the latents to the matched region of its output.

This is unlike most implementations of vanilla VAEs, where the posterior's parameters, commonly the mean and diagonal covariance of a Gaussian, are output by MLPs. Similarly the generative model will act on these given representations with MLPs. Those learnt representations are thus intrinsically non-local, which in turn gives them the ability to learn easily the arrangement of parts and wholes in an image.

To demonstrate this, we train a simple $L = 1$ *Spatial*–VAE where continuous-valued latent variables are arranged spatially, as in VQ–VAEs: $p_\theta(\mathbf{z}) = \prod_{m=1}^{M} \mathcal{N}(\mathbf{z}^m|\mathbf{0}, \mathbb{I})$ and $q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{m=1}^{M} \mathcal{N}(z^m|\mu_\phi^m(\mathbf{x}), \sigma_\phi^m(\mathbf{x}))$, $\mathbf{z}^m \in \mathbb{R}^{16}$, with p and q convolutional networks each composed of 2 ResNet block with 32 channels, and the number of latents M is the 1/4 the number of pixels in the input. We also train an equivalent rVQ–VAE, with embedding space dimensionality $d_e = K = 16$. We use SVHN, CIFAR-10, and (to make the effect most striking) a toy dataset containing images that are each uniform blocks of colours. See Fig 3.1 for the resulting reconstructions and samples for the three datasets for both models. We also provide examples of toy MLP-parameterised VQ-VAEs providing coherent samples in the appendix for this chapter.

Embedding an image into the latent space for reconstruction is relatively easy. For the discrete model, the encoder learns to output embeddings $\mathbf{e}_\phi(\mathbf{x})$ to place high probability over the right codebook embedding for its local region of the image, and do so over the entire set of M spatially-arranged latents. Similarly, the Spatial–VAE learns to place posterior probability over regions in the latent space appropriate for each latent position. However, when sampling from each model's prior, we end up with very mixed up generated images. Even for the toy dataset, the draws for both models are rainbow images where each patch of the image is separately given a random colour from the training set.

The poor quality of naive VQ–VAE draws is not intrinsically from having discrete latent variables, but from having discrete latent variables *that are arranged spatially and are parameterised in both the posterior and generative models using convolutional neural networks*. However, it is the choice to have spatial latent variables that provides high quality reconstructions. To get around this, one can train a powerful autoregressive model over samples from the aggregate posterior in $\mathbf{z}$. In Vanilla–VQ–VAEs the aggregate posterior is a sum of δ functions, so it resembles an empirical data distribution. Thus training a high-performance density estimator over aggregate posterior samples is reasonable, and provides realistic draws (van den Oord et al., 2017; Razavi et al., 2019a). In this manner of operation, the encoder-decoder networks can be viewed as tools for non-linear dimensionality-reduction, so that the density estimator can be trained in a lower-dimensional space, the learnt latent space, rather than on the raw data directly. While that is a proven approach, our goal is to combine the benefits of VQ–VAEs (high quality reconstructions, the desirable property of learning discrete representations, ease of training) with having a unified modelling approach, with models trained end-to-end.

We develop ways to make discrete VAEs more expressive and flexible by adding hierarchical structure. This removes the need of a two-stage training process, and gives us the benefits of hierarchical representations such as having different layers learning different aspects of the data. Further, if autoregressive models are used to produce samples, we are required to perform as many forward passes through

the model as there are latent variables. In the hierarchical case, as in Razavi et al. (2019a), this remains the case.

In this chapter we will be training very deep hierarchies of latent variables, up to 32 layers. Therefore, if we had autoregressive models for sampling in a hierarchical model of this form, the additional calls that would be needed to produce a single sample would be very demanding. For our deepest models trained on 32×32 images it would be ≈ 2000 internal, sequential forward passes. For 64×64 images it would be $\approx 10,000$[1]. Instead, with our approach, we are able to generate samples using a single forward pass.

3.6 Relaxed-Responsibility Hierarchical Discrete VAEs

We will factorise our q and p as in Eqs (3.6,3.7). This is directly analogous to hierarchical VAEs with continuous latent variables, but here we will be using a novel variety of vector quantisation to define our models.

3.6.1 Relaxed-Responsibility Vector-Quantisation

Our first main contribution is a method of parameterising the generative model and approximate posterior for models containing vector-quantised discrete latents, which improves the ability of hierarchical models of this type to learn effectively. We call this method *Relaxed-Responsibility Vector-Quantisation* (RRVQ). We found that without these improvements models of this form had low performance and were often unstable during training, and that the two changes we propose are synergistic – working better together than either alone.

3.6.1.1 Proposal for q

Vector-Quantisation has historic links to mixture models, mixtures of experts, and classical methods of inference. The exponential moving average method of updating

[1]These numbers are the total numbers of individual latents in our models.

the codebook in VQ–VAEs is closely linked to K-means (MacQueen, 1967). rVQ is linked to mean-field variational inference for a mixture of Gaussians: we can interpret the embedding codebook as recording the means of the cluster components, all having isotropic unit variance, and are a-priori equal in probability (Bishop, 2006, Sec 10.2). Eq (3.4) is equivalent to saying that the posterior at each position in $\mathbf{z}$ is equal to the cluster responsibilities for the embedding vector $\mathbf{e}_\phi(\mathbf{x})$ at that position.

We develop this link further, increasing the expressiveness of the parameterisation of the latents $\mathbf{z}_\ell$ in our hierarchical model, by relaxing the restriction that all components have unit isotropic covariance. We introduce a second codebook $\mathbf{E}_{\Sigma,\ell}$ for each layer, recording the diagonal covariance matrices of each component. The responsibilities then used for defining $\pi_{\phi,\ell}(\mathbf{e}_{\phi,\ell})$ are

$$\pi_{\phi,\ell}^{m,k}(\mathbf{e}_{\phi,\ell}, \mathbf{E}_{\Sigma,\ell})) \propto \frac{1}{\sqrt{(2\pi)^{d_e}\, \mathbf{E}_{\Sigma,\ell}^k}} \exp\left(-\frac{1}{2\mathbf{E}_{\Sigma,\ell}^k} \left|\mathbf{e}_{\phi,\ell}^m - \mathbf{E}_{\mu,\ell}^k\right|_2^2\right), \tag{3.9}$$

where m indexes over the latent positions, k over the codebook entries, $\mathbf{e}_{\phi,\ell} \in \mathbb{R}^{M \times d_e}$, $\mathbf{E}_{\mu,\ell}$ is the codebook of means for the ℓ^{th} layer and $\mathbf{e}_{\phi,\ell}$ is the embedding-space output of a network taking the appropriate inputs for the current layer, as written in Eq (3.6).

Viewing VQ as a mixture-of-experts model (Jacobs et al., 1991), where each codebook embedding mean is a local expert, we can view this extension as allowing the neighbourhoods of different experts to be more diffuse or more concentrated. By learning $\mathbf{E}_{\Sigma,\ell}$, codebook embeddings with large associated diagonal covariance will have their means used preferentially when the model outputs embeddings $\mathbf{e}_{\phi,\ell}$ are far away from the codebook means, and those with small diagonal covariance will dominate at short ranges, being highly confident of being the appropriate expert when $\mathbf{e}_{\phi,\ell}$ is close.

3.6.1.2 Proposal for p

What about for the generative model? One obvious approach is to parameterise the (log) probabilities of $p_\theta(\mathbf{z}_\ell|\mathbf{z}_{>\ell})$ directly by a deep net. However, we found

training to be unstable in hierarchical VQ–VAEs that directly parameterised these conditional probabilities.

Training instability in VAEs come from large D_{KL} values that then lead to numerical overflow and large gradients. These large D_{KL} values come from highly-confident distributions that have limited overlap between them, i.e low entropy distributions. These highly-confident distributions are the result of the underlying neural network outputs taking large-magnitude values. It is thus reasonable to hope that distributions that have higher-entropy when given large neural network outputs will lead to more stable training.

What is a reasonable, flexible form for the generative model that will provide stable training while preserving or even improving performance? We might expect rVQ parameterisation of discrete variables to lead to less-peaked, higher-entropy distributions, than a naive implementation using a softmax of raw logits – the method we found to be unstable.

To that end, we consider the functional form of the entropy (i) of rVQ-parameterised categorical distributions and (ii) of categorical distributions obtained directly as the softmax of a vector of logits. For simplicity, for rVQ we consider the case $\mathbf{E}_\Sigma = \mathbf{1}$, i.e. Eq (3.4), but in § 3.7.3 we show that $p(\mathbf{z})$ parameterised either this way or with learnt $\mathbf{E}_\Sigma$ both lead to stable training – though, as we would expect, learning $\mathbf{E}_\Sigma$ increases performance. In Theorems 3.1 and 3.2 we consider the worst-case arrangement of codebook means/logits for rVQ and Softmax respectively, such that a single large-magnitude value of the underlying network outputs has maximum impact driving the resulting discrete distribution to be close to one-hot.

Theorem 3.1. *(Minimum entropy from rVQ) Consider the worst-case arrangement of rVQ codebooks vectors, i.e. resulting in the minimum entropy categorical distribution: all but one of the codebook embeddings are an equal and greater distance away from the input embedding. For large-magnitude input embeddings of distance d from the solitary, closest codebook embedding along the line of separation and the remaining $K-1$ codebook embeddings at a distance $d+\delta$ along the same*

line of separation, the entropy of the resulting categorical distribution, Eq (3.4) *is, to first order*

$$\mathcal{H}_{\text{rVQ}} = (K-1)\,(1+g)\exp(-g) + O\left(\exp(-g)^2\right) \tag{3.10}$$

where $g = \left(\frac{\delta^2}{2} + \delta d\right)$.

Proof: See Appendix B.4.1.

Theorem 3.2. *(Minimum entropy from Softmax) Consider the worst-case arrangement of logits, i.e. resulting in the minimum entropy categorical distribution: all but one of the logits take the same value* c*, with one logit taking the larger value* $c+\ell$*,* $\ell > 0$*. For large-magnitude difference in logits* ℓ*, the entropy of the resulting categorical distribution is, to first order,*

$$\mathcal{H}_{\text{softmax}} = (K-1)\,(1+\ell)\exp(-\ell) + O\left(\exp(-\ell)^2\right) \tag{3.11}$$

Proof: See Appendix B.4.2.

We computationally verify the tightness of these first order approximations in Appendix B.4.3 and find them to be very accurate, quickly become correct to one part in 10^{-6}.

Corollary 3.2.1. *Viewing* $\ell + c = d$ *as the large-magnitude output of a neural network, for large* d *rVQ-parameterised categorical distributions have higher entropy than softmax-parameterised ones if* $\delta < 1$*.*

This corollary comes from subbing $\ell = d - c$ into Eq (3.11) and then taking the ratio between the dominant terms in Eq (3.10) and Eq (3.11), which is greater than 1 for large d if $\delta < 1$. This result tells us that, for large neural network outputs, rVQ-parameterised distributions have higher entropy than those parameterised via logits, even for the most unlucky arrangement of codebook embeddings, as long as the largest distance between codebooks is < 1.

Thus we choose to parameterise the conditional distributions in $p_\theta(\vec{\mathbf{z}})$ along the same lines as for q, namely rVQ-parameterisation via an embedding space, rather than directly using raw logits, and sharing the same codebooks as for q. So, in the generative model each conditional distribution in $p_\theta(\vec{\mathbf{z}})$, rather than receiving the

parameters needed to define it directly, instead has its probabilities parameterised via the responsibilities given by embeddings $\mathbf{e}_{\theta,\ell} \in \mathbb{R}^{M \times d_e}$ output by a deep net:

$$\pi_{\theta,\ell}^{m,k}\left(\mathbf{e}_{\theta,\ell}\right) \propto \frac{\exp\left(-\frac{1}{2\mathbf{E}_{\Sigma,\ell}^{k}}\left|\mathbf{e}_{\theta,\ell}^{m}-\mathbf{E}_{\mu,\ell}^{k}\right|_2^2\right)}{\sqrt{\left(2\pi\right)^{d_e}\mathbf{E}_{\Sigma,\ell}^{k}}}. \tag{3.12}$$

3.6.2 Overall Model

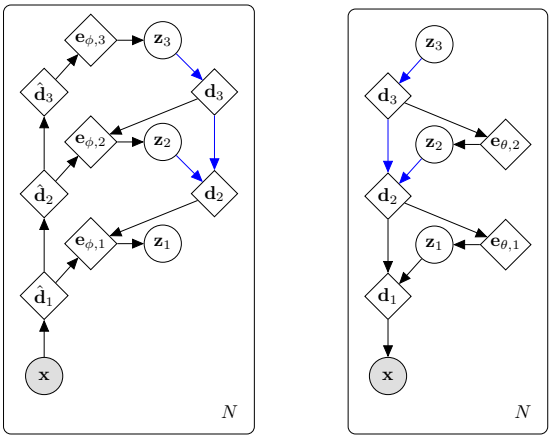
By combining Relaxed-Responsibility VQ with a hierarchical discrete VAE structure, we obtain our proposed model, a Relaxed-Responsibility Vector Quantised VAE (RRVQ–VAE).

Definition 3.3. *A Relaxed-Responsibility Vector Quantised VAE is a fully-convolutional hierarchical deep generative model with L layers of discrete latent variables. The forward model factorises as in Eq* (3.7) *and amortised variational posterior as in Eq* (3.6)*. The discrete latents are parameterised by relaxed-responsibility vector quantisation, Eq* (3.9) *for q and Eq* (3.12) *for p. The model is trained w.r.t. its parameters* $\theta, \phi, \mathbf{E}_\mu, \mathbf{E}_\Sigma$ *to maximise the objective:*

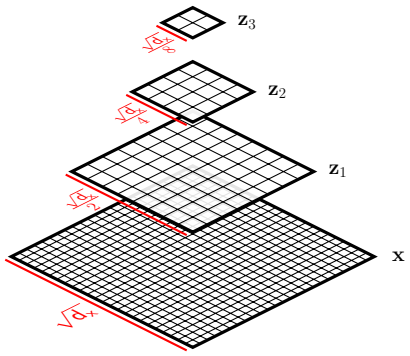
$$\begin{aligned}\mathcal{L}^{\mathrm{RRVQ}}(\mathcal{D};\theta,\phi,\mathbf{E}_\mu,\mathbf{E}_\Sigma) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\Big[\,&\mathbb{E}_{\vec{\mathbf{z}}\sim q}\log p_\theta\left(\mathbf{x}|\vec{\mathbf{z}};\mathbf{E}_\mu,\mathbf{E}_\Sigma\right)\\ &- D_{\mathrm{KL}}\left(q_\phi\left(\vec{\mathbf{z}}|\mathbf{x};\mathbf{E}_\mu,\mathbf{E}_\Sigma\right)||p(\vec{\mathbf{z}};\mathbf{E}_\mu,\mathbf{E}_\Sigma)\right)\Big].\end{aligned} \tag{3.13}$$

See Fig 3.2 for a graphical representation of this model. There is a deterministic upwards chain in the inference network, the representations $\{\hat{\mathbf{d}}_\ell\}$. Similarly, there is a deterministic downwards chain of representations $\{\mathbf{d}_\ell\}$ in the generative model. These representations enable the conditional structure given in Eqs (3.6-3.7): that in the generative model we have an autoregressive structure over layers, and similarly that in the posterior each layer of latents is conditioned both on $\mathbf{x}$ and on those above it in the hierarchy. We choose to have a progressively smaller number of latent variables per layer as we ascend the hierarchy. If we continue decreasing the number until the top-most latent is a single discrete variable, it is reasonable for us to place a uniform categorical prior over it. Following continuous VAE models, including Ladder-VAEs (Sønderby et al., 2016), ResNet-VAEs (Kingma

et al., 2016), BIVA (Maaløe et al., 2019), and recent papers (Vahdat & Kautz, 2020; Child, 2021), we enforce weight sharing between the generative and inference networks, indicated by blue arrows.



(a) Variational Posterior **(b)** Gen. Model



(c) Diagrammatic Representation

Figure 3.2: RRVQ–VAE with $L = 3$, (a) the variational posterior and (b) generative model, as defined in Eq (3.8). Blue arrows indicate shared networks. For simplicity the codebooks are not represented. (c) is a diagrammatic representation of the model, showing the spatial arrangement of latents, whose multiplicity we decrease by a factor of 4 each layer. As described above, the deterministic variables $\hat{\mathbf{d}}$ and $\mathbf{d}$ are present to give the required autoregressive factorisation over layers, for q and p respectively.

3.7 Experiments

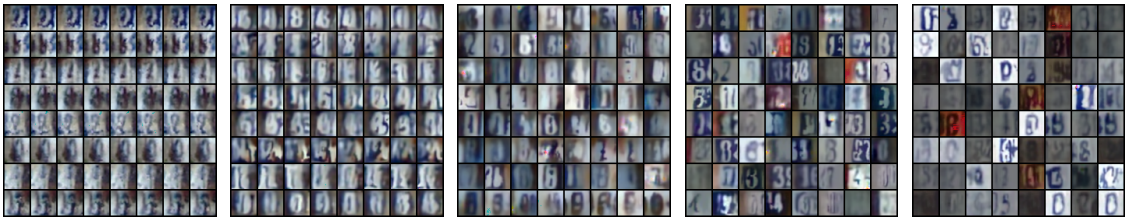
We train our models on CIFAR-10, SVHN and CelebA. We train very deep models with $L = 32$ layers, as well as smaller $L = 5$ models for visualisation and ablation studies. For each, the models for CIFAR-10 and SVHN have identical specification,

Table 3.1: Bits Per Dim Results: Comparison of our model, RRVQ–VAE, to various baselines in bits-per-dim (bpd) for train & test sets – lower better. We benchmark against rVQ–VAEs, VIMCO-trained discrete VAEs (van den Oord et al., 2017) and FouST-trained models with binary latents and $L = 1$ or $L = 4$ layers (Pervez et al., 2020). For additional context we also give values for hierarchical Spatial-VAEs, the conditionally-Gaussian latent variable version of our models.

Model	Test bpd	Train bpd
CIFAR-10		
VIMCO	5.14	-
rVQ–VAE	4.77	4.87
FouST, $L = 4$	4.16	-
FouST, $L = 1$	4.02	-
RRVQ–VAE, $L = 32$	**3.94**	**3.81**
Spatial–VAE, $L = 32$	3.55	3.49
SVHN		
rVQ–VAE	3.73	4.17
RRVQ–VAE, $L = 32$	**2.30**	**2.52**
Spatial–VAE, $L = 32$	1.94	2.07
CelebA		
rVQ–VAE	5.31	5.31
RRVQ–VAE, $L = 32$	**2.97**	**2.97**
Spatial–VAE, $L = 32$	2.54	2.58

with some small changes for CelebA due to the different image size. We implement these models using fully convolutional networks composed of ResNet blocks.

The number of latent variables per layer decreases as we ascend the hierarchy, as represented in Fig 3.2(c). For the $L = 32$ model we decrease the number of latents by a factor of 4 every 8 layers, forming 4 blocks each of decreasing numbers of latents. For $L = 5$ models we reduce the number of latents by a factor of 4 each layer. Each layer of latent variables has its own pair of codebooks for means and diagonal covariances. For further model description and implementation details, see Appendix B.3.



(a) $\ell = 1$ **(b)** $\ell = 2$ **(c)** $\ell = 3$ **(d)** $\ell = 4$ **(e)** $\ell = 5$

Figure 3.3: Layerwise sampling in 5 layer RRVQ–VAE trained on SVHN. Note that layer $\ell = 2$ seems to represent digit identity: resampling in this layer changes digit identity while keeping the rest of the image roughly the same.

3.7.1 Numerical Results

We show in Table 3.1 numerical results from our $L = 32$ models, benchmarked against rVQ–VAEs and various baselines. We measure the bits-per-dim (bpd) for the training and test set (using non-relaxed categorical distributions).

From one end of the spectrum, our baseline is a non-hierarchical rVQ–VAE trained with a set of uniform categorical priors. From the other end, we also train the continuous version of our hierarchical model, *Spatial-VAEs*, where we have Gaussian latent variables rather than codebook embeddings at each latent position. We also benchmark against the results in Pervez et al. (2020): single-latent-layer and hierarchical VAEs with binary latents trained using FouST, a spectral version of the straight-through estimator, for CIFAR-10. The results show clearly the benefit our approach brings to discrete VAEs, from the improved values reached of test and train bits-per-dim. Our models help close the gap between discrete latent variable models and those with continuous latent variables.

3.7.2 Analysis of Samples and Representations

Table 3.2: Digit Classification Results: SVHN embeddings from the $\ell = 2$ layer of our $L = 5$ model and from an $L = 1$ baseline were each used to train small convnets. We give the test-set accuracy over 4 runs.

Model	Test Set Acc
rVQ–VAE	0.578 ± 0.008
RRVQ–VAE, $L = 5$	$\mathbf{0.626 \pm 0.007}$

In addition to the 32 layer models, we also trained a 5 layer version for simpler plotting and analysis. Fig 3.3 demonstrates the effect each hierarchical layer has on the final draws when we train this smaller model on SVHN. For this figure we sample repeatedly (plotted along each row) in each layer, conditioned on sampled value of all the layers above (each set of these values shown in a different row). We then propagate deterministically down through the layers below – we take the mode for each subsequent layer.

The resulting latent structure indicates that different layers are representing different aspects of the data: the layers showing a degree of separation in their purpose. For instance it might seem that $\ell = 2$ describes digit identity. To verify this, we trained simple convnets on the embeddings from the $L = 1$ rVQ baseline and those from the $\ell = 2$ layer of this model. From the results in Table 3.2 we can see that digit identity is concentrated in the $\ell = 2$ layer. The rVQ–VAE provides good reconstructions, so its embeddings do, necessarily, encode digit identity, yet the $\ell = 2$ layer (which does not encode all information for reconstruction) makes digit identity easier for a convnet to ascertain. Further to this, Fig B.6 shows reconstructions and Fig B.7 ancestral samples for our $L = 32$ models. We also experiment with using these $L = 5$ models for compression – see Appendix B.5.

3.7.3 Ablation Study

How does our approach compare to other possible hierarchical extensions of rVQ–VAEs? For the $L = 5$ models we trained various ablations of our proposal: with or without a learnt codebook of covariances; and with the generative model represented either via a Relaxed-VQ lookup or directly outputting a (log) probability over embeddings. Thus all of these have an ELBO as in Eq (3.8), but vary in how we parameterise p and q.

In Table 3.3 we show the test and train bits-per-dim (bpd) obtained for these hierarchical discrete VAEs. We can see that $\sigma = 1$ with *direct probabilities in p* (the top-left corner results for each dataset), arguably the most naive approach

Table 3.3: Ablation Study for $L = 5$ models on CIFAR-10 and SVHN: We show the train and test bits-per-dim We can have: the generative model log probabilities directly output by a net (*Direct-Cat in p*) or parameterised using responsibilities in the embedding space (*Embed-Cat in p*), and we can learn a codebook of diagonal covariances for the responsibilities (σ *learnt*) or have them all fixed to one ($\sigma = 1$). RRVQ is when we have *Embed-Cat in p* and σ *learnt*. Note that Direct-Cat with σ learnt is unstable during training for SVHN.

	Direct-Cat in p		Embedding-Cat in p	
CIFAR-10				
$\sigma = 1$	Train:	5.00	Train:	5.06
	Test:	5.05	Test:	5.11
σ learnt	Train:	5.08	Train:	**4.40** *(RRVQ)*
	Test:	5.10	Test:	**4.65** *(RRVQ)*
SVHN				
$\sigma = 1$	Train:	3.44	Train:	3.51
	Test:	3.32	Test:	3.41
σ learnt	Train:	–	Train:	**3.02** *(RRVQ)*
	Test:	–	Test:	**2.96** *(RRVQ)*

to a hierarchical VQ–VAE, is outperformed by approximately half of a bpd by full RRVQ (bottom-right corner). Interesting, either of the two changes made to obtain RRVQ made in isolation lead either no substantive change in performance (if anything, slight degradation) or rendered training so unstable that it was impossible to obtain a result. Clearly there is a synergistic property here, that the these two changes together lead to improved performance of the models.

3.8 Conclusion

We have presented a novel parameterisation for stochastic Vector Quantisation, Relaxed-Responsibility Vector Quantisation. RRVQ learns a codebook of variances alongside the codebook of means, using the responsibilities under the Gaussian mixture model represented by those quantities to define discrete distributions, both within the approximate posterior using for inference and in the forward model.

We then use this is as a building block to develop a novel variety of hierarchical discrete VAE, Relaxed-Responsibility Vector-Quantised VAEs. RRVQ–VAEs are the

highest-performance unified probabilistic deep generative models with hierarchies of discrete latent variables to be trained end-to-end on the datasets studied.

RRVQ–VAEs are highly expressive; their hierarchy of representations separate out different aspects of the data. Helpfully, they are stable during training. Their capacity and flexibility is demonstrated by them reaching the lowest bits-per-dim results for models of their type, and do so without training a secondary autoregressive prior over posterior samples. Further, they avoid the large number of forward passes that a hierarchical model of that form would need to produce a single sample. Models of that type would require a separate forward pass for each individual latent in each layer, reminisicent of a PixelCNN say.

We hope that this work inspires further research into discrete hierarchical VAEs, with the aim of completely closing the gap between hierarchical discrete VAEs and those with continuous latent variables.

If you want to encourage someone to do something, make it easy.

— Richard Thaler

4

Learning Bijective Feature Maps for Linear ICA

Contents

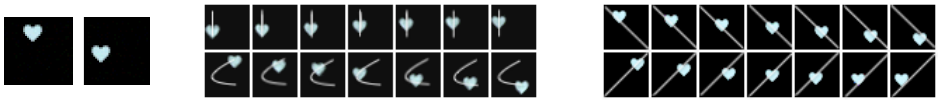
4.1 Motivation

Separating high-dimensional data like images into independent latent factors, i.e independent component analysis (ICA), remains an open research problem. As we show, existing probabilistic deep generative models, which are tailor-made for image data, underperform on non-linear ICA tasks. To address this, we propose a DGM which combines bijective feature maps with a linear ICA model to learn interpretable latent structures for high-dimensional data. Given the complexities of jointly training such a hybrid model, we introduce novel theory that constrains linear ICA to lie close to the manifold of orthogonal rectangular matrices, the Stiefel manifold. By doing so we create models that converge quickly, are easy to train, and achieve better unsupervised latent factor discovery on images than each of flow-based models, linear ICA, and Variational Autoencoders.

4.2 Introduction

In linear Independent Component Analysis (ICA), data is modelled as having been created from a linear mixing of independent latent sources (Cardoso, 1989a,b, 1997; Jutten & Herault, 1991; Comon, 1994; Bell & Sejnowski, 1995). The canonical problem is blind source separation; the aim is to estimate the original sources of a mixed set of signals by learning an *unmixing* matrix, which when multiplied with data recovers the values of these sources. While linear ICA is a powerful approach to undo the mixing of signals like sound (Everson & Roberts, 2001; Hyvärinen et al., 2001), it has not been as effectively developed for learning compact representations of high-dimensional data like images, where the linear assumption is limiting. Non-linear ICA methods, which assume the data has been created from a non-linear mixture of latent sources, offer better performance on such data.

In particular, flow-based models have been proposed as a non-linear approach to *square* ICA, where we assume the dimensionality of our latent source space is the same as that of our data (Deco & Brauer, 1995; Parra et al., 1995; Dinh et al., 2015,



(a) Sample Images **(b)** VAE Sources **(c)** Bijecta Sources

Figure 4.1: Here we take a dSprites heart and, using a randomly sampled affine transformation, move it around a black background (a). The underlying sources of the dataset are *affine* transformations of the heart. In (b-c) images in the centre correspond to the origin of the learnt source space. Images on either side correspond to linearly increasing values along one of the learnt latent sources whilst the other source remains fixed. Bijecta (c) has learned affine transformations as sources (white diagonals), whereas a VAE (with ICA-appropriate prior) (b) has learned non linear transforms (white curves). The VAE has not discovered the underlying latent sources.

2017). Flows parameterise a bijective mapping between data and a feature space of the same dimension and can be trained under a maximum likelihood objective for a chosen base distribution in that space. While these are powerful generative models, for image data one typically wants fewer latent variables than the number of pixels in an image. In such situations, we wish to learn a *non-square* (that is, not dimensionality preserving) ICA representation.

In this work, we highlight the fact that existing probabilistic deep generative models, in particular Variational Autoencoders, underperform on non-linear ICA tasks, trained unsupervised[1]. As such there is a real need for a probabilistic DGM that can perform these tasks. To address this we propose a novel methodology for performing non-square non-linear ICA using a model, termed *Bijecta*, with two jointly trained parts: a highly-constrained non-square linear ICA model, operating on a feature space output by a bijective flow. The bijective flow is tasked with learning a representation for which linear ICA is a good model. It is as if we are *learning the data* for our ICA model.

We find that such a model fails to converge when trained naively with no constraints. To ensure convergence, we introduce novel theory for the parameterisation of

[1]Recently there has been a renaissance in results for non-linear ICA using deep generative models not in the pure unsupervised setting, the setting we are in here, but rather *where some sufficiently-informative side information is available*. See Khemakhem et al. (2020a) for more, including early indications, that in part motivated this work, that VAEs, without this side-information, do not perform well at non-linear ICA.

decorrelating, non-square ICA matrices that lie close to the Stiefel manifold (Stiefel, 1935), the space of orthonormal rectangular matrices. We use this result to introduce a novel non-square linear ICA model that uses Johnson-Lidenstrauss projections, a family of randomly generated matrices (Johnson & Lindenstrauss, 1984; Woodruff, 2014). Using these projections, Bijecta successfully induces dimensionality reduction in flow-based models and scales non-square non-linear ICA methods to high-dimensional image data. Further we show that it is better able to learn independent latent factors than each of its constituent components in isolation and VAEs. For a preliminary demonstration of the inability of VAEs and the ability of Bijecta to discover ICA sources see Fig 4.1 above.

4.3 Background

4.3.1 Independent Component Analysis

The goal of ICA is to learn a set of statistically independent sources that explain our data. ICA is a highly diverse modelling paradigm with numerous variants: learning a mapping vs learning a model, linear vs non-linear, different loss functions, different generative models, and a wide array of methods of inference (Cardoso, 1989a; Mackay, 1996; Lee et al., 2000).

Here, in the manner of Mackay (1996); Cardoso (1997), we specify a generative model of appropriate form and find point-wise maximum likelihood estimates of model parameters. Concretely, we have a model with latent sources $\mathbf{s} \in \mathcal{S} = \mathbb{R}^{d_s}$ generating data $\mathbf{x} \in \mathcal{X} = \mathbb{R}^{d_x}$, with $d_s \leq d_x$. The linear ICA generative model factorises as

$$p(\mathbf{x}, \mathbf{s}) = p(\mathbf{x}|\mathbf{s})p(\mathbf{s}), \qquad p(\mathbf{s}) = \prod_{i=1}^{d_s} p(s_i),$$

where $p(\mathbf{s})$ is a set of independent distributions appropriate for ICA. In linear ICA all mappings are simple matrix multiplications, so the sources *cannot* be Gaussian distributions. Recall that we are mixing our sources to generate our data: A linear mixing of Gaussian random variables is itself Gaussian, so unmixing is impossible (Lawrence & Bishop, 2000). To be able to unmix, to break this symmetry, we can choose as our prior $p(\mathbf{s})$ any heavy-tailed or light-tailed non-Gaussian distribution

that gives us axis alignment and independence between sources. A common choice is the family of generalised Gaussian distributions,

$$p(s_i) = \mathrm{GG}(s_i|\mu, \alpha, \rho) = \frac{\rho}{2\alpha\Gamma(1/\rho)} \exp\left[\left(-\frac{|s_i - \mu|}{\alpha}\right)^{\rho}\right] \tag{4.1}$$

with mean μ, scale α and shape ρ. For $\rho = 2$ we recover the Normal distribution, and for $\rho = 1$ we have the (heavy-tailed) Laplace. As $\rho \to \infty$ the distribution becomes increasingly sub-Gaussian, tending to a uniform distribution. As such, the generalised Gaussian is a flexible framework for specifying ICA-appropriate priors as it allows for the specification of a sub or super Gaussian distribution by way of a single parameter, ρ (Roberts, 1998).

4.3.2 Manifolds for the unmixing matrix

In linear ICA we want to find the linear mapping $\mathbf{A}^+$ resulting in *maximally independent* sources. This is more onerous than merely finding decorrelated sources, as found by principal component analysis (PCA).

When learning a linear ICA model we typically have the mixing matrix $\mathbf{A}$ as the (pseudo)inverse of the unmixing matrix $\mathbf{A}^+$ and focus on the properties of $\mathbf{A}^+$ to improve convergence. $\mathbf{A}^+$ linearly maps from the data-space $\mathcal{X}$ to the source space $\mathcal{S}$. It can be decomposed into two linear operations. First we *whiten* the data such that each component has unit variance and these components are mutually uncorrelated. We then apply an orthogonal transformation and a scaling operation (Hyvärinen et al., 2001, §6.34) to 'rotate' the whitened data into a set of coordinates where the sources are independent *and* decorrelated. Whitening on its own is not sufficient for ICA — two sources can be uncorrelated *and* dependent (see Appendix C.1).

Thus we can write the linear ICA unmixing matrix as

$$\mathbf{A}^+ = \mathbf{\Phi}\mathbf{R}\mathbf{W}, \tag{4.2}$$

where $\mathbf{W} \in \mathbb{R}^{d_s \times d_x}$ is our whitening matrix, $\mathbf{R} \in \mathbb{R}^{d_s \times d_s}$ is an orthogonal matrix and $\mathbf{\Phi} \in \mathbb{R}^{d_s}$ is a diagonal matrix. Matrices that factorise this way are known

as the *decorrelating matrices* (Everson & Roberts, 1999): members of this family decorrelate through $\mathbf{W}$, and $\mathbf{\Phi R}$ ensures that sources are statistically independent, not merely uncorrelated. The optimal ICA unmixing matrix is the decorrelating matrix that decorrelates *and* gives independence.

4.3.3 Flows

Flows are models that stack numerous invertible changes of variables. One specifies a simple base distribution and learns a sequence of (invertible) transforms to construct new distributions that assign high probability to observed data. Given a variable $\mathbf{z} \in \mathcal{Z} = \mathbb{R}^{d_x}$, we specify the distribution over data $\mathbf{x}$ as

$$p(\mathbf{x}) = p(\mathbf{z}) \left\| \det \frac{\partial f^{-1}}{\partial \mathbf{x}} \right\|, \tag{4.3}$$

where f is a bijection from $\mathcal{Z} \to \mathcal{X}$, i.e. $\mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$, and $p(\mathbf{z})$ is the base distribution over the latent $\mathbf{z}$ (Rezende & Mohamed, 2015; Papamakarios et al., 2019). As such, one trains these models via pure maximum likelihood – no approximate method of inference is needed, setting them apart from other DGMs such as VAEs.

For more flexible distributions for $\mathbf{x}$, we specify $\mathbf{x}$ through a series of composed functions, from our simple initial p into a more complex multi-modal distribution; for example for a series of $K+1$ mappings, $\mathbf{z} = f_K \circ ... \circ f_0(\mathbf{x})$. By the properties of determinants under function composition

$$p(\mathbf{x}) = p(\mathbf{z}_K) \prod_{i=0}^{K} \left\| \det \frac{\partial f_{i+1}^{-1}}{\partial \mathbf{z}_i} \right\|, \tag{4.4}$$

where $\mathbf{z}_{i+1}$ denotes the variable resulting from the transformation $f_i(\mathbf{z}_i)$, $p(\mathbf{z}_K)$ defines a density on the K^{th}, and the bottom most variable is our data ($\mathbf{z}_0 = \mathbf{x}$).

Computing the determinant of the Jacobian ($\det \frac{\partial f^{-1}}{\partial \mathbf{z}}$) in Eq (4.3) can be prohibitively costly, especially when composing multiple functions as in Eq (4.4). To address this, flows use *coupling layers* that enforce a lower triangular Jacobian such that the determinant of the Jacobian is simply the product of its diagonal elements. We use recently proposed coupling layers based on rational quadratic

splines (RQS) to enforce this lower triangular structure (Durkan et al., 2019). They form highly flexible flows that typically require fewer composed mappings to achieve good performance relative to other coupling layers. See Appendix C.4 for details.

4.3.4 Variational Autoencoders for ICA?

Variational Autoencoders seem like a natural fit for learning a compressed set of statistically independent latent variables (Kingma & Welling, 2014; Rezende et al., 2014). It seems natural to train a VAE with an appropriate non-Gaussian prior, and expect that it would learn an appropriate ICA model. However, this is not the case. In Khemakhem et al. (2020a) some experiments suggest that VAEs with ICA-appropriate priors are unsuited to performing non-linear ICA. In our experiments (§4.8) we further verify this line of inquiry and show that VAEs struggle to match their aggregate posteriors to non-Gaussian priors and thus are unable to discover independent latent sources.

Disentangling (Bengio et al., 2013), potentially synonymous with non-linear non-square ICA, occurs when there is a one-to-one correspondence between dimensions of a latent space and interpretable aspects of the data(Higgins et al., 2017a; Burgess et al., 2017; Chen et al., 2018a; Mathieu et al., 2019). One dimension of the latent space could encode the rotation of a face for instance.

Disentangling is often enforced by an added penalisation to the VAE ELBO that acts akin to a regularisation method, such as the β-VAE (Higgins et al., 2017a) and β-TCVAE (Chen et al., 2018a). In a β-VAE one simply upweights the D_{KL} term in the ELBO by a factor β, with the prior and variational posterior defined as in a vanilla VAE.

Definition 4.1. *(Higgins et al., 2017a) A β-VAE is a deep generative model with forward model as in Eqs (1.34-1.36) and amortised variational posterior as in Eq* (1.37)*, trained w.r.t. its parameters θ, ϕ to maximise the objective*

$$\mathcal{L}^{\beta-\mathrm{VAE}}(\mathcal{D}; \beta, \theta, \phi) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \beta D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\right] \quad (4.5)$$

for a chosen, fixed value of β.

In a β-TCVAE (Chen et al., 2018a) one upweights the Total Correlation (TC) (Watanabe, 1960) of the aggregate posterior. Intuitively, the TC measures how well a distribution is approximated by the product of its marginals – that is, how much information is shared between variables due to dependence – and is often used as the objective Independent Component Analysis (Bell & Sejnowski, 1995).

Definition 4.2. *(Watanabe, 1960) The total correlation (TC) is a generalisation of mutual information to multiple variables. The TC is defined as is defined as the KL divergence from the joint distribution $p(\mathbf{s}), \mathbf{s} \in \mathbb{R}^d$ to the independent distribution over the dimensions of the variable $\mathbf{s}$: $p(\mathbf{s}_1)p(\mathbf{s}_2)\dots p(\mathbf{s}_n)$. Formally:*

$$\mathrm{TC}(\mathbf{s}) = D_{\mathrm{KL}}\left(p(\mathbf{s})||\prod_{j=1}^{d} p(\mathbf{s}_j)\right).$$

This enables us give the definition for a β-TCVAE.

Definition 4.3. *(Chen et al., 2018a) A β-TCVAE is a deep generative model with forward model as in Eqs (1.34-1.36) and amortised variational posterior as in Eq* (1.37)*, trained w.r.t. its parameters θ, ϕ to maximise the objective*

$$\begin{aligned}\mathcal{L}^{\beta-\mathrm{TCVAE}}(\mathcal{D};\beta,\theta,\phi) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\right] \\ - (\beta-1)D_{\mathrm{KL}}\left(q_\phi(\mathbf{z})||\prod_{j=1}^{d_z} q_\phi(\mathbf{z}_j)\right) \qquad (4.6)\end{aligned}$$

for a chosen, fixed value of β.

We will see both β-VAEs and β-TCVAEs again in subsequent chapters.

Disentangling can be difficult to achieve in practice, and requires precisely choosing the hyperparameters of the model *and* of the weighting of the added regularisation term (Locatello et al., 2019; Mathieu et al., 2019; Rolinek et al., 2019). That disentangling relies on forms of soft supervision renders the task of learning disentangled representations potentially problematic (Khemakhem et al., 2020a). When viewed as a purely unsupervised task it can be hard to establish a direct correspondence between a disentangling-VAE's training objective and the learning of a disentangled latent space. Further, at times these methods induce improper priors (in the β-TCVAE in particular (Mathieu et al., 2019)).

Stühmer et al. (2020) obtains a variety of non-linear ICA using VAEs with sets of Generalised Gaussian priors and their extensions. This approach offers advantages over β-VAEs and β-TCVAEs, for example reducing the reconstruction quality – disentangling tradeoff. Even then, however, β penalisation is still advantageous within this approach in helping to obtain disentangled representations.

As such there is a need for probabilistic DGMs that can separate sources without added hyperparameter tuning and that can do so by matching ICA-appropriate priors.

4.4 Related Work

One approach to extend ICA to non-linear settings is to have a non-linear mapping acting on the independent sources and data (Burel, 1992; Deco & Brauer, 1995; Yang et al., 1998; Valpola et al., 2003). In general, non-linear ICA models have been shown to be hard to train, having problems of unidentifiability: the model has numerous local minima it can reach under its training objective, each with potentially different learnt sources (Hyvärinen & Pajunen, 1999; Karhunen, 2001; Almeida, 2003; Hyvarinen et al., 2019). Some non-linear ICA models have been specified with additional structure to reduce the space of potential solutions, such as putting priors on variables (Lappalainen & Honkela, 2000) or specifying the precise non-linear functions involved (Lee & Koehler, 1997; Taleb, 2002), Recent work shows that conditioning the source distributions on some always-observed side information, say time index, can be sufficient to induce identifiability in non-linear ICA (Khemakhem et al., 2020a).

Modern flows were first proposed as an approach to non-linear square ICA (Dinh et al., 2015), but are also motivated by desires for more expressive priors and posteriors (Kingma et al., 2016; Papamakarios et al., 2019). Early approaches, known as symplectic maps (Deco & Brauer, 1995; Parra et al., 1995, 1996), were also proposed for use with ICA. Flows offer expressive dimensionality-preserving (and sometimes volume-preserving) bijective mappings (Dinh et al., 2017; Kingma

& Dhariwal, 2018). Flows have been used to provide feature extraction for linear discriminative models (Nalisnick et al., 2019). Orthogonal transforms have been used in normalising flows before, to improve the optimisation properties of Sylvester flows (Van Den Berg et al., 2018; Golinski et al., 2019). Researchers have also looked at constraining neural network weights to the Stiefel-manifold (Li et al., 2020a; Choromanski et al., 2020).

4.5 Non-Square ICA using Flows

Our solution combines linear ICA with a dimensionality-preserving invertible flow f_θ. The flow acts between our data space of dimensionality and the representation fed to the linear ICA generative model; learning a representation that is well fit by the simple, linear ICA model. As we demonstrate in experiments (§4.8), this hybrid model, which we call Bijecta, succeeds where VAEs fail: it can match non-Gaussian priors and is able to discover independent latent sources on image datasets.

4.5.1 A Linear ICA base distribution for flows

Our aim here is to develop a non-square ICA method that is both end-to-end differentiable *and* computationally efficient, such that it can be trained jointly with a flow via stochastic gradient ascent. We begin by choosing our base ICA source distribution to be a set of independent generalised Gaussian distributions, Eq (4.1) with $\mu = 0$, $\alpha = 1$ and ρ varying per experiment; and the ICA model's likelihood to a be a Gaussian.

$$p(s_i) = \mathrm{GG}(s_i|\mu = 0, \alpha = 1, \rho), \text{for } i \in \{1, \ldots, d_s\},$$
$$p(\mathbf{z}|\mathbf{s}) = \mathcal{N}(\mathbf{z}|\mathbf{A}\mathbf{s}, \mathbf{\Sigma}_\theta),$$

where $\mathbf{A} \in \mathbb{R}^{d_x \times d_s}$ is our (unknown) ICA mixing matrix, which acts on the sources to produce a linear mixture; and $\mathbf{\Sigma}_\theta$ is a learnt or fixed diagonal covariance. This linear mixing of sources yields an intermediate representation $\mathbf{z}$ that is then mapped

to the data by a flow. Our model has three sets of variables: the observed data $\mathbf{x}$, the flow representation $\mathbf{z} = f^{-1}(\mathbf{x})$, and ICA latent sources $\mathbf{s}$. It can be factorised as

$$p_\theta(\mathbf{x}, \mathbf{s}) = p_\theta(\mathbf{x}|\mathbf{s})p(\mathbf{s}) = p(\mathbf{z}|\mathbf{s})p(\mathbf{s}) \left\| \det \frac{\partial f_\theta^{-1}}{\partial \mathbf{z}} \right\| \tag{4.7}$$

While it is simple to train a flow by maximum likelihood method when we have a simple base distribution in $\mathcal{Z}$, here to obtain a maximum likelihood objective we would have to marginalise out $\mathbf{s}$ to obtain the evidence in $\mathcal{Z}$; a computationally intractable procedure:

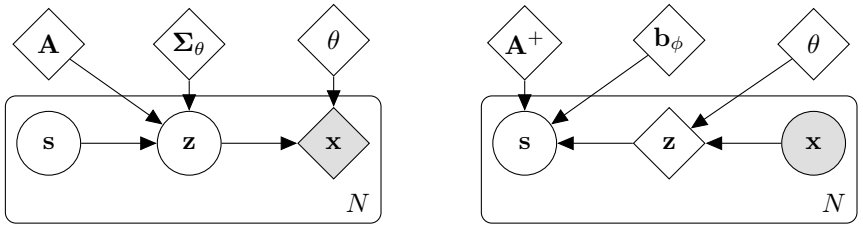
$$p(\mathbf{z}; \mathbf{A}, \mathbf{\Sigma}_\theta) = \int \mathrm{d}\mathbf{s}\, p(\mathbf{z}|\mathbf{s}; \mathbf{A}, \mathbf{\Sigma}_\theta)p(\mathbf{s}). \tag{4.8}$$

A contemporary approach is to use amortised variational inference for the linear ICA part of our model. In effect we introduce a linear VAE as a subcomponent to perform inference for this part of the model. This means we introduce an approximate amortised posterior for $\mathbf{s}$ and perform importance sampling on Eq (4.8), taking gradients through our samples using the reparameterisation trick (Kingma & Welling, 2014; Rezende et al., 2014). As discussed in previous chapters, amortised stochastic variational inference offers numerous benefits: it scales training to large datasets by using stochastic gradient descent, our trained model can be applied to new data with a simple forward pass, and we are free to choose the functional & probabilistic form of our approximate posterior. Further our ICA model is end-to-end differentiable, making it possible to jointly train with a flow.

We choose a linear mapping in our posterior, with $q_\phi(\mathbf{s}|\mathbf{z}) = \text{Laplace}(\mathbf{s}|\mathbf{A}^+\mathbf{z}, \mathbf{b}_\phi)$, where we have introduced variational parameters $\phi = \{\mathbf{A}^+, \mathbf{b}_\phi\}$ corresponding to an unmixing matrix and a diagonal diversity. Using samples from this posterior we can define a lower bound $\mathcal{L}$ on the evidence in $\mathcal{Z}$

$$\log p(\mathbf{z}; \mathbf{A}, \mathbf{\Sigma}_\theta) \geq \mathcal{L}(\mathbf{z}; \phi, \mathbf{A}, \mathbf{\Sigma}_\theta) = \mathbb{E}_{\mathbf{s}\sim q}\left[\log p(\mathbf{z}|\mathbf{s}) - D_{\mathrm{KL}}(q_\phi(\mathbf{s}|\mathbf{z})||p(\mathbf{s})\right) \tag{4.9}$$

Using the change of variables equation, Eq (4.3), and the lower bound on the evidence for ICA in (4.9) for $\mathcal{Z}$, we can obtain a variational lower bound on the



(a) Generative Model **(b)** Variational Posterior

Figure 4.2: The generative model (a) and variational posterior (b), as defined in Eq (4.10).

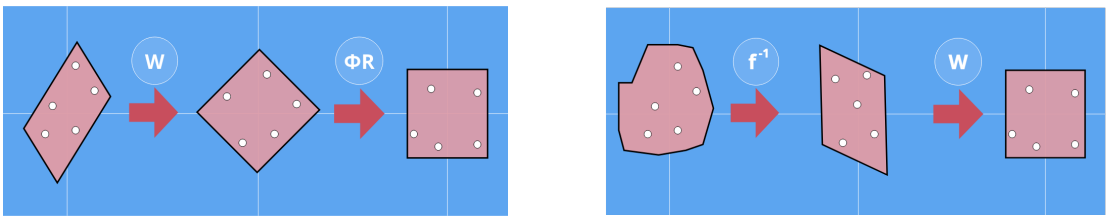
evidence for our data $\mathbf{x}$ as the sum of the ICA model's ELBO (acting on $\mathbf{z}$) and the log determinant of the flow:

$$\log p_\theta(\mathbf{x}; \mathbf{A}, \mathbf{\Sigma}_\theta) \geq \mathcal{L}(\mathbf{x}; \theta, \phi, \mathbf{A}, \mathbf{\Sigma}_\theta) = \mathcal{L}(\mathbf{z}; \phi, \mathbf{A}, \mathbf{\Sigma}_\theta) + \log \left\| \det \frac{\partial f_\theta^{-1}}{\partial \mathbf{z}} \right\| \tag{4.10}$$

As such our model is akin to a flow model, but with an additional latent variable $\mathbf{s}$; the base distribution $p(\mathbf{z})$ of the flow is defined through marginalising out the linear mixing of the sources. We refer to a model with n non-linear splines in the flow mapping from $\mathcal{X}$ to $\mathcal{Z}$ as an n-layer Bijecta model.

In the case of non-square ICA, where our ICA model is not perfectly invertible, errors when reconstructing a mapping from $\mathcal{S}$ to $\mathcal{Z}$ may amplify when mapping back to $\mathcal{X}$. To mitigate this we add an additional regularisation term in our loss that penalises the L_1 error of each point when reconstructed into $\mathcal{X}$. This penalisation can be weighted according to the importance of high-fidelity reconstructions for a given application.

We attempted to train Bijecta with unconstrained mixing and unmixing matrices, but found that jointly training a linear model with a powerful flow was not trivial and models failed to converge when naively optimising Eq (4.10). We found it crucial to appropriately constrain the unmixing matrix to get models to converge. We detail these constraints in the next section.



(a) Linear ICA **(b)** Bijecta

Figure 4.3: (a) Sequence of actions that are performed by the elements of $\mathbf{A}^+$, the unmixing matrix of linear ICA. $\mathbf{W}$ whitens the correlated data and $\mathbf{\Phi R}$ then ensures that the whitened (decorrelated) data is also independent. (b) Sequence of actions that are performed by the elements of Bijecta. f^{-1} maps data to a representation for which the whitening matrix *is* the ICA matrix. $\mathbf{W}$ now whitens $f^{-1}(\mathbf{x})$ and the result is *also* statistically independent.

4.6 Whitening, without looking

What are good choices for the mixing and unmixing matrices? Recall in Sec 4.3.2 we discussed various traditional approaches to constraining the unmixing matrix. For our flow-based model, design choices as to the parameterisation of $\mathbf{A}^+$ stabilise and accelerate training. As before, the mixing matrix $\mathbf{A}$ is unconstrained during optimisation. Without the constraints on $\mathbf{A}^+$ we describe in this section, however, we found that joint training of a flow with linear ICA did not converge.

Recall Eq (4.2) — linear ICA methods carry out whitening $\mathbf{W}$, performing dimensionality reduction projecting from a d_x-dimensional space to a d_s-dimensional space, and the remaining rotation and scaling operations are square. When training with a flow the powerful splines we are learning can fulfill the role of the square matrices $\mathbf{R}$ and $\mathbf{\Phi}$, but doing this ahead of the whitening itself. Put another way, the outputs from the flow can be learnt such that they are simply a whitening operation away from being effective ICA representations in $\mathcal{S}$. Thus, to minimise the complexities of jointly training a powerful flow with a small linear model, we can simply set $\mathbf{A}^+ = \mathbf{W}$, such that the unmixing matrix projects from d_z to d_s and is decorrelating. Statistical independence will come from the presence of the D_{KL} term in Eq (4.9): the flow will learn to give $\mathbf{z}$ representations that, when whitened,

are good ICA representations in $\mathcal{S}$. See Fig 4.3 for a visual illustration of this process and a comparison with the steps involved in linear ICA.

In previous linear ICA methods, the whitening procedure $\mathbf{W}$ has been derived in some data-aware way. A common choice is to whiten via the Singular Value Decomposition (SVD) of the data matrix, where $\mathbf{W} = \mathbf{\Sigma}^{-1}\mathbf{U}^T$, $\mathbf{\Sigma}$ is the rectangular diagonal matrix of singular values of $\mathbf{X}$, and the columns of $\mathbf{U}$ are the left-singular vectors. Computing the SVD of the whole dataset is expensive for large datasets; for us, in the context of Bijecta, we would be re-calculating the SVD of the representations $\mathbf{Z} = f^{-1}(\mathbf{X})$ of the entire dataset after every training step. One route around this would be online calculation of the whitening matrix (Cardoso & Laheld, 1996; Hyvärinen et al., 2001). This introduces an extra optimisation process that also has to be tuned, and would interact with the training of the flow.

To tackle these shortcomings of existing whitening methods, we propose a new method for linear non-square ICA that uses Johnson–Lindenstrauss (JL) transforms (also known as sketching) (Johnson & Lindenstrauss, 1984; Woodruff, 2014), which not only works effectively as a linear ICA method, but also works in conjunction with a flow model. These JL transforms have favourable properties for ICA, as we demonstrate in theoretical results. Further, this method samples part of the whitening matrix at initialisation and leaves it fixed for the remainder of training, requiring *no hyper-parameter tuning* and making it extremely computationally efficient. This method is novel and efficient when used as a whitening method within linear ICA, and when combined with a flow as in Bijecta is a powerful method for non-linear ICA as we demonstrate in experiments.

4.6.1 Approximately-Stiefel matrices

We have set $\mathbf{A}^+ = \mathbf{W}$, the whitening matrix. $\mathbf{W}$ has two aims in non-square ICA. The first is dimensionality reduction, projecting from a d_x-dimensional space to a d_s-dimensional space. The second is to decorrelate the data it transforms, meaning

that the resulting projection will have unit variance and mutually uncorrelated components. More formally we wish for $\mathbf{W}$ of dimensionality $d_s \times d_x$ to be decorrelating.

The set of orthogonal decorrelating rectangular matrices lie on the Stiefel Manifold (Stiefel, 1935) denoted $\mathcal{V}$. For matrices with r rows and c columns, a matrix $\mathbf{G} \in \mathcal{V}(r, c)$ iff $\mathbf{G}\mathbf{G}^* = \mathbf{I}$ ($\mathbf{G}^*$ is the conjugate transpose of $\mathbf{G}$). Constraining the optimisation of $\mathbf{W}$ to this manifold can be computationally expensive and complex (Bakir et al., 2004; Harandi & Fernando, 2016; Siegel, 2021) and instead we choose for $\mathbf{W}$ to be *approximately* Stiefel, that is to lie close to $\mathcal{V}(d_s, d_x)$. This is justified by the following theorem, proved in Appendix C.2:

Theorem 4.1. *Let* $\mathbf{G}$ *be a rectangular matrix and* $\tilde{\mathbf{G}}$ *be its projection onto* $\mathcal{V}(r, c)$*. As the Frobenius norm* $||\mathbf{G} - \tilde{\mathbf{G}}|| \to 0$ *we have that* $||\mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T - \mathbf{\Psi}|| \to 0$*, where* $\mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T$ *is the cross-correlation of the projection of data* $\mathbf{X}$ *by* $\mathbf{G}$*, and* $\mathbf{\Psi}$ *is some diagonal matrix.*

Simply put, this shows that as a matrix $\mathbf{G}$ approaches the Stiefel manifold $\mathcal{V}(r, c)$ the off-diagonal elements of the cross-correlation matrix of the projection $\mathbf{G}\mathbf{X}$ are ever smaller, so $\mathbf{G}$ is ever more decorrelating. Given these properties we want our whitening matrix to lie close to the Stiefel manifold.

4.6.1.1 Johnson-Lindenstrauss projections

By Theorem 4.1 we know that we want our whitening matrix to be close to $\mathcal{V}(d_s, d_x)$. How might we enforce this closeness? By the definition of the Stiefel manifold, we can intuit that a matrix $\mathbf{G}$ will lie close to this manifold if $\mathbf{G}\mathbf{G}^\mathbf{T} \approx \mathbf{I}$. We formalise this as:

Theorem 4.2. *Let* $\mathbf{G} \in \mathbb{R}^{d_s \times d_x}$ *and let* $\tilde{\mathbf{G}}$ *be its projection onto* $\mathcal{V}(d_s, d_x)$*. As the Frobenius norm* $\|\mathbf{G}\mathbf{G}^T - \mathbf{I}\| \to 0$*, we also have* $\|\tilde{\mathbf{G}} - \mathbf{G}\| \to 0$*.*

The proof for this is presented in Appendix C.3. Using this theorem, we now propose an alternative to SVD-based whitening. Instead of having $\mathbf{W}$ be the result of SVD on the data matrix, we define our whitening matrix as a data-independent Johnson–Lindenstrauss transform. We must ensure that $\mathbf{W}$, our rectangular matrix,

is approximately orthogonal, lying close to the manifold $\mathcal{V}(d_s, d_x)$. More formally by Theorem 4.2, our goal is to construct a rectangular matrix $\mathbf{W}$ such that $\mathbf{W}\mathbf{W}^T \approx \mathbf{I}$.

We construct approximately orthogonal matrices for $\mathbf{W}$ by way of a simple JL projection $\mathbf{W}$ for $\mathbb{R}^{d_x} \to \mathbb{R}^{d_s}$ is sampled at initialisation from a simple binary distribution (Achlioptas, 2003):

$$W_{i,j} = \begin{cases} +1/\sqrt{d_s}, & \text{with probability } \frac{1}{2} \\ -1/\sqrt{d_s}, & \text{with probability } \frac{1}{2} \end{cases} \tag{4.11}$$

This distribution satisfies $\mathbb{E}[\mathbf{W}\mathbf{W}^T] = \mathbf{I}$, and such a draw has $\mathbf{W}\mathbf{W}^T \approx \mathbf{I}$. We choose to fix $\mathbf{W}$ after initialisation such that $\mathbf{A}^+ = \mathbf{W}$ never updates—$\mathbf{A}^+$ is *not* learnt—greatly simplifying optimisation. With this now covered, we can define our proposed model:

Definition 4.4. *A Bijecta is a deep generative model composed of a flow with an linear ICA acting on the output of the flow, trained under the objective Eq* (4.10)*, with the unmixing matrix* $\mathbf{A}^+ = \mathbf{W}$ *as defined in Eq* (4.11)*.*

4.7 Linear ICA using Johnson-Lindenstrauss Projections

While we wish to use JL whitening on its own as the unmixing matrix as part of our larger Bijecta model, here we outline how JL whitening can be used effectively within a standard linear-ICA setup.

Thus, instead of having $\mathbf{W} = \mathbf{\Sigma}^{-1}\mathbf{U}^T$, we define our whitening matrix as $\mathbf{W} = \mathbf{\Lambda}\mathbf{Q}$, a learnt diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{d_s}$ and a fixed, data-independent rectangular matrix $\mathbf{Q} \in \mathbb{R}^{d_s \times d_x}$. As above, we must ensure that $\mathbf{Q}$, our rectangular matrix, is approximately orthogonal, lying close to the manifold $\mathcal{V}(d_s, d_x)$, so we use the method of Achlioptas (2003), as described above in Section 4.6.1.1.

This gives us our factorised form for $\mathbf{A}^+$,

$$\mathbf{A}^+ = \mathbf{\Phi}\mathbf{R}\mathbf{\Lambda}\mathbf{Q}. \tag{4.12}$$

As we fix $\mathbf{Q}$ after initialisation, optimisation is solely for the matrix $\mathbf{\Phi R \Lambda}$, greatly simplifying our optimisation problem.

By Theorems 4.1 and 4.2 we know that $\mathbf{A}^+$ using $\mathbf{Q}$ as a component will be close to the manifold of decorrelating matrices, Eq (4.12). For this model to be trained by gradient ascent on the evidence lower bound, we will need suitable constraints on the other parts of $\mathbf{A}^+$, which we will now discuss.

4.7.1 The $SO(d_s)$ Lie group for R

Recall that $\mathbf{R}$ is a square and orthogonal decorrelating matrix of dimensionality $\mathbb{R}^{d_s \times d_s}$. As such, we can constrain $\mathbf{R}$ to be in the orthogonal group $O(d_s)$.

We want to perform unconstrained optimisation in learning our matrix, so we wish to use a differentiable transformation from a class of simpler matrices to $O(d_s)$. One such transform is the Cayley transform (Cayley, 1846), which maps a given anti-symmetric matrix $\mathbf{M}$ (i.e., satisfying $\mathbf{M} = -\mathbf{M}^T$) to the group of special orthogonal matrices $SO(d_s)$ with determinant 1. $SO(d_s)$ is the elements of the group $O(d_s)$ with determinant 1. Unlike $O(d_s)$ it is path-connected, aiding optimisation. As such, we propose defining our square unmixing matrix using the Cayley transform of the anti-symmetric matrix $\mathbf{M}$,

$$\mathbf{R} = (\mathbf{I} - \mathbf{M})^{-1}(\mathbf{I} + \mathbf{M}). \tag{4.13}$$

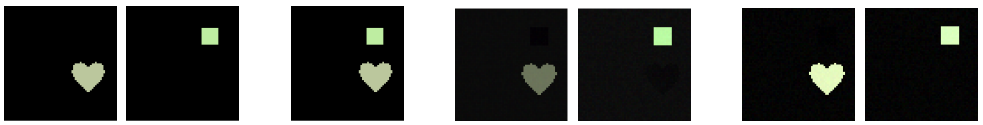
This can be formulated as an unconstrained problem, easing optimisation, by defining $\mathbf{M} = (\mathbf{L} - \mathbf{L}^T)/2$ and then optimising over the square real-valued matrix $\mathbf{L}$. This further reduces the optimisation space for $\mathbf{A}^+$ to $\mathbb{R}^{\frac{1}{2}d_s(d_s+3)}$.

These are all the constraints needed, so training of this linear model can proceed as described in Section 4.5.1.

4.8 Experiments

Here we show that our approach outperforms VAEs and flows with ICA-priors at discovering ICA sources in image data.

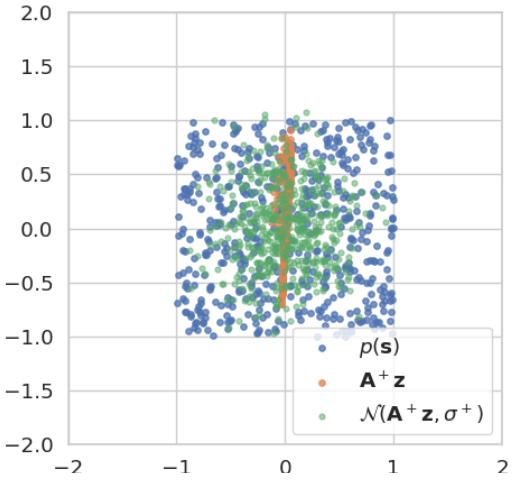
Linear ICA using JL Whitening But first, in Fig 4.4 as a sanity check, we show that a linear ICA model using JL projections to whiten can successfully unmix linearly mixed sources. We take a pair of images from dSprites (Matthey et al., 2017) and create linear mixtures of them. We see that linear ICA with JL projections (§ 4.7) can successfully discover the true sources, the images used to create the mixtures, in the columns of $\mathbf{A}$.



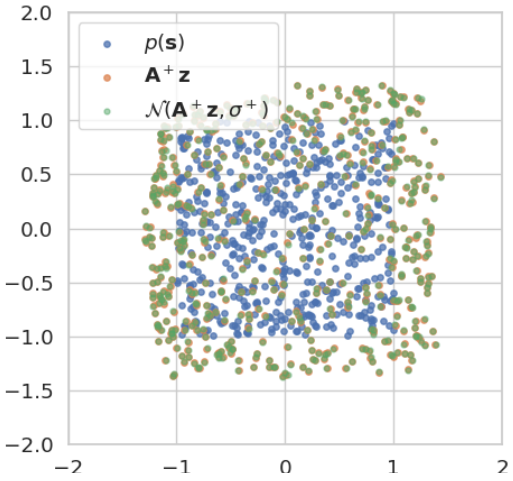
(a) Source Images **(b)** Mixed **(c)** $\mathbf{A}$ FastICA **(d)** $\mathbf{A}$ JL-Cayley ICA

Figure 4.4: Here we run linear ICA on a pair of images (a) that are mixed linearly ($\text{mix} = w_1 * \text{image}_1 + w_2 * \text{image}_2$) (b) to form a dataset with 512 points. In both cases w_1 and w_2 are sampled from a uniform distribution. We plot the mixing matrix $\mathbf{A}$ for our JL-Cayley model with a quasi-uniform GG prior with $\rho = 10$ (c) and for FastICA (Hyvärinen & Oja, 1997) as a benchmark. $\mathbf{A}$ should recover the source images, which occurs for both models.

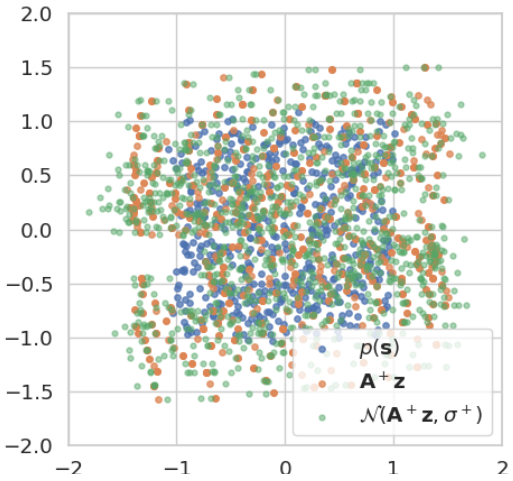
Affine Data Given that we have established that our novel theory for decorrelating matrices can produce standalone linear ICA models, we now want to ascertain that our hybrid model performs well in non-linear mixing situations. To do so we create a dataset consisting of a subset of dSprites where we have a light-blue heart randomly uniformly placed on a black field. The true latent sources behind these randomly sampled affine transformations are simply the coordinates of the heart. First, in Fig 4.5 we demonstrate that linear ICA models are unable to uncover the true latent sources. As expected non-linear mixing regimes motivate the use of flexible non-linear models.



(a) Linear ICA

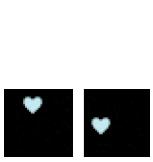


(b) Bijecta

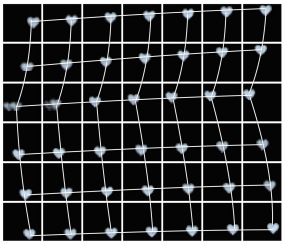


(c) VAE

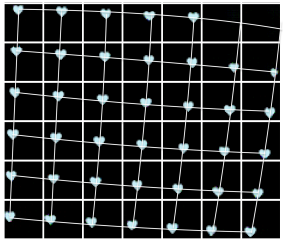
Figure 4.5: In (a), (b) we run linear ICA and a single-layer Bijecta on the affine transformation dataset of Fig 4.6. We take a dSprites heart and using a randomly sampled affine transformation, move it around a 32 by 32 background. We plot the posterior distribution $\mathcal{N}(\mathbf{A}^+\mathbf{z}, \boldsymbol{\sigma}^+)$ (green) and its mean $\mathbf{A}^+\mathbf{z}$ (orange) for both models. Clearly the posterior from Bijecta is better able to match the quasi-uniform GG prior with $\rho = 10$ ($p(\mathbf{s})$ in blue) than the linear ICA model, highlighting that the addition of the flow allows for linear unmixing.



(a) Sample Images



(b) VAE Latent Traversals



(c) Bijecta Latent Traversals

Figure 4.6: Here we demonstrate that Bijecta is capable of unmixing non-linearly mixed sources, better than VAEs with ICA-appropriate priors. We take a dSprites heart and, using a randomly sampled affine transformation, move it around a 32 by 32 background (a). With 2-D GG priors with $\rho = 10$ for a convolutional VAE (b) and for Bijecta (c) we plot the generations resulting from traversing the 2-D latent-source space in a square around the origin. We sketch the learnt axis of movement of the sprite with white lines. In (b) the VAE does not ascribe consistent meaning to its latent dimensions. It has failed to discover consistent independent latent sources: it has a sudden change in the learnt axes of movement along the second dimension, as seen by the kink in the white vertical lines. In (c) Bijecta is able to learn a simple affine transformation along each latent dimension, consistently spanning the space. In Fig 4.5 we show the posterior distributions of both these models and show that Bijecta is better able to match the GG prior than the VAE, supporting our findings here.

We now demonstrate that Bijecta can uncover the latent sources underpinning these affine transformations, whereas VAEs with ICA-appropriate priors fail to do so. For details of VAE architecture, see Appendix C.5. These VAEs are able to learn to reconstruct data well, but the learnt latent space does not correspond to the underlying statistically independent sources (see Figs 4.1 and 4.5). In fact for VAEs the effect of the latent variables is not consistent throughout the latent space, as seen in Fig 4.6. For Bijecta, the learnt latent space corresponds to the underlying statistically independent sources (see Figs 4.1 and 4.5), and the meaning of the latent variables is consistent in Fig 4.6. Further in Fig 4.6 the model seems able to extrapolate outside the training domain: it generates images where the heart is partially rendered at the edges of the frame, even removing the heart entirely at times, even though such images are not in the training set.

Natural Images The previous experiments show that our model is capable of isolating independent sources on toy data. We complement this finding with

experiments on a more complex natural image dataset, CelebA (Liu et al., 2015), and show that here too our model outperforms VAEs in learning factorisable representations.

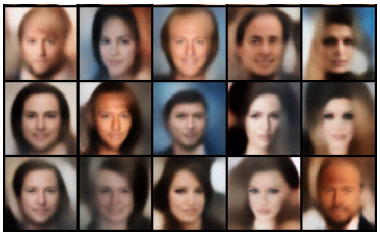
An ersatz test of this can be done by synthesising images where we sample from a factorised approximation of Bijecta's posterior. If the learned latent sources are actually independent, then the posterior over latent sources given the entire dataset should factorise into a product across dimensions, i.e. $q(\mathbf{s}) = \prod_i q(\mathbf{s}_i)$. In this case, we can fit an approximation to the posterior by fitting d_s independent one-dimensional density estimates on $q(\mathbf{s}_i)$. If the sources are not independent, then this factorised approximation to the posterior will be missing important correlations and dependencies. In Fig 4.7a samples from this factorised approximation look reasonable, suggesting that Bijecta has learnt representations that are statistically independent.

To quantify this source-separation numerically, we measure the total correlation (TC) of the aggregate posteriors of Bijecta ($q(\mathbf{s}|\mathbf{z})$) and VAEs ($q(\mathbf{z}|\mathbf{x})$) as Chen et al. (2018a) do. It directly measures how well an ICA model has learnt decorrelated and independent latent representations (Everson & Roberts, 2001).

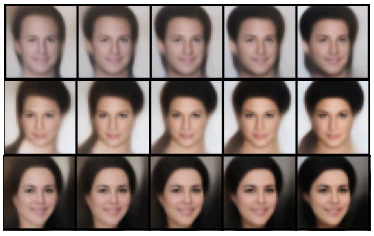
In Table 4.1, we show that Bijecta learns an aggregate posterior with significantly lower TC values than both VAEs with Laplace priors, *and* β-TCVAEs – which in their training objective penalise the TC by a factor β (Chen et al., 2018a). Our model has learnt a better ICA solution. We also include numerical results in Appendix C.6.1 showing that Bijecta outperforms linear ICA on a variety of natural image datasets.

Table 4.1: Total Correlation Results: We evaluate the source separation of different models on CelebA via the TC of the validation set embeddings in the 32-D latent space of: Laplace prior VAEs, β-TCVAEs ($\beta = 15$), and Bijecta with a Laplace prior ($\pm$ indicates the standard deviation over 2 runs). VAEs use the same architecture and training as Chen et al. (2018a).

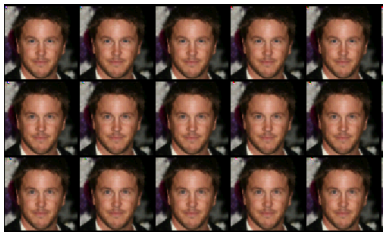
	Laplace-VAE	β-TCVAE	Laplace-Bijecta
TC:	106.7 ± 0.9	55.7 ± 0.1	$\mathbf{13.1 \pm 0.4}$



(a) Bijecta Factorised Posterior



(b) Bijecta latent 5 traversal



(c) RQS flow traversal

Figure 4.7: (a) shows decodings from an 8-layer Bijecta ($d_s = 32$) trained on CelebA with a Laplace prior (GG $\rho = 1$) where we sample from the factorised approximation to Bijecta's posterior. (b) shows latent traversals for 3 different datapoints all along the same axis-aligned direction, for this same model. (c) shows traversals for a single embedded training datapoint from CelebA moving along 3 latent directions in an RQS flow with Laplace base distribution. Though we have selected 3 dimensions, all $\mathcal{Z}$ dimensions had similar latent traversals. In (b-c) Images in the centre correspond to the original latent space embedding, on either side we move up to 6 standard deviations away along this direction with other dimensions remaining fixed. The flow has not discovered axis-aligned transforms, whereas Bijecta has learned informative latent dimensions: here the dimension encodes hair thickness. Note that identity is maintained throughout and that the transform is consistent across different posterior samples.

Dimensionality reduction on flow models To conclude, having shown that Bijecta outperforms VAEs on a variety of non-linear ICA tasks, we now contrast our model's ability to automatically uncover sources relative to flow models with heavy-tailed base distributions. We do so by measuring the cumulative explained variance by the dimensions in $\mathcal{Z}$ for both models. If a small number of dimensions in $\mathcal{Z}$ explain most of the variance of the flow-representations of the dataset, then the model has learnt a bijection which only requires a small number of dimensions to be invertible. It has in effect learnt the generating sources underpinning the data.

In Fig 4.8 we show that Bijecta induces better-compressed representations in $\mathcal{Z}$ than vanilla non-compressive flows. We plot the eigenvalues of the covariance matrix on the output of the flow, i.e. on $\mathrm{Cov}(f(\mathbf{X}))$, to see how much of the total variance

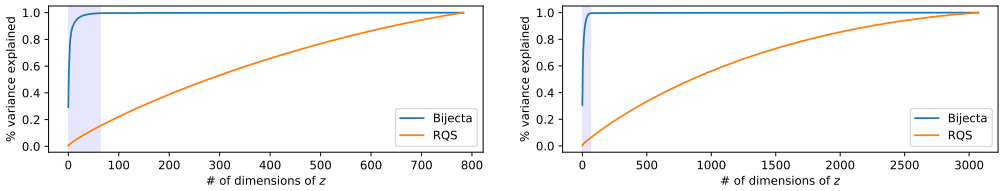


Figure 4.8: Explained variance plots for the embedding in $\mathcal{Z}$, as measured by the sums of the eigenvalues of the covariance matrix of the embeddings, for both our Bijecta model and for an RQS model of equivalent size trained with a Laplace base distribution (GG distribution with $\rho = 1$). For both Fashion-MNIST (left) and CIFAR 10 (right) datasets we see that the Bijecta model has learned a compressive flow, where most of the variance can be explained by only a few linear projections. The shaded region denotes the first 64 dimensions, corresponding to the size of the target source embedding $\mathcal{S}$.

in the learned feature space $\mathcal{Z}$ can be explained in a few dimensions. In doing so we see that a flow trained jointly with a linear ICA model with $d_s = 64$ effectively concentrates variation into a small number of intrinsic dimensions; this is in stark contrast with the RQS flows trained with only a Laplace base distribution. This demonstrates that our model is able to automatically detect relevant directions on a low dimensional manifold in $\mathcal{Z}$, and that the bijective component of our model is better able to isolate latent sources than a standard flow.

For a visual illustration of this source separation we show the difference in generated images resulting from smoothly varying along each dimension in $\mathcal{S}$ for Bijecta models and in $\mathcal{Z}$ for flows in Fig 4.7. Bijecta is clearly able to discover latent sources, whereby it learns axis-aligned transformations of CelebA faces, whereas a flow with equivalent computational budget and a heavy-tailed base distribution is not able to.

All flow-based baselines are trained using the objective in Eq (4.4), using Real-NVP style factoring-out (Durkan et al., 2019; Dinh et al., 2015), and are matched in size and neural network architectures to the flows of Bijecta models. See Appendix C.5 for more details.

4.9 Conclusion

We have developed a method for performing non-linear ICA large high-dimensional image datasets which combines state-of-the-art flow-based models and a novel

theoretically grounded linear ICA method. This model succeeds where existing probabilistic deep generative models fail: its constituent flow is able to learn a representation, lying in a low dimensional manifold in $\mathcal{Z}$, under which sources are separable by linear unmixing. In source space $\mathcal{S}$, this model learns a low dimensional, explanatory set of statistically independent latent sources.

Fool me once, shame on you; fool me twice, shame on me

— Anon

5

Improving the Robustness of VAEs to Adversarial Attack

Contents

5.1 Motivation

Variational autoencoders have been shown to be vulnerable to adversarial attacks, wherein they are fooled into reconstructing a chosen target image. However, how to defend against such attacks remains an open problem. We make significant advances

in addressing this issue by introducing methods for producing adversarially robust VAEs. Namely, we first demonstrate that methods proposed to obtain disentangled latent representations produce VAEs that are more robust to these attacks. However, this robustness comes at the cost of reducing the quality of the reconstructions. We ameliorate this by applying disentangling methods to hierarchical VAEs. The resulting models produce high–fidelity autoencoders that are also adversarially robust. We confirm their capabilities on several different datasets and with current state–of–the–art VAE adversarial attacks, and also show that they increase the robustness of downstream tasks to attack.

5.2 Introduction

As we have seen in previous chapters, variational autoencoders are a powerful approach to learning deep generative models. However, previous work has shown that they are vulnerable to adversarial attacks (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018; Kos et al., 2018): an adversary attempts to fool the VAE to produce reconstructions similar to a chosen target by adding distortions to the original input, as shown in Fig 5.1. This kind of attack can be harmful when the encoder's output is used downstream, as in Xu et al. (2017); Kusner et al. (2017); Theis et al. (2017); Townsend et al. (2019); Ha & Schmidhuber (2018); Higgins et al. (2017b). As VAEs are often themselves used to protect classifiers from adversarial attack (Schott et al., 2019; Ghosh et al., 2019), ensuring VAEs are robust to adversarial attack is an important endeavour.

Despite these vulnerabilities, little progress has been made in the literature on how to ***defend*** VAEs from such attacks. The aim of this paper is to investigate and introduce possible strategies for defence. We seek to defend VAEs in a manner that maintains reconstruction performance. Further, we are also interested in whether methods for defence increase the robustness of downstream tasks using VAEs.

Our first contribution is to show that regularising the variational objective during training can lead to more robust VAEs. Specifically, we leverage ideas from the

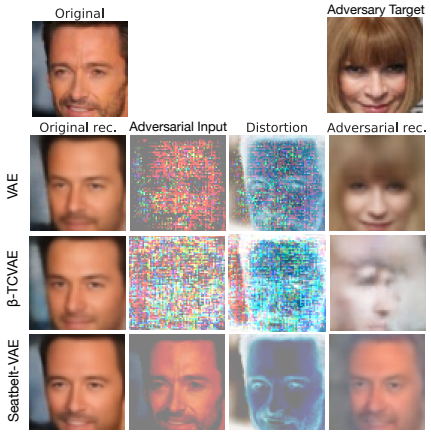


Figure 5.1: Adversarial attacks on CelebA for different models. Here we start with the image of Hugh Jackman and introduce an adversary that tries to produce reconstructions that look like Anna Wintour. This is done by applying a distortion (third column) to the original image to produce an adversarial input (second column). We can see that the adversarial reconstruction for the Vanilla VAE looks substantially like Wintour, indicating a successful attack. Adding a regularisation term using the β-TCVAE produces an adversarial reconstruction that does not look like Wintour, but it is also far from a successful reconstruction. The hierarchical version of a β-TCVAE (which we call Seatbelt-VAE) is sufficiently hard to attack that the output under attack still looks like Jackman, not Wintour.

disentanglement literature (Mathieu et al., 2019) to improve VAEs' robustness by learning smoother, more stochastic representations that are less vulnerable to attack. In particular, we show that the total correlation (TC) term used to encourage independence between latents of the learned representations (Kim & Mnih (2018); Chen et al. (2018a); Esmaeili et al. (2019), and in the previous chapter in Definitions 4.2 & 4.3), also serves as an effective regulariser for learning robust VAEs.

Though a clear improvement over the standard VAE, a severe drawback of this approach is that the gains in robustness are coupled with drops in the reconstruction performance, due to the increased regularisation. Furthermore, we find that the achievable robustness with this approach can be limited (see Fig 5.1) and thus potentially insufficient for particularly sensitive tasks. To address this, we apply TC–regularisation to *hierarchical* VAEs. By using a richer latent space representation than a standard VAE, the resulting models are not only more robust still to adversarial attacks than single-layer models with TC regularisation, but can also provide reconstructions which are comparable to, and often even better than, the standard (unregularised, single-layer) VAE.

In this chapter we provide insights into what makes VAEs vulnerable to attack and how we might go about defending them. To that end we draw novel connections

between adversarial robustness and methods proposed in the context of disentanglement. We demonstrate that regularised VAEs, trained with an up-weighted total correlation, are significantly more robust to attacks than vanilla VAEs. Building on this we develop regularised hierarchical VAEs that provide both further robustness to adversarial attack and improved reconstructions. Finally, we show that robustness to adversarial attack also confers increased robustness to downstream tasks.

5.3 Background

5.3.1 Attacking VAEs

In adversarial attacks an agent is trying to manipulate the behaviour of some model towards a goal of their choosing, such as fooling a classifier to misclassify an image through adding a small perturbation (Akhtar & Mian, 2018; Gilmer et al., 2018). For many deep learning models, very small changes in the input, imperceptible or unimportant to the human eye, can produce large changes in output.

Attacks on VAEs have been proposed where the adversary looks to apply small input distortions that produce reconstructions close to a target adversarial image (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018; Kos et al., 2018). An example of this is shown in Fig 5.1, where a standard VAE is successfully attacked to turn Hugh Jackman into Anna Wintour.

Unlike more established adversarial settings, only a small number of such VAE attacks have been suggested in the literature. The current known most effective mode of attack is a *latent space attack* (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018; Kos et al., 2018). This aims to find a distorted image $\mathbf{x}^* = \mathbf{x} + \mathbf{d}$ such that its posterior $q_\phi(\mathbf{z}|\mathbf{x}^*)$ is close to that of the agent's chosen target image $q_\phi(\mathbf{z}|\mathbf{x}^t)$ under some metric. This then implies that the likelihood $p_\theta(\mathbf{x}^t|\mathbf{z})$ is high when given draws from the posterior of the adversarial example. It is particularly important to be robust to this attack if one is concerned with using the encoder network of

a VAE as part of a downstream task. For a VAE with a single stochastic layer, the latent-space adversarial objective is

$$\Delta_{\mathrm{r}}(\mathbf{x}, \mathbf{d}, \mathbf{x}^t; \lambda) = r(q_\phi(\mathbf{z}|\mathbf{x}+\mathbf{d}), q_\phi(\mathbf{z}|\mathbf{x}^t)) + \lambda||\mathbf{d}||_2, \tag{5.1}$$

where $r(\cdot,\cdot)$ is some divergence or distance, commonly a D_{KL}(Tabacof et al., 2016; Gondim-Ribeiro et al., 2018). We are penalising the L_2 norm of $\mathbf{d}$ too, so as to aim for attacks that change the image less. We can then simply optimise to find a good distortion $\mathbf{d}$.

Alternatively, we can aim to directly increase the ELBO for the target datapoint under the attacked input (Kos et al., 2018):

$$\Delta_{\mathrm{output}}(\mathbf{x}, \mathbf{d}, \mathbf{x}^t; \lambda) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}+\mathbf{d})}\left[\log(\mathbf{x}^t|\mathbf{z})\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}+\mathbf{d})||p(\mathbf{z})) + \lambda||\mathbf{d}||_2. \tag{5.2}$$

5.4 Defending VAEs

Given these approaches to attacking VAEs, the critical question is now how to defend them. This problem was not considered by prior works. To address it, we first need to consider what makes VAEs vulnerable to adversarial attacks. We argue that two key factors dictate whether we can perform a successful attack on a VAE: a) whether we can induce significant changes in the encoding distribution $q_\phi(\mathbf{z}|\mathbf{x})$ through only small changes in the data $\mathbf{x}$, and b) whether we can induce significant changes in the reconstructed images through only small changes to the latents $\mathbf{z}$. The first of these relates to the *smoothness* of the encoder mapping, the latter to the smoothness of the decoder mapping.

Consider, for the sake of argument, the case where the encoder–decoder process is almost completely noiseless. Here successful reconstruction places no direct pressure for similar encodings to correspond to similar images: given sufficiently powerful networks, very small changes to embeddings $\mathbf{z}$ can imply very large changes to the reconstructed image; there is no ambiguity in the "correct" encoding of a particular datapoint. In essence, we can have lookup–table style behaviour – nearby

realisations of $\mathbf{z}$ do not necessarily relate to each other and very different images can have very similar encodings.

This will now be very vulnerable to adversarial attacks: small input changes can lead to large changes in the encoding, and small encoding changes can lead to large changes in the reconstruction. It will also tend to overfit and have gaps in the aggregate posterior, $q_\phi(\mathbf{z}) = \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)$, as each $q_\phi(\mathbf{z}|\mathbf{x}_n)$ will be sharply peaked. These gaps can then be exploited by an adversary.

There are two mechanisms by which we can reduce this lookup-table behaviour, thereby reducing gaps in the aggregate posterior. First, we can try to regulate the level of noise in the per-datapoint posterior covariance, to then obtain smoothness in the overall embeddings. Having a stochastic encoding creates uncertainty in the latent that gives rise to a particular image, forcing similar latents to correspond to similar images. Adding noise forces the VAE to smooth the encode-decode process in that similar images will lead to similar embeddings in the latent space, ensuring that small changes in the input result in small changes in the latent space and result in small changes in the decoded outputs. This proportional input-output change is what we refer to as a 'simple' encode-decode process, which is the second mechanism that can reduce look-up table behaviour.

The fact that the VAE is vulnerable to adversarial attack suggests that its standard setup does not obtain sufficiently smooth and simple representations to provide an adequate defence. Introducing additional regularisation to enforce simplicity or increased posterior covariance thus provides a prospect for defending VAEs. We could attempt to obtain this by direct regularisation of the networks (e.g. weight decay). Here, however, we focus on macro-level regularisation approaches as discussed in the next section. The reason for this is that controlling the macroscopic behaviour of the networks through low-level regularisations can be difficult to control and, in particular, difficult to calibrate. Further, as the most effective attack on VAEs currently attack the latent space, it is reasonable that regularisation methods that directly act on the properties of the latent space form a good place to start.

5.4.1 Disentangling Methods and Robustness

Recent research into disentangling VAEs (Higgins et al., 2017a; Siddharth et al., 2017; Kim & Mnih, 2018; Chen et al., 2018a; Esmaeili et al., 2019; Mathieu et al., 2019) and the information bottleneck (Alemi et al., 2017, 2018) has looked to regularise the ELBO with the hope of providing more interpretable embeddings. These regularisers also have influences on the smoothness and stochasticity of the embeddings learned. We have already introduced some of these methods in the context of non-linear ICA in § 4.3.4.

Of particular relevance to us here is introduce the notion, introduced in Mathieu et al. (2019), of ***overlap*** in the embedding of a VAE: that is, the level of overlap between per-datapoint posteriors as they combine to form the aggregate posterior. They show how controlling this is critical to achieving a smoothly varying latent embedding. Overlap encapsulates both the level of uncertainty in the encoding process and also a locality of this uncertainty.

To learn a smooth representation we not only need our encoder distribution to have an appropriate entropy, we also want the different possible encodings to be similar to each other. Critically, Mathieu et al. (2019) show that many methods proposed for disentangling, and in particular the β-VAE (Higgins et al. (2017a), & Definition 4.1), provide a mechanism for directly controlling this overlap.

Recall that it is gaps, holes, in the aggregate posterior that adversaries can exploit. We want to close up these holes without degrading the model too much. In regions of $\mathcal{Z}$ when the aggregate posterior places no density the decoder is unconstrained by the ELBO (Rezende & Viola, 2018). It is these regions, with associated unconstrained decoder behaviour, that enable adversaries to have an easy time attacking the model. Thus our aim—making robust VAEs that are smooth in the sense of having relatively flat aggregate posterior density across $\mathcal{Z}$ such that they have no holes—is equivalent to controlling Mathieu et al. (2019)'s overlap.

Thus we see that controlling overlap may also provide a mechanism for improving VAEs' robustness. This observation now hints at an interesting question: *can we use methods initially proposed to encourage disentanglement to encourage robustness*?

It is important to underline here again, repeating § 4.3.4, that disentangling can be difficult to achieve in practice, typically requiring precise choices in the hyperparameters of the model and the weighting of the added regularisation term, and often also a fair degree of luck (Locatello et al., 2019; Mathieu et al., 2019; Rolinek et al., 2019). As such, we are not suggesting that inducing *disentangled representations* is a practical mechanism for inducing robustness, or indeed that disentangled representations should be any more robust. Rather, as highlighted above, we are interested in whether regularisers, traditionally used to encourage disentanglement, reliably lead to adversarially robust VAEs.

Indeed, we will find that though our approaches—based on these regularisers—provide reliable and significant improvements in robustness, these improvements are not generally due to any noticeable improvements in disentanglement itself (see Appendix D.4.1).

Regularising for Robustness There are a number of different disentanglement methods that one might consider using to train robust VAEs. Perhaps the simplest would be to use a β-VAE (Higgins et al. (2017a), & Definition 4.1), wherein we up-weight the D_{KL} term in the VAE's ELBO by a factor $\beta \geq 1$. However, the β-VAE only increases overlap at the expense of substantial reductions in reconstruction quality as the data likelihood term has, in effect, been down-weighted (Kim & Mnih, 2018; Chen et al., 2018a; Mathieu et al., 2019).

Because of these shortfalls in β-VAEs, we instead propose to regularise through penalisation of a total correlation (TC) term (Kim & Mnih (2018); Chen et al. (2018a), Definitions 4.2 & 4.3). As discussed in the previous chapter, this looks to directly force independence across the different latent dimensions in aggregate posterior $q_\phi(\mathbf{z})$, such that the aggregate posterior factorises across dimensions. This

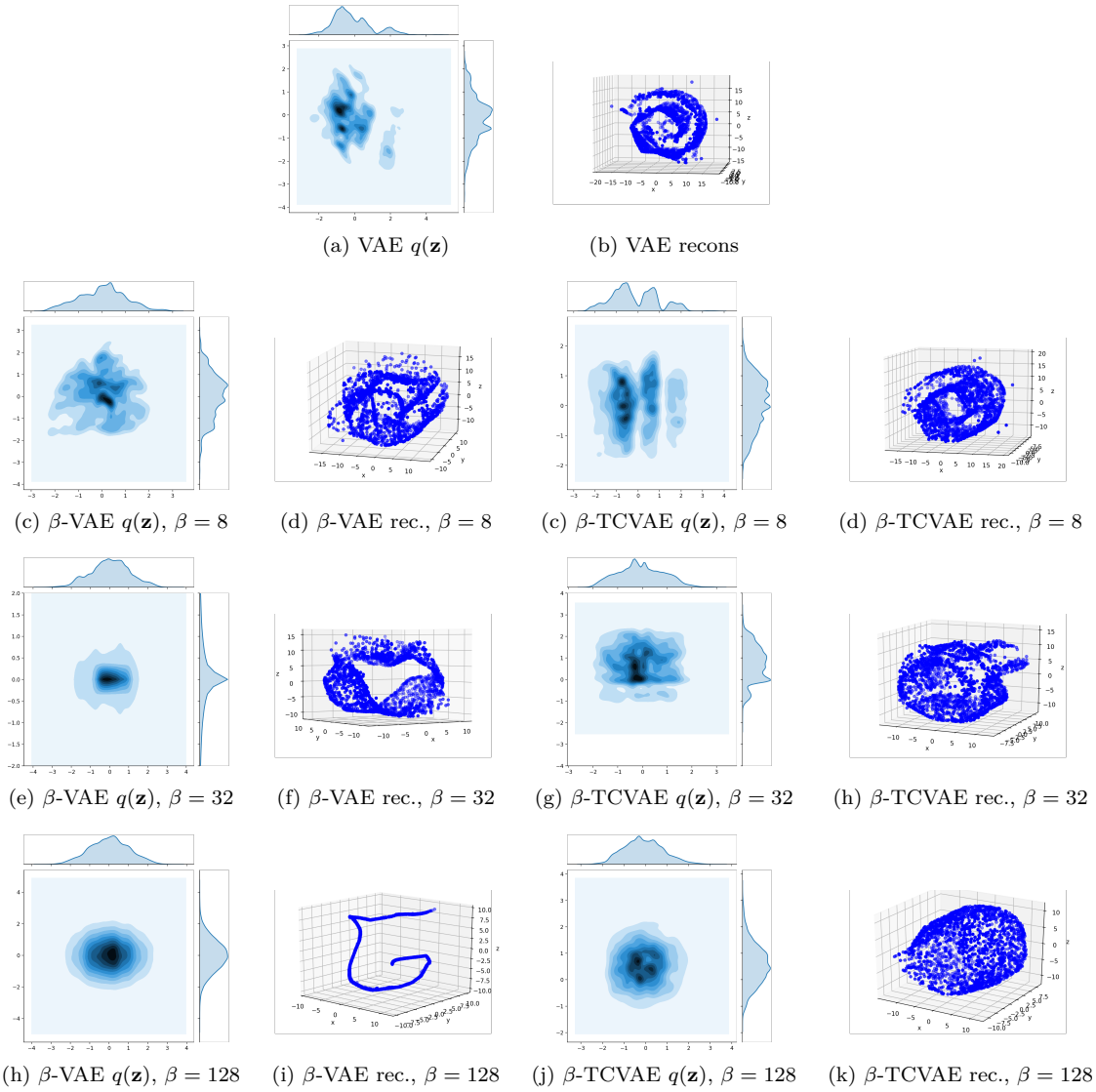
approach has been shown to have a smaller deleterious effect on reconstruction quality than found in β-VAEs (Chen et al., 2018a).

Let us pause to consider why each of these regularisation methods can be expected to increase overlap, and why TC-penalisation might do so in a superior way to β-VAE-upweighting of $D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$.

In a β-VAE with large β we are asking that the amortised posterior is close to the prior for all inputs. So for $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbb{I})$ we are forcing $\boldsymbol{\mu}_\phi(\mathbf{x})$ to $\mathbf{0}$ and $\boldsymbol{\sigma}_\phi(\mathbf{x})$ to $\mathbf{1}$. Naturally this will lead our aggregate posterior to have a high degree of overlap between its constituent mixture components, because all of them are being driven to be the same. (We will return to this phenomena in more theoretical depth next chapter, but this is all we need for now.) With all per-datapoint posteriors being driven to be the same, information about the initial input data is necessarily lost in these representations.

For a β-TCVAE, however, the demand for the aggregate posterior to be well-approximated by the product of its marginals does not entail a fixed scale, nor does it push all the per-datapoint posteriors towards the prior. Rather we are directly asking for statistical independence between coordinate directions. Holes in the aggregate posterior are (as long as they are off-axis) a form of dependency between the latent variables. By demanding that the aggregate posterior factorises, we are thus asking the model to *smooth out* any holes (or peaks) that do not lie along the axes of the latent space. We might reasonably hope that this can be achieved without causing as serious degradation to model quality, as measured by the fidelity of reconstructions and the values of the ($\beta = 1$) ELBO.

Here we demonstrate the effect of these regularisers, both quantitatively and qualitatively, to bolster our intuitions about how they work and thus why TC-penalisation is a promising approach for our purposes. First we perform some experiments on toy 3D 'Swiss Roll' data, Fig 5.2. We train 2D-latent-space VAEs: vanilla, β-VAEs, and β-TCVAEs. We plot the aggregate posterior and the reconstructions (the means of the data likelihood conditioned on a sample



(a) VAE $q(\mathbf{z})$ (b) VAE recons

(c) β-VAE $q(\mathbf{z})$, $\beta = 8$ (d) β-VAE rec., $\beta = 8$ (c) β-TCVAE $q(\mathbf{z})$, $\beta = 8$ (d) β-TCVAE rec., $\beta = 8$

(e) β-VAE $q(\mathbf{z})$, $\beta = 32$ (f) β-VAE rec., $\beta = 32$ (g) β-TCVAE $q(\mathbf{z})$, $\beta = 32$ (h) β-TCVAE rec., $\beta = 32$

(h) β-VAE $q(\mathbf{z})$, $\beta = 128$ (i) β-VAE rec., $\beta = 128$ (j) β-TCVAE $q(\mathbf{z})$, $\beta = 128$ (k) β-TCVAE rec., $\beta = 128$

Figure 5.2: β-VAEs and β-TCVAEs trained on swiss roll data, with a vanilla VAE as baseline. $\beta \in \{8, 32, 128\}$.

of each per-datapoint posterior). Clearly the amount of overlap increases with β for both kinds of model, but the β-TCVAEs does this in a more structured way and, unlike the β-VAE, does not suffer from (eventually catastrophic) degradation in model quality for large β.

Secondly, in Fig 5.3 we train β-VAEs, and β-TCVAEs on CelebA. We plot the density of $||\boldsymbol{\sigma}_\phi(\mathbf{x})||_2$ for β=10, which shows that while $\boldsymbol{\sigma}_\phi(\mathbf{x})$ concentrates at a particular value for β-VAEs, but for β-TCVAEs it takes a broader range of values – values above the saturation point of β-VAEs. We also plot both the reconstruction

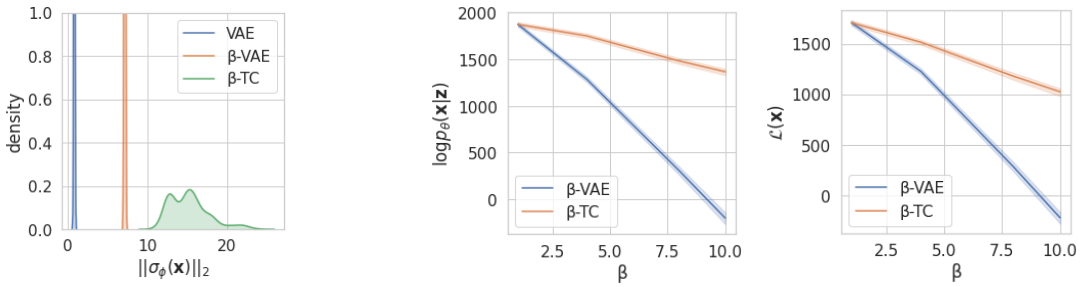


Figure 5.3: [Left] density plot of $||\boldsymbol{\sigma}_\phi(\mathbf{x})||_2$ (the norm of the encoder standard deviation) for a VAE, a β-VAE and a β-TCVAE each trained on CelebA, $\beta = 10$. The β-VAE's posterior variance saturates, while the β-TCVAE's does not and as such is able to induce more overlap. [Right] the likelihood ($\log p_\theta(\mathbf{x}|\mathbf{z})$) and ELBO for both as a function of β. Clearly the model quality degrades to a lesser degree for the TC-penalised models under increasing β.

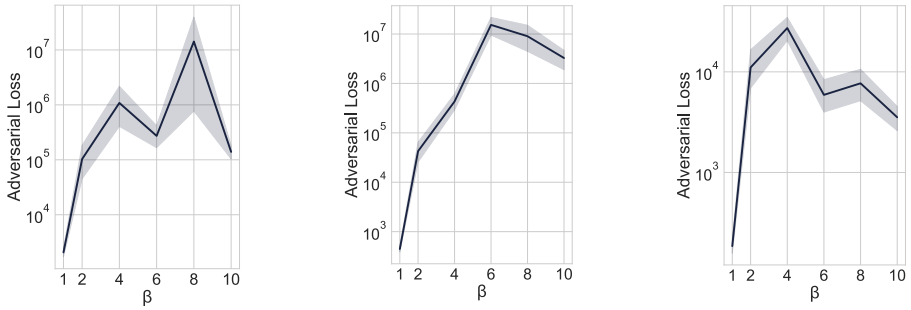
quality, as measured by $\log p_\theta(\mathbf{x}|\mathbf{z})$, and ELBO (measured with β=1) as a function of β. From these we can see that while TC-penalisation gives increased posterior variance (and so increased overlap), it causes less damage to model performance.

To summarise: the greater overlap, the lesser degradation of reconstruction quality and the lesser degradation of model quality induced by β-TCVAE make them highly suitable for our purposes.

5.4.2 Adversarial Attacks on TC-Penalised VAEs

We now consider attacking these TC-penalised VAEs and demonstrate one of the key contributions of the paper: that empirically this form of regularisation makes adversarial attacks on VAEs harder to carry out. To do this, we first train them under the β-TCVAE objective (i.e. Eq (4.6)), jointly optimising θ, ϕ for a given β. Once trained, we then attack the models using the latent-space attack method outlined in § 5.3.1, finding an input distortion $\mathbf{d}$ that minimises the latent attack loss Δ as per Eq (5.1) with $r(\cdot,\cdot) = D_{\mathrm{KL}}(\cdot||\cdot)$.

One possible metric for how successful such attacks have been is the achieved value reached of the attack loss Δ_{KL}. If the latent space distributions for the original input and for the distorted input match closely for a small distortion, then Δ_{KL} is small and the model has been successfully fooled – reconstructions from



(a) dSprites **(b)** Chairs **(c)** 3D Faces

Figure 5.4: Attacker's achieved loss Δ_{KL} (i.e. Eq (5.1) with $r = D_{\mathrm{KL}}$) for β-TCVAE for different β values and datasets. Higher loss indicates more robustness. Shading corresponds to the 95% CI produced by attacking 20 images for each combination of $d_{\mathbf{z}} = \{4, 8, 16, 32, 64, 128\}$ and taking 50 geometrically distributed values of λ between 2^{-20} and 2^{20} (giving 1000 total trials). Note that the loss axis is logarithmic. $\beta > 1$ clearly induces a much larger loss for the adversary relative to $\beta = 1$ for all datasets.

samples from the attacked posterior would be indistinguishable from those from the target posterior. Meanwhile, the larger the converged value of the attack loss the less similar these distributions are and the more different the reconstructed image is to the adversarial target image.

We carry our these attacks for dSprites (Matthey et al., 2017), Chairs (Aubry et al., 2014) and 3D faces (Paysan et al., 2009), for a range of β and λ values. We pick values of λ following standard methodology (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018), and use L-BFGS-B for gradient descent (Byrd et al., 1995). We also varied the dimensionality of the latent space of the model, $d_{\mathbf{z}}$, but found it had little effect on the effectiveness of the attack.

In Fig 5.4 we show the effect on the attack loss Δ_{KL} for varying β, averaged over different original input-target pairs and values of $d_{\mathbf{z}}$. Note that the plot is logarithmic in the loss. We see a clear pattern for each dataset that the loss values reached by the adversary increases as we increase β from the standard VAE (i.e. $\beta = 1$). This analysis is also borne out by visual inspection of the effectiveness of these attacks, for example as shown in Fig 5.1. We will return to give further experimental results in Section 5.6. An interesting aspect of Fig 5.4 is that in many

cases the adversarial loss starts to decrease if β is too large: as β increases there is less pressure in the objective to produce good reconstructions.

5.5 Hierarchical TC–Penalised VAEs

We are now armed with the fact that penalising the TC in the ELBO induces robustness in VAEs. However, TC-penalisation in single layer VAEs comes at the expense of model reconstruction quality (Chen et al., 2018a), albeit less than that in β-VAEs. Our aim is to develop a model that is robust to adversarial attack while mitigating this trade-off between robustness and sample quality. To achieve this, we now consider instead using hierarchical VAEs (Rezende et al., 2014; Sønderby et al., 2016; Kingma et al., 2016; Zhao et al., 2017; Maaløe et al., 2019; Vahdat & Kautz, 2020; Child, 2021). These are known for their superior modelling capabilities and more accurate reconstructions. As these gains stem from using more complex hierarchical latent spaces, rather than less noisy encoders, this suggests they may be able to produce better reconstructions and generative capabilities, while also remaining robust to adversarial attacks when appropriately regularised.

As mentioned in § 3.3.2, the simplest hierarchical extension of conditional stochastic variables in the generative model is the Deep Latent Gaussian Model (DLGM) of Rezende et al. (2014). Here the forward model factorises as a chain, $p_\theta(\mathbf{x}, \vec{\mathbf{z}}) = p_\theta(\mathbf{x}|\mathbf{z}^1) \prod_{i=1}^{L-1} p_\theta(\mathbf{z}^i|\mathbf{z}^{i+1}) p(\mathbf{z}^L)$, where each $p_\theta(\mathbf{z}^i|\mathbf{z}^{i+1})$ is a Gaussian distribution with mean and variance parameterised by deep nets, while $p(\mathbf{z}^L)$ is an isotropic Gaussian. Unfortunately, we found that naively applying TC-correlation penalisation to DLGM-style VAEs did not confer the improved robustness we observed in single layer VAEs. We postulate that this observed weakness is inherent to the structure of chain factorisation in the generative model. Recall from § 3.3.2 that chain factorisation has well known drawbacks, from the data-likelihood depending solely on $\mathbf{z}^1$, the bottom-most latent variable. Here attackers only need to manipulate $\mathbf{z}^1$ to produce a successful attack.

To account for this, we instead use a generative model in which the likelihood $p_\theta(\mathbf{x}|\vec{\mathbf{z}})$ depends on *all* the latent variables in the chain $\vec{\mathbf{z}}$, rather than just the bottom layer $\mathbf{z}^1$, as has been done in Kingma et al. (2016); Maaløe et al. (2019); Vahdat & Kautz (2020); Child (2021), again as discussed in § 3.3.2. This leads to the following factorisation of the generative structure:

$$p_\theta(\mathbf{x}, \vec{\mathbf{z}}) = p_\theta(\mathbf{x}|\vec{\mathbf{z}}) \prod_{i=1}^{L-1} p_\theta(\mathbf{z}^i|\mathbf{z}^{i+1}) p(\mathbf{z}^L). \tag{5.3}$$

To construct the ELBO, we must further introduce an inference network $q_\phi(\vec{\mathbf{z}}|\mathbf{x})$. On the basis of simplicity and that it produces effective empirical performance, we use the factorisation:

$$q_\phi(\vec{\mathbf{z}}|\mathbf{x}) = q_\phi(\mathbf{z}^1|\mathbf{x}) \prod_{i=1}^{L-1} q_\phi(\mathbf{z}^{i+1}|\mathbf{z}^i, \mathbf{x}), \tag{5.4}$$

where each conditional distribution $q_\phi(\mathbf{z}^{i+1}|\mathbf{z}^i, \mathbf{x})$ takes the form of a Gaussian[1]. Again, marginalising out intermediate $\mathbf{z}^i$ layers, $q_\phi(\mathbf{z}^L|\mathbf{x})$ is a non-Gaussian, highly flexible distribution. To defend this model against adversarial attack, we apply TC regularisation term as per the last section. We refer to the resulting models as Seatbelt-VAEs. We obtain a decomposition of the ELBO for this model, revealing the existence of a TC term for the top-most layer (see Appendix D.1 for proof).

Theorem 5.1. *The Evidence Lower Bound, for a hierarchical VAE with forward model as in Eq* (5.3) *and amortised variational posterior as in Eq* (5.4)*, can be decomposed to reveal the total correlation (see Definition 4.2), of the aggregate posterior of the top-most layer of latent variables:*

$$\mathcal{L}(\mathcal{D}; \theta, \phi) = \mathbb{E}_{q(\vec{\mathbf{z}}, \mathbf{x})} \log p_\theta(\mathbf{x}|\vec{\mathbf{z}}) + \textcircled{\text{R}} + \textcircled{\text{S}_a} + \textcircled{\text{S}_b} - D_{\mathrm{KL}}\Big(q(\mathbf{z}^L) || \prod\nolimits_j q(z_j^L)\Big), \tag{5.5}$$

where the last term is the required TC term, and, using j to index over the coordinates in $\mathbf{z}^L$,

$$\textcircled{\text{R}} = \int \mathrm{d}\mathbf{x} \prod\nolimits_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\vec{\mathbf{z}}|\mathbf{x}) q(\mathbf{x}) \log \frac{\prod_{k=1}^{L-1} p_\theta(\mathbf{z}^k|\mathbf{z}^{k+1})}{q_\phi(\mathbf{z}^1|\mathbf{x}) \prod_{m=1}^{L-2} q_\phi(\mathbf{z}^{m+1}|\mathbf{z}^m, \mathbf{x})} \tag{5.6}$$

$$\textcircled{\text{S}_a} = - \mathbb{E}_{q_\phi(\mathbf{z}^{L-1})} D_{\mathrm{KL}}(q_\phi(\mathbf{z}^L, \mathbf{x}|\mathbf{z}^{L-1}) || q_\phi(\mathbf{z}^L) q(\mathbf{x})) \tag{5.7}$$

$$\textcircled{\text{S}_b} = - \sum\nolimits_j D_{\mathrm{KL}}(q_\phi(\mathbf{z}_j^L) || p(\mathbf{z}_j^L)). \tag{5.8}$$

[1]We also experimented with our posterior factorised as in § 3.3.2, and found overall results to be very similar.

and $q(\mathbf{x})$ is the empirical data distribution.

In other words, following the Factor and β-TCVAEs, we up-weight the TC term for $\mathbf{z}^L$. We can upweight this term then recombine the decomposed parts of the ELBO, to give us the following compact form of this objective.

Definition 5.1. *A Seatbelt-VAE is a hierarchical VAE with forward model as in Eq* (5.3) *and amortised variational posterior as in Eq* (5.4)*, trained w.r.t. its parameters θ, ϕ to maximise the objective*

$$\mathcal{L}^{\text{Seatbelt}}(\mathcal{D}; \beta, \theta, \phi) := \mathbb{E}_{q_\phi(\vec{\mathbf{z}}, \mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}, \vec{\mathbf{z}})}{q_\phi(\vec{\mathbf{z}}|\mathbf{x})}\right] - (\beta - 1) D_{\text{KL}}\Big(q(\mathbf{z}^L)||\prod_j q(z_j^L)\Big) \tag{5.9}$$

for a chosen, fixed value of β, where $q(\mathbf{x})$ is the empirical data distribution.

We see that when $L = 1$ a Seatbelt-VAE reduces to a β-TCVAE. We use the $\beta = 1$ case as a baseline in our experiments as it corresponds to a Vanilla VAE for $L = 1$ and for $L > 1, \beta = 1$ it produces a hierarchical model with a likelihood function conditioned on all latents.

As with the β-TCVAE, training $\mathcal{L}^{\text{Seatbelt}}_{\mathcal{D};\beta,\theta,\phi}$ using stochastic gradient ascent with minibatches of the data is complicated by the presence of aggregate posteriors $q_\phi(\mathbf{z})$ which depend on the entire dataset. To deal with this, Appendix D.2 we derive a minibatch estimator for TC-penalised hierarchical VAEs, building off that used for β-TCVAEs Chen et al. (2018a). Like the estimator from Chen et al. (2018a) that it builds off, however, it is a biased estimator: it is a nested expectation, for which unbiased, finite–variance, estimators do not generally exist (Rainforth et al., 2018). Consequently, moderate to large batch sizes are needed; for small batch sizes its practical behaviour mimics that of the β-VAE (Mathieu et al., 2019). In practice we did not find this to cause us any problems.

We note that, as in Chen et al. (2018a), large batch sizes are generally required to provide accurate TC estimates.

Attacking Hierarchical TC–Penalised VAEs In the above hierarchical model the likelihood over data is conditioned on all layers, so manipulations to any layer have the potential to be significant. We focus on simultaneously attacking all layers, noting that, as shown in Appendix D.3, this is more effective that just targeting the top or base layers individually. Hence our adversarial objective for Seatbelt-VAEs is the following generalisation of that introduced in Tabacof et al. (2016); Gondim-Ribeiro et al. (2018); Kos et al. (2018), to attack all the layers at the same time:

$$\Delta_r^{\text{Seatbelt}}(\mathbf{x}, \mathbf{d}, \mathbf{x}^t; \lambda) = \lambda||\mathbf{d}||_2 + \sum_{i=1}^{L} r(q_\phi(\mathbf{z}^i|\mathbf{x}+\mathbf{d}), q_\phi(\mathbf{z}^i|\mathbf{x}^t)). \tag{5.10}$$

The attack via the ELBO, Eq (5.2), generalises trivially to hierarchical VAEs.

5.6 Experiments

Expanding on the brief experiments in Section 5.4.2, we perform a battery of adversarial attacks on each of the introduced models. We do this for three different adversarial attacks: first, (as in § 5.4.2) a latent attack, Eqs (5.1,5.10) using the D_{KL} divergence between attacked and target posteriors; second, we attack via the model's output, aiming to make the target maximally likely under the attacked model as in Eq (5.2); finally, a new latent attack method as per Eqs (5.1,5.10) where we use $r(\cdot,\cdot) = W_2(\cdot,\cdot)$, the 2-Wasserstein distance between attacked and target posteriors.

We then evaluate the effectiveness of these attacks in three ways. First, like Fig 5.1, we can plot the attacks themselves, to see how effective these attacks are in fooling us. Secondly, we can measure the adversary's loss under the attack objective. Thirdly, we give the negative adversarial likelihood of the target image $\mathbf{x}^t$ given an attacked latent representation $\mathbf{z}^*$. Larger, more positive, values of $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ correspond to less successful attacks as they correspond to large distances between the target and the adversarial reconstruction. Lower values correspond to successful attacks as they correspond to a small distance between the adversarial target and the reconstruction. We also measure reconstruction quality of these models, as a function of degree of regularisation. Finally, we also measure how downstream tasks

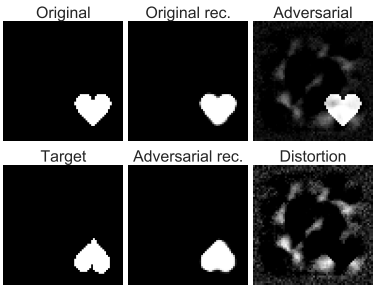
that use output of these models perform under attack. We train classifiers, on the reconstructions and on the latent representations, and see how robust performance is when the upstream VAE is attacked.

We demonstrate that our hierarchical TC–Penalised VAEs (Seatbelt-VAEs) confer superior robustness to β-TCVAEs and standard VAEs, while preserving the ability to reconstruct inputs effectively. Through this, we demonstrate that they are a powerful tool for learning robust deep generative models.

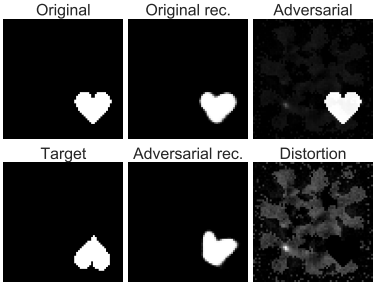
Following previous work (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018) we randomly sample 10 input-target pairs for each dataset and for each image pair we consider 50 different values of λ geometrically-distributed from 2^{-20} to 2^{20}. Thus each individual trained model undergoes 500 attacks for each attack mode. As before, we used L-BFGS-B for gradient descent (Byrd et al., 1995). We perform these experiments on Chairs (Aubry et al., 2014), 3D faces (Paysan et al., 2009), and CelebA (Liu et al., 2015). Details of neural architectures and training are given in Appendix D.6.

5.6.1 Visual Appraisal of Attacks

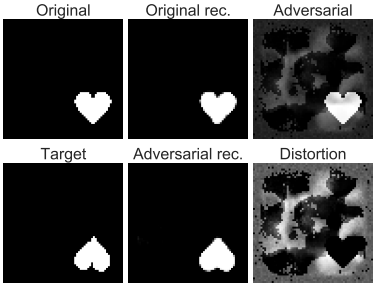
We first visually appraise the effectiveness of attacks that use the D_{KL} divergence on vanilla VAEs, β-TCVAEs, and Seatbelt-VAEs. As mentioned in Section 1, Fig 5.1 shows the results of latent space attacks on three models trained on CelebA. It is apparent that the β-TCVAE provides additional resilience to the attacks compared with the standard VAE. Furthermore, this figure shows that Seatbelt-VAEs are sufficiently robust to almost completely thwart the adversary: its adversarial construction still resembles the original input. Moreover, this was achieved while also producing a clearer non–adversarial reconstruction. One might expect attacks targeting a single generative factor underpinning the data to be easier. However, we find that these models protect effectively against this as well. For example, see Fig 5.5 for plots showing an attacker attempting to rotate a dSprites heart.



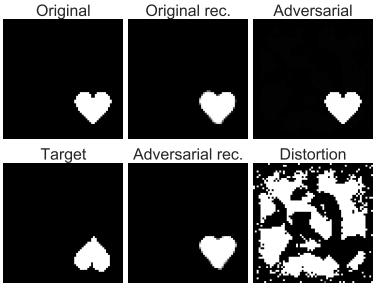
(a) β-TCVAE, β=1



(b) β-TCVAE, β=2



(c) SB-VAE, β=1



(d) SB-VAE, $\beta = 2$

Figure 5.5: D_{KL} latent space attacks *only on rotation* of a heart-shaped dSprite for β-TCVAEs ($d_{\mathbf{z}} = 64$) and Seatbelt-VAEs ($L = 2$) for $\beta = \{1, 2\}$. The attacks are conducted by applying a distortion (third column of each image) to the original image (top first column) to produce an adversarial input (bottom second column of each image) to try to cause the output of the target image (bottom first column). Here we show the most successful adversarial distortion in terms of adversarial loss for each model. It is apparent that Seatbelt-VAEs are the most resilient to attack. Note that the distortions plots (bottom right) are scaled to [0,1] for ease of viewing.

In both figures we follow the method of Gondim-Ribeiro et al. (2018) to plot attacks. Those shown are representative of the adversarial inputs the attacker was able to find over the 50 different values of λ. The Seatbelt-VAE input only undergoes a small perturbation because it is sufficiently robust that the attacker is not able to make the reconstruction look more like the target image in any meaningful way, such that the optimiser never drifts far from the initial input. Note that the β-TCVAE is also robust here. The attacker is unable to induce the desired adversarial reconstruction, even though the attack may be of large magnitude. In contrast, attacks on vanilla-VAEs are able to move through the latent space and find a perturbation that reconstructs to the adversary's target image.

5.6.2 Quantitative Analysis of Robustness

Having ascertained perceptually that Seatbelt-VAEs offer the strongest protection to adversarial attack, we now demonstrate this quantitatively. Fig 5.6 shows $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ and Δ over a range of datasets and βs for Seatbelt-VAEs ($L = 4$) and β-TCVAEs for our three different attacks. It demonstrates that the combination of depth and high TC-penalisation offers the best protection to adversarial attacks and that the hierarchical extension confers much greater protection to adversarial attack than a single layer β-TCVAE. For Seatbelt-VAEs, as we go to the largest values of β for both Chairs and 3D Faces, adversarial loss Δ_{KL} grows by a factor of $\approx 10^7$ and $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ for those attacks doubles. For all attacks, TC-penalised models outperformed standard VAEs (β=1) and Seatbelt-VAEs outperform single-layer VAEs. β-TCVAEs do not experience such a large uptick in adversarial loss and negative adversarial likelihood. These results show that the hierarchical approach can offer very strong protection from the adversarial attacks studied.

In Appendix D.3 we provide plots detailing these metrics for a range of L values. In Appendix D.4 we also calculate the L_2 distance between target images and adversarial outputs and show that the loss of effectiveness of adversarial attacks is not due to the degradation of reconstruction quality from increasing β. We also test

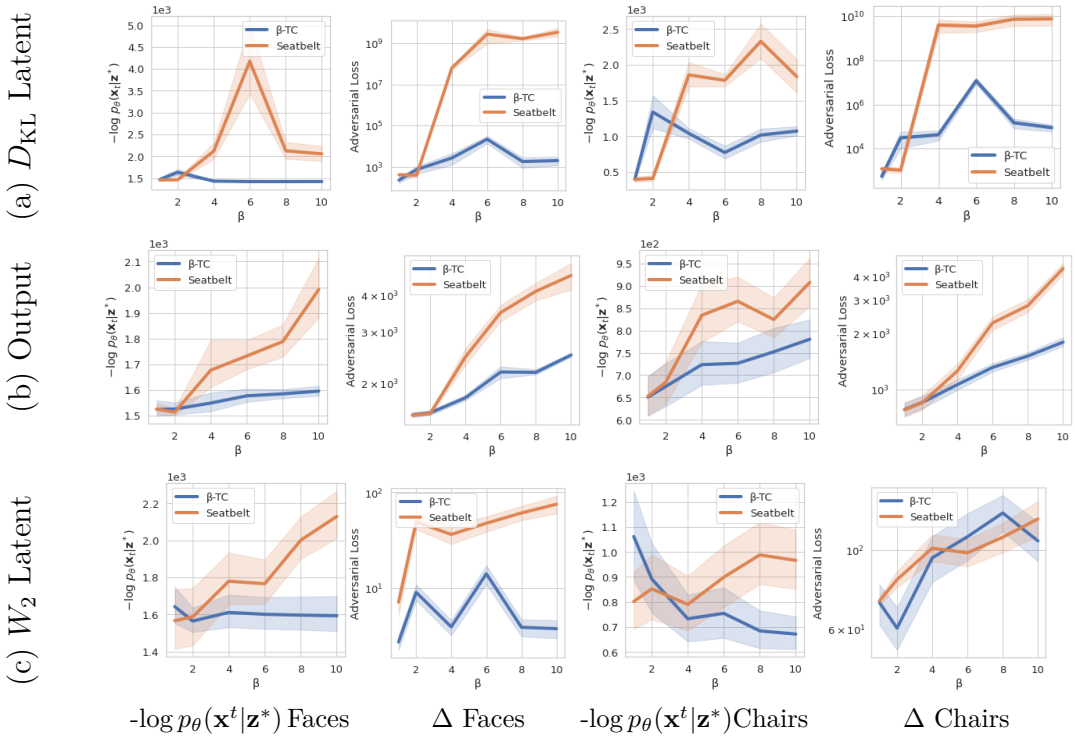
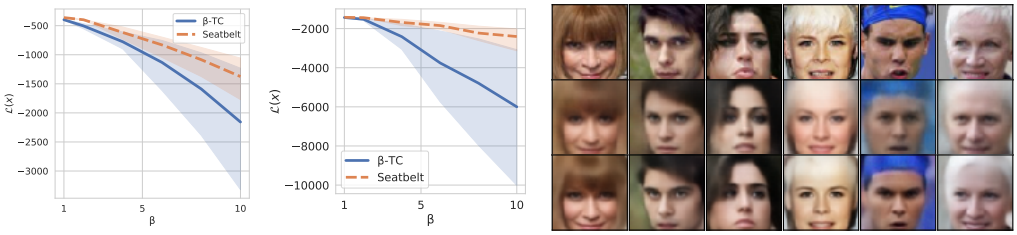


Figure 5.6: Plots showing the robustness of Seatbelt-VAEs (L=4) and β-TCVAEs models for different values of β for three different attack methods: a) Latent space attack via D_{KL} in Eqs (5.1,5.10), b) Attack via the model output as in Eq 5.2, and c) Latent space attack via the 2-Wasserstein (W_2) distance in Eqs (5.1,5.10). Note that the β-TCVAE with $\beta = 1$ corresponds to a vanilla VAE and that $L > 1$ $\beta = 1$ models correspond to hierarchical baselines. We show the negative adversarial likelihood of a target image $\mathbf{x}^t$ given an attacked latent representation $\mathbf{z}^*$ for Faces (1$^{\text{st}}$ col) and Chairs (3$^{\text{rd}}$ col) respectively. Larger values of $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ mean less successful adversarial attacks. We also show the adversarial loss Δ in 2$^{\text{nd}}$ and 4$^{\text{th}}$ cols, which have a logarithmic axis. Shading in results corresponds to the 95% CI over variation for 10 images for each combination of $d_{\mathbf{z}} = \{4, 8, 16, 32, 64, 128\}$ and λ taking 50 geometrically distributed values between 2^{-20} and 2^{20}.

VAE robustness to random noise. We noise the inputs and evaluate the model's ability to reconstruct the original input. Through this we are evaluating their ability to denoise. See Appendix D.5 for an illustration of this for TC-penalised models. The ability to denoise is linked to robustness to attacks, as we will discuss in great detail in the next chapter.

ELBO and Reconstructions Though Seatbelt-VAEs offer better protection to adversarial attack than β-TCVAEs, we also motivate their utility by way of their reconstruction quality. In Fig 5.7 we plot the ELBO of the two TC-penalised models,



(a) Chairs ELBO **(b)** 3D Faces ELBO **(c)** Reconstructions

Figure 5.7: Effect of varying β on the reconstructions of TC-penalised models. In subfigures (a) and (b) we plot the final ELBO of TC-penalised models trained on the Chairs and 3D faces, calculated *without* the β penalisation applied during training. Shading gives the 95% CI over variation due to variation of $d_{\mathbf{z}} = \{32, 64, 128\}$ for β-TCVAE and also $L = \{2, 3, 4, 5\}$ for Seatbelt. As β increases $\mathcal{L}$ degrades more slowly for Seatbelt-VAE, relative to β-TCVAE, (c) serves as a visual confirmation of these results. The top row shows CelebA input data. The bottom row, the reconstructions from a Seatbelt-VAE with $L = 4$ and $\beta = 20$, clearly maintains facial identity better than those from a β-TCVAE, the middle row: many of the individuals' finer facial features lost by the β-TCVAE are maintained by the Seatbelt-VAE.

calculated *without* the β penalisation that was applied during training. We further show the effect of depth and TC-penalisation on CelebA reconstructions. These plots show that Seatbelt-VAEs' reconstructions are more resilient to increasing β than β-TCVAEs'.

5.6.3 Protection to Downstream Tasks

Finally, we consider the protection that Seatbelt-VAEs might provide to downstream tasks, noting that VAEs are often used as subcomponents in larger machine learning systems (Higgins et al., 2017b), or as a mechanism to protect another model from attack (Schott et al., 2019; Ghosh et al., 2019). Table 5.1 shows results for classification tasks using 2-layer MLPs and fully-convolutional nets trained on the reconstructions or on the embeddings. It shows the drop in accuracy caused by an adversary that picks a target with a different label and attacks the VAEs' embedding using the attack objective with $\lambda = 1$. We see that Seatbelt-VAEs produced significantly better accuracies under these attacks.

Table 5.1: Robustness of downstream classification tasks under adversarial attack. We consider classifiers trained either on the reconstructed image (denoted $p(y|\tilde{\mathbf{x}})$) or on the latent representations ($p(y|\mathbf{z})$). We show accuracy when the model is attacked, resulting in perturbed embeddings $\mathbf{z}^*$ and reconstructions ($\tilde{\mathbf{x}}^*$). Parentheses show the drop in accuracy resulting from the attack – the smaller the drop in magnitude the better

		Accuracy by Model		
Dataset	Task	VAE	β–TCVAE	Seatbelt-VAE
SVHN	$p_{\text{MLP}}(y\|\tilde{\mathbf{x}})$	0.17 (-0.35)	0.22 (-0.29)	**0.35** (**-0.15**)
	$p_{\text{Conv}}(y\|\tilde{\mathbf{x}})$	0.13 (-0.54)	0.36 (-0.28)	**0.41** (**-0.26**)
	$p_{\text{MLP}}(y\|\mathbf{z})$	0.15 (-0.57)	0.46 (-0.23)	**0.57** (**-0.21**)
CIFAR10	$p_{\text{MLP}}(y\|\tilde{\mathbf{x}})$	0.17 (-0.32)	0.25 (-0.21)	**0.38** (**-0.09**)
	$p_{\text{Conv}}(y\|\tilde{\mathbf{x}})$	0.07 (-0.37)	0.32 (-0.10)	**0.34** (**-0.07**)
	$p_{\text{MLP}}(y\|\mathbf{z})$	0.16 (-0.41)	0.26 (-0.23)	**0.39** (**-0.09**)

5.7 Conclusion

We have shown that VAEs can be rendered more robust to adversarial attacks by regularising the evidence lower bound. This increase in robustness can be strengthened by extending these regularisation methods to hierarchical VAEs, forming Seatbelt-VAEs, which uses a generative structure where the likelihood makes use of all the latent variables. Designing robust VAEs is becoming pressing as they are increasingly deployed as subcomponents in larger pipelines. As we have shown, methods typically used for disentangling, motivated by their ability to provide interpretable representations, also confer robustness.

People who make no noise are dangerous.

— Jean de La Fontaine

6

Towards a Theoretical Understanding of the Robustness of Variational Autoencoders

Contents

6.1 Motivation

We make inroads into understanding the robustness of Variational Autoencoders to adversarial attacks and other input perturbations. While in the previous chapter

we have developed algorithmic approaches to attacking and defending VAEs, we have not formalised what it means for a VAE to be robust. To address this, we develop a novel criterion for robustness in probabilistic models: r-robustness. We then use this to construct the first theoretical results for the robustness of VAEs, deriving margins in the input space for which we can provide guarantees about the resulting reconstruction. Informally, we are able to define a region within which any perturbation will produce a reconstruction that is similar to the original reconstruction. To support our analysis, we show that VAEs trained using disentangling methods not only score well under our robustness metrics, but that the reasons for this can be interpreted through our theoretical results.

6.2 Introduction

Variational Autoencoders (Rezende et al., 2014; Kingma & Welling, 2014) have been found to be more robust to input perturbations than their deterministic counterparts, particularly those originating from adversarial attacks (Schott et al., 2019; Ghosh et al., 2019). This trait has made them useful in protecting downstream tasks (Schott et al., 2019; Ghosh et al., 2019).

Nevertheless, they are still not completely impervious to attack (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018; Kos et al., 2018): a hypothetical adversary can attack a VAE by applying small input perturbations take look to invoke meaningful changes in the encoding. Typically this is done by trying to find perturbations which produce reconstructions close to a distinct target datapoint chosen by the adversary, rather than being representative of the original input. Such attacks have been shown to be successful in a wide range of scenarios (Tabacof et al., 2016; Gondim-Ribeiro et al., 2018; Kos et al., 2018). In the previous chapter we have made progress towards defending against them from an empirical and algorithmic perspective, by repurposing approaches designed to learn disentangled latent representations (Burgess et al., 2017; Chen et al., 2018a; Mathieu et al., 2019).

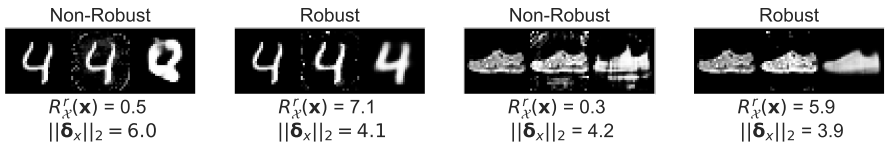


Figure 6.1: Reconstructions under attack for robust and non-robust VAEs. Each subfigure shows from left to right: the original input, a perturbed input made by an adversarial attack, and the reconstruction of the perturbed input. We show results for VAEs that are robust ($R^r_{\mathcal{X}}(\mathbf{x}) \geq \|\boldsymbol{\delta}_x\|_2$) and non-robust ($R^r_{\mathcal{X}}(\mathbf{x}) < \|\boldsymbol{\delta}_x\|_2$) for a given point $\mathbf{x}$ and adversarially selected perturbation $\boldsymbol{\delta}_x$. We see that the robust VAE reconstructions are visually closer to the original input.

However, a deeper understanding of the mechanisms underpinning the robustness of VAEs and their derivatives is still lacking. Furthermore, there are currently no theoretical foundations for this robustness or even any frameworks or formalisations for exactly what it means for a VAE to be "robust." In other words, what would it mean to have a certifiably–robust VAE? Moreover, are there scenarios where we might be able to provide theoretical guarantees of such robustness?

As a first step to addressing these questions, we develop the first metric with which to evaluate the robustness of VAEs: r-robustness. Informally, for a given input, a VAE is r-robust to a given perturbation if it is more likely that its reconstruction will fall within a ball of radius r around the undistorted maximum likelihood reconstruction, than outside it. The smaller the value of r for which we can confirm r-robustness, the more robust we can guarantee the VAE to be. Through r-robustness, we provide theoretical foundations to understand the source of VAEs' robustness and provide insights into what can cause them to more or less robust.

Using this, we next develop a *margin* of robustness, $R^r_{\mathcal{X}}(\mathbf{x})$, such that the VAE is r-robust to *any* possible perturbations of the input $\mathbf{x}$ within this margin. This, in turn, allows us to provide a notion of a certifiably–robust reconstruction as it forms a guarantee that no attack limited to the margin can reliably undermine it. An example of this is shown in Fig 6.1, where we demonstrate that large $R^r_{\mathcal{X}}(\mathbf{x})$ are associated with model–input pairs that are robust to adversarially generated input perturbations. Analogously to the concept of an adversarial risk (Uesato et al., 2018), $R^r_{\mathcal{X}}(\mathbf{x})$ can further be converted to a metric for the *overall* robustness of a VAE, by taking its expectation over the data generating distribution.

To make inroads towards imposing a priori constraints on a VAE that ensure that it is certifiably robust, we further derive a theoretical bound for $R^r_{\mathcal{X}}(\mathbf{x})$ as a function of the encoder variance and Jacobian. This provides insights into the characteristics of VAEs that contribute to robustness. Building on this result, we show empirically that VAEs with larger encoder variances and smaller Jacobians typically produce larger margins $R^r_{\mathcal{X}}(\mathbf{x})$ and are thus more robust to perturbations. We further demonstrate how these beneficial characteristics can be induced using methods introduce to learn disentangled representations, deriving new results for how these methods can be interpreted.

To summarise, our core contributions are that we first define a robustness metric, r-robustness, that is tailored to probabilistic generative models. We develop a margin $R^r_{\mathcal{X}}(\mathbf{x})$ on a VAE's input space within which it is r-robust to perturbations. Finally, we offer theoretical and empirical analysis—based on $R^r_{\mathcal{X}}(\mathbf{x})$ and existing disentanglement methods—that can aid the construction of robust VAEs.

6.3 Robustness of VAEs

6.3.1 A Probabilistic Metric of Robustness

Deep learning models can be brittle. Some of the most sophisticated deep learning classifiers can be broken by simply adding small perturbations to their inputs (Szegedy et al., 2014; Shamir et al., 2019; Goodfellow et al., 2015; Papernot et al., 2017; Moosavi-Dezfooli et al., 2017). Here, perturbations that would not fool a human break neural network predictions. A model's weakness to such perturbations is called its *sensitivity*. For classifiers, we can straightforwardly define an associated *sensitivity margin*: it is the radius of the largest metric ball centred on an input $\mathbf{x}$ for which a classifier's original prediction holds for all possible perturbations within that ball.

Defining such a margin for VAEs is conceptually more difficult as, in general, the reconstructions are, in general, continuous rather than discrete. To put it another

way, there is no step-change in VAE reconstructions that is akin to a change of a predicted class in classifiers; *any* perturbation in the input space will result in a change in the VAE output. To complicate matters further, a VAE's latent space is stochastic: the same input can result in different reconstructions.

As a first step to deriving robustness margins for VAEs, we now introduce a criterion for measuring robustness in probabilistic models: r-robustness. We start by presenting it in the general setting, before linking it to the specific case of VAEs.

Definition 6.1. *A model, f, operating on a point $\mathbf{x}$, that outputs a continuous random variable is r-robust for $r \in \mathbb{R}^+$, to a perturbation $\boldsymbol{\delta}$ and for an arbitrary norm $||\cdot||$ iff*

$$p(||f(\mathbf{x}+\boldsymbol{\delta}) - f(\mathbf{x})|| \leq r) > p(||f(\mathbf{x}+\boldsymbol{\delta}) - f(\mathbf{x})|| > r).$$

We will assume from now on that the norm is taken to be the 2-norm $||\cdot||_2$, such that r-robustness determines a bound for which changes in the output $f(\mathbf{x})$ induced by the perturbation $\boldsymbol{\delta}$ are more likely to fall within the hyper-sphere of radius r, than not. As r decreases, the criterion for model robustness becomes stricter. We note that r-robustness can be viewed as a probabilistic analogy to the criterion for regression models presented by Nguyen & Raff (2019). We also note that r-robustness can be generalised to $p(||f(\mathbf{x}+\boldsymbol{\delta}) - f(\mathbf{x})||_2 \leq r) > m$, where $1-m$ is the allowable risk (with $m = 0.5$ in Definition 6.1).

Because this criterion is applicable to probabilistic models with continuous output spaces, it is directly relevant for ascertaining robustness in VAEs. By considering the smallest r for which the criterion holds, we can think of it as a metric that provides a probabilistic measure of the *extent* to which outputs are altered given a corrupted input: the smaller the value of r for which we can confirm r-robustness, the more robust the model.

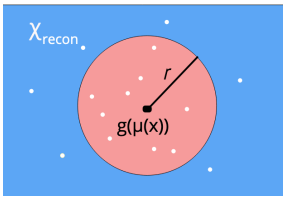


Figure 6.2: Illustration of r-robustness in a VAE. White dots represent possible reconstructions, with the diversity originating from the encoder stochasticity. For r-robustness to hold, the probability of our reconstruction falling within the red area—a hypersphere of radius r centred on $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$—needs to be greater than or equal to the probability of falling outside.

6.3.2 A Robustness Margin for VAEs

We want to define a margin in a VAE's input space for which it is robust to perturbations of a given input. Perturbations that fall within this margin should not break our criterion for robustness. Formally, we want a margin in $\mathcal{X}$, $R^r_{\mathcal{X}}(\mathbf{x})$, for which any distorted input $\mathbf{x} + \boldsymbol{\delta}_x$, where $||\boldsymbol{\delta}_x||_2 < R^r_{\mathcal{X}}(\mathbf{x})$ is the perturbation, satisfies r-robustness when reconstructed.

However, to consider the robustness of VAEs, we must not only take into account the perturbation $\boldsymbol{\delta}_x$, but also the stochasticity of encoder. We can think of the decoder as taking in noisy inputs because of this stochasticity. Naturally, this noise can itself potentially cause issues in the robustness of VAE: if the level of noise is too high, we will not achieve reliable reconstructions even without perturbing the original inputs. As such, before even considering perturbations, we first need to adapt our r-robustness framework to deal with this stochasticity.

6.3.2.1 r-robustness for VAEs

Given an input $\mathbf{x}$, r-robustness dictates that we want to define some region in the reconstruction space, $\mathcal{X}_{\text{recon}}$, within which most of the decoded samples from the latent embedding $\mathbf{z}$ will fall. We will assume that the encoder is a Gaussian as this is standard practice. Denoting $g_\theta(\mathbf{z})$ as the deterministic mapping inside the VAE's decoder network and $\boldsymbol{\mu}_\phi(\mathbf{x})$ as the mean embedding of the encoder, we can define $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$ to be the "maximum likelihood" reconstruction, noting this is a deterministic function. Our aim is now to find a hyper-sphere of radius r centred

on $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$ within which most of the possible VAE outputs for a given point $\mathbf{x}$ lie. Larger r are indicative of a greater variance in the encoding process, and as such are likely to be associated with poorer quality reconstructions.

Denoting $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ as the reparameterised stochasticity of the encoder, we define the distance from the maximum likelihood reconstruction, induced by this sampling as

$$\Delta(\mathbf{x}) = g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x})) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x})) \tag{6.1}$$

Using this, we see that a VAE is r-robust to the stochasticity of the encoder iff (see also Fig 6.2)

$$p(||\Delta(\mathbf{x})||_2 \leq r) > p(||\Delta(\mathbf{x})||_2 > r). \tag{6.2}$$

Informally, we want it to be more probable for reconstructions to fall within this radius r than not.

6.3.2.2 Robustness to distortions in data-space

Given that we have established conditions for r in Eq (6.2) that take into account latent space sampling, we can now return to our original objective, which was to determine a margin in the data-space $\mathcal{X}$ for which a VAE is robust to perturbations on its input. Recall that this implicitly means that we want to define a bound for robustness given two sources of perturbations: the stochasticity of the encoder, and a hypothetical input perturbation $\boldsymbol{\delta}_x$.

For simplicity of analysis, we consider the case where the perturbation is applied only to the encoder mean input and not the encoder variance input, noting that the latter is typically stable across inputs and so is less of a concern. In Fig 6.4 we demonstrate that adversarial attacks on VAE encoders are dominated by the perturbation to the embedding mean that is induced, thereby justifying this assumption. We note also that one can usually also simply fix the encoder variance to a constant for all datapoints without incurring substantial performance drops (Ghosh et al., 2020), thereby providing a means to ensure this assumption holds exactly if needed.

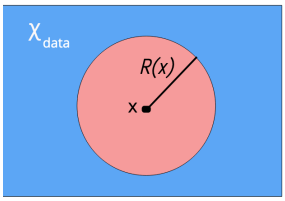
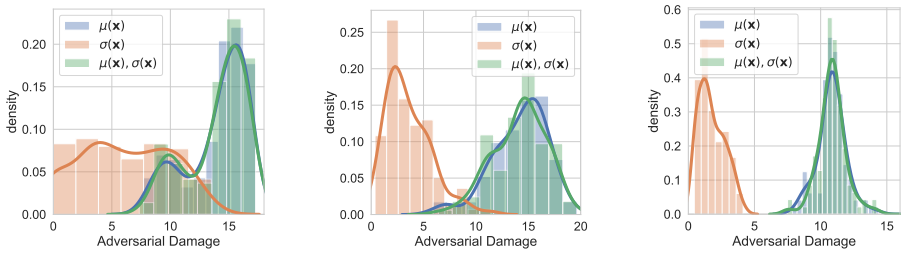


Figure 6.3: Illustration of the margin $R^r_{\mathcal{X}}(\mathbf{x})$, which is defined in the **input** space $\mathcal{X}$. Red represents represents the subspace where the model is r-robust, such that $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r) > p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 > r)$ holds for all $\mathbf{x} + \boldsymbol{\delta}_x$ falling in this region, that is all $\boldsymbol{\delta}_x : \|\boldsymbol{\delta}_x\|_2 \leq R^r_{\mathcal{X}}(\mathbf{x})$.



(a) MNIST **(b)** fMNIST **(c)** CIFAR10

Figure 6.4: *Maximum damage* adversarial attacks (see Eq (6.5)) on multiple VAEs trained on MNIST (a), fashion-MNIST (b), and CIFAR10 (c). We attack 25 datapoints for each VAE and propagate the attacks to the encoder mean ($\mu(\mathbf{x})$), the encoder standard deviation ($\sigma(\mathbf{x})$), or both ($\mu(\mathbf{x}), \sigma(\mathbf{x})$). Attack norms are capped to 10. Shown are distribution plots of the adversarial damage, i.e. the L_2 distance between the reconstruction resulting from the attack and the maximum likelihood reconstruction $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$. Clearly attacks on $\mu(\mathbf{x})$ are more harmful than on $\sigma(\mathbf{x})$, and most of the damage from attacks on both $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ stems from the attack on $\mu(\mathbf{x})$.

We define the distance from the maximum likelihood reconstruction, $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$, induced by the stochasticity of the encoder *and* an input perturbation $\boldsymbol{\delta}_x$ as

$$\Delta(\mathbf{x}, \boldsymbol{\delta}_x) = g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x} + \boldsymbol{\delta}_x) + \boldsymbol{\eta}\sigma_\phi(\mathbf{x})) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x})).$$

We can now define the condition for which r-robustness is satisfied on the VAE output given the two sources of perturbation as

$$||\boldsymbol{\delta}_x||_2 < R^r_{\mathcal{X}}(\mathbf{x}) \;\Leftrightarrow\; p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r) > 0.5 \tag{6.3}$$

Thus, $R^r_{\mathcal{X}}(\mathbf{x})$ is the margin of robustness of the VAE such that $\forall \boldsymbol{\delta}_x \;:\; ||\boldsymbol{\delta}_x||_2 < R^r_{\mathcal{X}}(\mathbf{x})$, $\mathbf{x} + \boldsymbol{\delta}_x$ is more likely than not to be reconstructed within a radius r of the maximum likelihood reconstruction $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$. A high level illustration of this is given in Fig 6.3, and Fig 6.5 shows a simple empirical demonstration of how $R^r_{\mathcal{X}}(\mathbf{x})$ relates to the probability of producing a good reconstruction under random input perturbations.

We note that, analogously to the concept of an adversarial risk (Uesato et al., 2018), $R^r_{\mathcal{X}}(\mathbf{x})$ can further be converted to a metric for the *overall* robustness of a VAE, by taking its expectation over the data generating distribution, namely $R^r_{\mathcal{X}} = \mathbb{E}_{\mathcal{D}}\left[R^r_{\mathcal{X}}(\mathbf{x})\right]$.

6.3.3 Characterising the Margin

Given this definition, we now wish to try and characterise $R^r_{\mathcal{X}}(\mathbf{x})$. In particular, we would like to understand what characteristics of the VAE are likely to make it relatively larger or smaller. Ideally, we also want to establish scenarios where we might be able to provide guarantees of a minimum size for $R^r_{\mathcal{X}}(\mathbf{x})$, such that we might be able to make inroads into how one might a priori construct a certifiably-robust VAE.

A perturbation in $\mathcal{X}$, $\boldsymbol{\delta}_x$, induces a perturbation in $\mathcal{Z}$, $\boldsymbol{\delta}_z$. To determine the margins for robustness in $\mathcal{X}$, we first apply the Neyman-Pearson lemma (Neyman & Pearson, 1933; Cohen et al., 2019), assuming a "worst-case" decoder. This decoder has subspaces in $\mathcal{Z}$, where it is is either robust or non-robust, that are divided by a boundary that is normal to both the induced perturbation $\boldsymbol{\delta}_z$ and to the dimension of minimal variance in $\mathcal{Z}$, $\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i$. We then determine the minimum perturbation norm in $\mathcal{X}$ which induces a perturbation in $\mathcal{Z}$ that crosses this boundary.

Theorem 6.1. *Consider a VAE with a diagonal-variance Gaussian encoder, an input* $\mathbf{x}$*, and an output margin* $r \in \mathbb{R}$ *such that the VAE is* r*-robust to the stochasticity of the encoder when the* $\mathbf{x}$ *is unperturbed as per* (6.2)*. Assuming standard regularity assumptions (discussed in the proof) hold for* $\boldsymbol{\mu}_\phi(\mathbf{x})$*, then*

$$R^r_{\mathcal{X}}(\mathbf{x}) \geq \frac{(\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i)\Phi^{-1}(p(||\Delta(\mathbf{x})||_2 \leq r))}{||\mathbf{J}^{\mu}_{\phi}(\mathbf{x})||_F} + \mathcal{O}(\boldsymbol{\varepsilon}) \tag{6.4}$$

where $\mathcal{O}(\boldsymbol{\varepsilon})$ *represents higher order dominated terms that disappear in the limit of small perturbations,* Φ^{-1} *is the probit function,* $\mathbf{J}^{\mu}_{\phi}(\mathbf{x})_{i,j} = \partial \boldsymbol{\mu}_\phi(\mathbf{x})_i / \partial \mathbf{x}_j$ *is the Jacobian of* $\boldsymbol{\mu}_\phi(\mathbf{x})$*, and* $||\cdot||_F$ *is the Frobenius norm.*

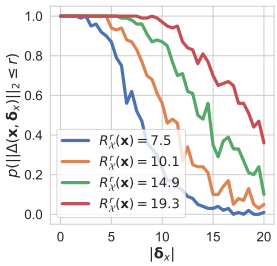
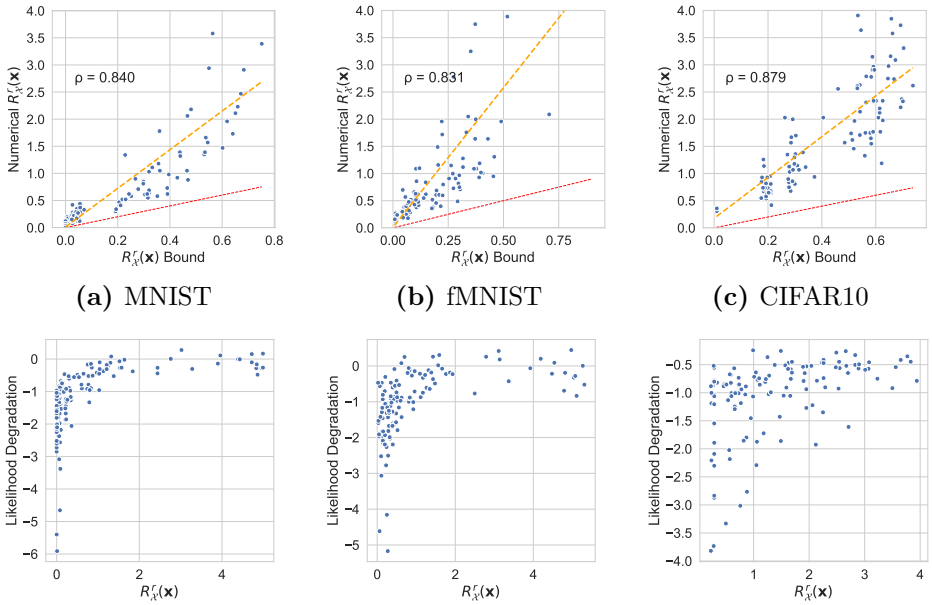


Figure 6.5: $R^r_{\mathcal{X}}(\mathbf{x})$ for four VAEs of varying robustness trained on MNIST. We fix the input $\mathbf{x}$ and perturbation direction $\boldsymbol{\delta}_x / \|\boldsymbol{\delta}_x\|_2$, but vary the perturbation size $\|\boldsymbol{\delta}_x\|_2$. We assess the proportion of samples which fall within $r = 4$ of the maximum likelihood reconstruction.

The proof is provided in Appendix E.2. This bound is based on a first order approximation of $\boldsymbol{\mu}_\phi(\mathbf{x} + \boldsymbol{\delta}_x)$ around the original input $\mathbf{x}$; the impact of $\mathcal{O}(\boldsymbol{\varepsilon})$ thus depends on how well this approximation holds. As such, the result is particularly applicable to networks with piece-wise linear activation functions such as the ReLU, which are locally linear and are among the most widely used activation functions. For these activation functions this bound is locally exact: $\mathcal{O}(\boldsymbol{\varepsilon})$ is exactly zero if the size of the bound is smaller than what is required to go outside the locally linear region.

This gives us margins for which VAEs are certifiably robust, up to first order expansions, to adversarial perturbations on their inputs; they have similar forms to the sensitivity margins for classifiers defined by Sokolić et al. (2017); Jakubovitz & Giryes (2018) in that both scale *inversely* with the network Jacobian. More generally, these results provide insights into the features which lead to robust VAEs. As shown in Figure 6.6, the bound seems to be relatively tight in practice, even when attacking both the encoder mean *and* variance. It also has a near-linear relationship with the empirically estimated robustness, such that it forms a powerful and convenient robustness metric in its own right.

Examining the bound, we see that for a given r, $R^r_{\mathcal{X}}(\mathbf{x})$ increases as the stochasticity of the encoder, i.e $\boldsymbol{\sigma}_\phi(\mathbf{x})$, increases, provided that this does not overly affect $\Phi^{-1}(p(||\Delta(\mathbf{x})||_2 \leq r))$ (see below). As $\boldsymbol{\sigma}_\phi(\mathbf{x})$ tends to 0 we recover the deterministic setting, which confers no additional protection to attack and as $\boldsymbol{\sigma}_\phi(\mathbf{x})$ increases we obtain increased protection. However, $\boldsymbol{\sigma}_\phi(\mathbf{x})$ can also have a knock–on effect on $\Phi^{-1}(p(||\Delta(\mathbf{x})||_2 \leq r))$. When $\boldsymbol{\sigma}_\phi(\mathbf{x})$ is small, this knock–on effect will typically be



(a) MNIST **(b)** fMNIST **(c)** CIFAR10

(d) MNIST **(e)** fMNIST **(f)** CIFAR10

Figure 6.6: (a-c) show the empirically estimated $R^r_{\mathcal{X}}(\mathbf{x})$ against the bound for $R^r_{\mathcal{X}}(\mathbf{x})$ defined in Theorem 1, ignoring higher order terms. Each dot represents a network–input pair, with 5 separately trained networks and 25 distinct inputs considered. We show the line of best fit (in orange), the correlation coefficient ρ, and the line $y = x$ (in red) representing the theoretical bound itself. (d-f) show the relative log likelihood degradation resulting from a 'maximum-damage' adversarial attack against the numerically estimated $R^r_{\mathcal{X}}(\mathbf{x})$ for these same VAEs and inputs (see Section 6.4.1).

small relative to the direct effect of changing $\boldsymbol{\sigma}_\phi(\mathbf{x})$, but as it becomes large there is always a point where this knock–on effect will take over. That is, our reconstructions will become increasingly poor and $\Phi^{-1}(p(||\Delta(\mathbf{x})||_2 \leq r))$ will eventually become negative, such that r-robustness does not hold even without perturbation. We can quantify this by noting that there is always a minimum r for r-robustness to be satisfied in (6.2). We derive a bound characterising the minimum r for which we can confirm robustness in Appendix E.1.

6.4 Empirical Investigations

We now consider a series of empirical investigations to back up our frameworks and theoretical results. We start by assessing whether the concept of r-robustness corresponds to more commonly used measures of model robustness. Here we estimate $R^r_{\mathcal{X}}(\mathbf{x})$ numerically as in Appendix E.4 and as demonstrated in Fig 6.5. Using

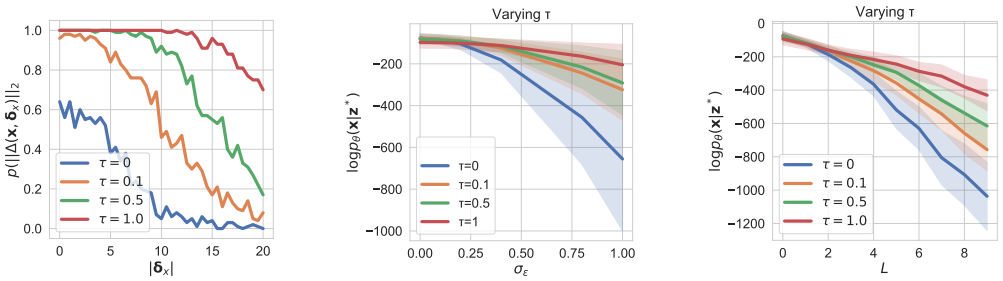


Figure 6.7: Ablation study on the bounds defined by Theorem 6.1. We train models on MNIST with $\sigma_\phi(\mathbf{x})$ offset by a constant $\tau \in [0, 0.1, 0.5, 1]$. [Left] probability that reconstructions in $\mathcal{X}_{\text{recon}}$ fall within a radius $r = 4$ centred on the 'maximum likelihood' reconstruction, $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r)$, as a function of $|\boldsymbol{\delta}_x|$, the magnitude of perturbations. $R^r_{\mathcal{X}}(\mathbf{x})$ is the radius $|\boldsymbol{\delta}_x|$ for which $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq 4) > 0.5$ and clearly increases with τ. [Centre] we add noise $\sim \mathcal{N}(0, \sigma^2_\epsilon)$ to a point $\mathbf{x}$ forming a noisy $\mathbf{x}^*$ and $\mathbf{z}^*$, and measure the likelihood of the original point $\mathbf{x}$ under this noisy embedding. [Right] we show the same plot where the perturbations are *maximum damage* attacks, Eq (6.5), where L is the maximum allowed magnitude of the attack distortion. Large τ VAEs have high likelihoods for the original point $\mathbf{x}$ as L and σ^2_ϵ increase: they are robust to attack and effective denoising models. Confidence intervals are the standard deviations of values over the entire MNIST dataset.

these empirical estimations, we establish connections between r-robustness and other performance metrics during adversarial attack, confirming that larger $R^r_{\mathcal{X}}(\mathbf{x})$ correspond to model–input pairs that are more robust to adversarial attacks.

6.4.1 r-robustness and Adversarial Settings

We begin by evaluating our metrics in adversarial settings. We want to find the most damaging perturbations $\boldsymbol{\delta}_x$ that challenge the robustness metrics we have derived. We consider an adversary trying to distort the input data to maximally disrupt a VAE's reconstruction. Our adversary maximises, w.r.t. $\boldsymbol{\delta}_x$, the distance between the VAE reconstruction and the original datapoint $\mathbf{x}$, a novel adversarial attack we call *maximum damage*. We attack the encoder mean *and* variance:

$$\boldsymbol{\delta}^*_x = \arg\max_{\boldsymbol{\delta}_x} \mathbb{E}_{\boldsymbol{\eta}\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \left[||g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x} + \boldsymbol{\delta}_x) + \boldsymbol{\eta}\sigma_\phi(\mathbf{x} + \boldsymbol{\delta}_x)) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))||_2 \right]. \quad (6.5)$$

Note that we estimate the attack objective, the internal part of the above equation, for a given value of $\boldsymbol{\delta}_x$ using a single sample of $\boldsymbol{\eta}$. However, we sample a new value of $\boldsymbol{\eta}$ for each new value of $\boldsymbol{\delta}_x$, for example as the adversary performs optimisation

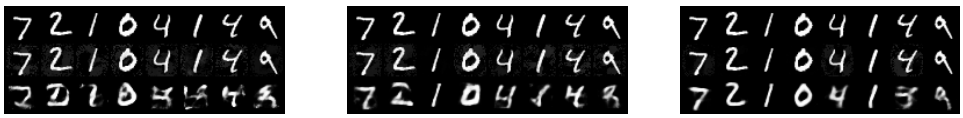
over $\boldsymbol{\delta}_x$ in performing this attack. We evaluate the success of an attack as follows. Given an embedding $\mathbf{z}^*$ formed from the mean encoding of $\mathbf{x} + \boldsymbol{\delta}_x$, we measure the likelihood of the original point $\mathbf{x}$ and quantify the degradation in model performance as the relative log likelihood degradation ($|(\log p(\mathbf{x}|\mathbf{z}^*) - \log p(\mathbf{x}|\mathbf{z}))|/\log p(\mathbf{x}|\mathbf{z})$), where $\mathbf{z}$ is the embedding of $\mathbf{x}$. Fig 6.6 (d-f) shows that as $R^r_{\mathcal{X}}(\mathbf{x})$ increases this degradation lessens, indicating less damaging attacks. As such, larger margins for r-robustness correspond to models that are more robust to attack.

6.4.2 Evaluating the derived bounds

Using Theorem 6.1, we can gain insights into which characteristics of a VAE contribute to robustness. The encoder variance plays a prominent role and is a parameter that is easy to control. The encoder Jacobian is also present, but we found that controlling such values directly can be difficult. Penalising the norm of this Jacobian in the VAE training objective degrades VAE generative performance, making it difficult to compare models. As such we restrict our experiments to varying the encoder variance. We do so by training models that have $\boldsymbol{\sigma}_\phi(\mathbf{x})$ offset by a constant τ, such that we artificially increase the encoder variance minimum. In Fig 6.7 we show that as τ increases, the numerically estimated $R^r_{\mathcal{X}}(\mathbf{x})$ also increases, supporting our claim that models with larger encoder variances have larger margins of robustness. This figure also shows that likelihood of reconstructing the original input x increases as τ increases in both an adversarial attack setting and a noisy perturbation setting. We thus see that larger τ also provides more effective denoising properties.

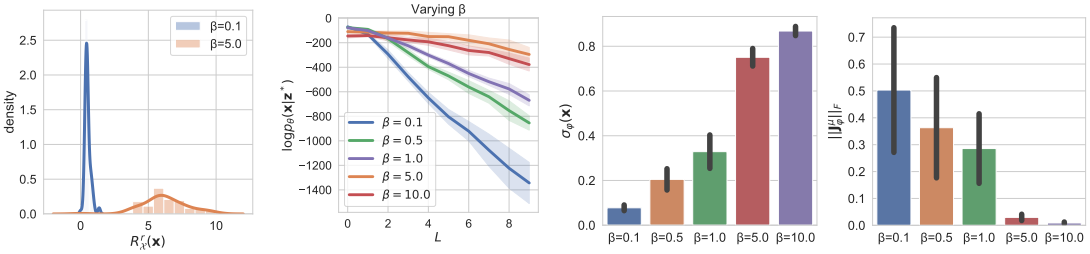
6.5 Robustness of Disentangled VAEs

We now apply our analysis to disentangling methods, which we empirically showed to be more robust to adversarial attacks and noisy data in the last chapter. First, we demonstrate this visually in Fig 6.8 where we see that β-VAEs are more resilient to attack as β increases, and thus implicitly latent space overlap increases (Mathieu



(a) $\beta = 0.1$ **(b)** $\beta = 1$ **(c)** $\beta = 10$

Figure 6.8: For β-VAEs trained with $\beta \in \{0.1, 1, 10\}$ we show in consecutive rows first the original data point, a perturbed version made by maximum damage adversarial attacks, and then the reconstruction given by the model. As β increases the models become more robust to attack.



(a) Sampled $R^r_{\mathcal{X}}(\mathbf{x})$ **(b)** Adv Attack **(c)** Encoder Variance **(d)** Encoder Jacobian

Figure 6.9: (a) distribution of the numerically estimated $R^r_{\mathcal{X}}(\mathbf{x})$ ($m = 0.5$) across the MNIST dataset. We see that $R^r_{\mathcal{X}}(\mathbf{x})$ increases dataset-wide for larger β. (b) likelihood of the original input given a maximum damage adversarial attack as in Eq (6.5). L is the maximum allowed norm of the attack. Large β models retain high likelihoods even for large L, meaning they are robust to attack. (c) and (d) show that the encoder variance increases and the encoder Jacobian norm ($||\mathbf{J}^\mu_\phi(\mathbf{x})||_F$) decreases as β increases, supporting our analysis that the changes in these values underpin the robustness observed. Confidence intervals for all plots are the standard deviation of values over the entire MNIST dataset. See Appendix E.5 for similar experiments on other datasets.

et al., 2019). Second, we provide analysis to show that disentangling methods induce models with smaller encoder Jacobian norms *and* larger posterior variances, implying that they have larger margins $R^r_{\mathcal{X}}(\mathbf{x})$ by Theorem 6.1, a result we confirm empirically.

Disentangling increases encoder variance Empirically, increasing $\beta > 1$ in a β-VAE increases the variance of the trained encoder, saturating at the variance of the prior for large β (Mathieu et al., 2019; Locatello et al., 2019). We can shed light on this behaviour by finding the optimum forms of the posterior distribution under these objective functions using calculus of variations. We find that the optimal posterior has the form of a tempered or fractional posterior (Holmes & Walker, 2017; Wenzel et al., 2020; Miller & Dunson, 2019), with an exponent $1/\beta$ on the likelihood:

Theorem 6.2. *For a β-VAE, the optimum posterior is:*

$$q_\phi(\mathbf{z}|\mathbf{x}) \propto p(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z})^{1/\beta}.$$

The proof is given in Appendix E.3.

This result gives the optimal posterior as a function of β. It also tells us the β-VAE's optimal posterior in the limit of large β is the prior, as we would expect. Because the prior variance is naturally larger than that of the encoder, the encoder variance increases with β.

Disentangling penalises Jacobian norm Assuming a Gaussian $p_\theta(\mathbf{x}|\mathbf{z})$, an encoder covariance optimal to first order, and activation functions that are piece-wise-linear, the β–VAE objective can be approximated as (Kumar & Poole, 2020)

$$\min_{\phi,\theta} \frac{1}{2}||\mathbf{x} - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))||^2 + \frac{\beta}{2}||\mathbf{J}^\mu_\phi(\mathbf{x})\mathbf{x}||^2_F + \frac{\beta}{2}\log|\mathbf{I} + \frac{1}{\beta}\mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))\mathbf{J}^T_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))|, \tag{6.6}$$

As β increases $||\mathbf{J}^\mu_\phi(\mathbf{x})||^2_F$ is more penalised and we expect to learn encoders with smaller Jacobians.

Taking these results together, we expect two things to occur as β increases: the encoder variance should increase by Theorem 6.2 and the norm of the encoder Jacobian should decrease. We confirm this empirically in Fig 6.9(c,d). By Theorem 6.1 these two effects of increasing β should increase $R^r_\mathcal{X}(\mathbf{x})$ in tandem. In Fig 6.9(a) we confirm that the numerical estimate for $R^r_\mathcal{X}(\mathbf{x})$ increases *dataset-wide* for large β, that is we get a larger value for $R^r_\mathcal{X}$, or metric for the overall robustness of the VAE. In Appendix E.5, we also show that the distribution of the *bound* for $R^r_\mathcal{X}(\mathbf{x})$ from Theorem 6.1 increases dataset-wide with increasing β. In both cases it is noticeable that $R^r_\mathcal{X}(\mathbf{x})$ is quite a well-behaved distribution, with reasonably low variance and skew. This suggests that $R^r_\mathcal{X}$ can be reliably estimated in practice.

In Fig 6.9b we further show that these larger $R^r_\mathcal{X}$ values translate into larger likelihoods of the original input under adversarial attack, while in Appendix E.5 we confirm that model sensitivity to noise is improved for larger β. We note, however,

that having a β that is too large will completely undermine reconstructions (Higgins et al., 2017a; Chen et al., 2018a) and lead to VAEs that are never robust because we cannot confirm r-robustness, even without input perturbations (Eq (6.2)). See Appendix E.4.1.1 for results on this.

6.6 Conclusion

We have defined a novel robustness metric tailored to probabilistic generative models, r-robustness, which can be used to assess the robustness of VAEs to adversarial attack. We defined a margin on a VAE's input space within which it is r-robust to perturbations and show that small norms of the encoder Jacobian and larger encoder variances are core contributors to robustness. Further, we offered theoretical and empirical analysis based on this margin, demonstrating that existing disentangling methods increase robustness by altering the optimal encoder variance and the norm of the encoder Jacobian.

We demand rigidly defined areas of doubt and uncertainty!

— Douglas Adams, *The Hitchhiker's Guide to the Galaxy*

7 Conclusion

In this thesis we have explored deep generative models from various standpoints, proposing a new learning regime, new approaches for hierarchical models with discrete variables, a new way to perform non-linear independent component analysis, and new study, both empirical and theoretical, of the robustness of deep generative models to adversarial attack.

In Chapter 2 we considered how to appropriately structure models so that they are robust to learning when present with extremely sparse labelling, to jointly learn to cluster and classify. This learning regime, semi-unsupervised learning, where for some classes only unlabelled examples are found in the training set, is potentially the case when practitioners might think they are performing vanilla semi-supervised learning. How can we get models that learn appropriate partitions of data when performing a mixture of clustering and semi-supervised learning? We showed how, for some standard machine learning datasets, deep generative models with Gaussian mixture priors can learn effectively in this regime, and can be structured to learn what is, in effect, a classifier as an inference artifact. This classifier, implemented as a feed-forward neural network, can be used like any other deep learning classifier on any new datapoints that an analyst may obtain.

In the next chapter, Chapter 3, we explored how hierarchical deep generative models with discrete latent variables could be improved and stabilised by a particular parameterisation of their latent variables. The building block for these is an extended form of vector quantisation, which we used for both the generative and posterior models. With this structure we were able to train very deep hierarchies of discrete latent variables on a range of image datasets. This approach narrows the performance gap, in terms of the value of the ELBO reached, between hierarchical discrete VAEs and VAEs with continuous variables. Recent advances have reestablished hierarchical continuous VAEs as the state-of-the-art for probabilistic modelling of high dimensional data like images (Vahdat & Kautz, 2020; Child, 2021), and we hope that this augurs well for future improvements and interest in hierarchical discrete models of similar form.

In machine learning, one of the most important areas of study has been Independent Component Analysis (ICA). The aim is to learn statistically independent latent representations of data. In the classic setting of this problem, blind source separation, the aim is to unmix audio recordings into the individual instruments or voices that were composed to form the recordings. However, non-linear unmixing for high-dimensional data, such as images, remains an active area of research. In Chapter 4 we propose a new approach to non-linear non-square (that is, compressive) ICA. We use a flow to learn a bijective feature map and then we place a linear ICA model over the flow representations. We learn both parts, the flow and the linear ICA model, jointly. Although we do learn the mixing matrix in the linear ICA part of the model, the unmixing matrix is fixed at initialisation, sampling a sketching matrix. In effect we are asking the highly expressive flow to produce representations for which our partially-fixed linear ICA model is highly explanatory. Informally, it is as if we are learning the data for a fixed model. We find that this approach learns the correct latent representations when trained on toy image datasets for which the true underlying independent factors are known a priori, unlike VAEs with ICA-appropriate priors. On more complicated image datasets, the numerical

value of total correlation of the aggregate posterior of the model is better than the values for VAEs when trained using ICA-appropriate priors.

In the final two chapters, 5 & 6, we took a deep dive into the robustness of VAEs to adversarial attack, a problem area not previously studied. In an attack on a VAE, the aim of the adversary is to fool the model into reconstructing a distorted input to a chosen target. We showed empirically in Chapter 5 that, by regularising the latent space and so controlling the noise of the amortised posteriors, we can reliably induce robust VAEs. We penalise the total correlation, a method proposed in Chen et al. (2018a) to obtain 'disentangled' representations, to obtain these robust representations. This method works by controlling the amount of *overlap* (Mathieu et al., 2019), so that there are not 'gaps' or 'holes' in the aggregate posterior that render attacks on VAEs easy. We showed that for a range of datasets and for a range of attack methods this penalisation confers robustness. It also confers this robustness reliably, unlike achieving disentangling which has been shown to be very tricky to obtain in practice using these methods (Rolinek et al., 2019; Locatello et al., 2019). Further, we show that hierarchical models with this kind of penalisation in their top-most latent variable are more robust still, being extremely challenging to attack for the datasets and attack modes studied. We also demonstrated that downstream classification tasks are more robust to attack when using TC-penalised VAEs (classifying either the reconstructions or the $\mathbf{z}$ representations) than when using vanilla VAEs, and our hierarchical TC-penalised VAEs are more robustness still.

However, while these approaches to protect against adversarial attack are well motivated by considerations of the geometry of the aggregate posterior and empirically work, there is still a question of what we truly mean when we talk about a VAE being robust to attack. In Chapter 6 we make the first inroads into this, introducing the concept of r-robustness. A VAE is r-robust to a perturbation $\boldsymbol{\delta}$ at a point $\mathbf{x}$ if the VAE's reconstruction under the distorted input (the maximum of the likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ given a sample from $q_\phi(\mathbf{z}|\mathbf{x}+\boldsymbol{\delta})$) will fall within an r-ball (under a chosen norm) of the undistorted reconstruction more than half the time. We then derived a margin $R^r_{\mathcal{X}}(\mathbf{x})$ on $\boldsymbol{\delta}_x$ such that for all distortion with $||\boldsymbol{\delta}_x|| < R^r_{\mathcal{X}}(\mathbf{x})$

a VAE is r-robust. The form of this margin depends on the minimum of the posterior's standard deviation and on the reciprocal of the Frobenius norm of the Jacobian of the mean of the posterior. Using this margin we can now understand formally how a simple method for robustness in VAEs, β-VAEs (Definition 4.1), obtain increased robustness. β-penalisation both naturally increases the minimum of the posterior standard deviation, Theorem 6.2, and decreases the Frobenius norm of the Jacobian of the mean.

Extensions: Each of the works presented in this thesis offers potential directions for future research. Semi-unsupervised learning invites further study into the inductive biases in algorithms for clustering and for semi-supervised learning. Already some of this work has been undertaken, that might mitigate this problem by having a model that provides multiple clusterings of the given data (Willetts et al., 2019; Li et al., 2019). Still there is much more to be done, not least as these new clustering methods are either quite heuristic driven or offer only small (or negative!) performance improvements. Theoretical work in this area would be of great impact – results for the identifiability of non-linear clustering models.

Learning hierarchical VAEs with discrete latent variables is also fertile soil for more research. Both the learning of discrete latent variables and the training of hierarchical VAEs are areas of active research, and more can be done to bring them together.

Why are discrete hierarchical VAEs not as high performance as their continuous counterparts? That they are of course intrinsically low capacity when compared to continuous models of the same form is part of the answer. While the Gumbel-Softmax trick is an extremely useful and important advance, enabling a reparameterisation trick for discrete variables, it is biased. This bias has been pointed to as part of the reason why hierarchical discrete VAEs do not reach the highest performance (Liévin et al., 2019). While extensions have been proposed to the Gumbel-Softmax trick, such as Grathwohl et al. (2018), that are unbiased, so far there have not been any proposed that work well in the presence of hierarchies of latent variables. Further, the latest and greatest continuous hierarchical VAEs (Vahdat & Kautz, 2020; Child,

2021) have various subtle tricks in their neural implementation and method of training that have not been brought to bear in the discrete case. Some of these tricks may not be needed, as many of them are to stabilise training which is not always a concern in the discrete case. Some others may not carry over for one reason or another, but perhaps some of them might help. It may also be that 'sister' tricks need to be discovered for discrete hierarchical VAEs for them to reach their potential.

The non-linear non-square approach to ICA we propose, Bijecta, offers a new approach to discovering independent latents that explain high-dimensional data like images. While the approach we propose performs well empirically, we have not put forward any 'identifiability' results, which are much-prized in the ICA community as they give confidence in the uniqueness of the set of learnt representations (up to trivial transformations like scaling and permutations). Recently there has been a renaissance in identifiability results in nonlinear ICA (Khemakhem et al., 2020a,b; Roeder et al., 2021; Gresele et al., 2020). While many of these rely on some form of auxiliary observations or side-information, which renders them perhaps inappropriate for the purely unsupervised setting, approaches inspired by these recent works might be fruitful in obtaining Bijecta-like identifiable models.

In VAE robustness, having demonstrated how regularising the latent space can lead to robust models, both empirically and theoretically, there are various plausible follow up works. Firstly, in Chapter 5 we focus mostly on TC-penalisation. Other methods of controlling overlap are discussed in Mathieu et al. (2019), and could be applied here. Secondly, the theoretical basis of the very strong robustness we obtain in hierarchical VAE is poorly understood. Perhaps VAEs with VLAE-like p and q factorisation (Zhao et al., 2017), for example, which straddles the divide between hierarchical and single-latent-layer VAEs, could provide a useful test environment for both empirical and theoretical study of this phenomena. In Chapter 6 the definition of r-robustness is entirely agnostic to the norm under which to consider the reconstructions of the VAE, however we study L_2 norms exclusively. An obvious extension would be to study r-robustness and associated $R^r_{\mathcal{X}}(\mathbf{x})$ margins in more advanced geometries. For example, pixels in images are often represented as a set

of 8-bit integers, so one way to extend these ideas would be to study the number of changed pixels in the output and the degree of their change, the L_1 norm. Or perhaps the L_2 distance between activations in some layer of a previously-trained deep classifier might provide a more perceptual measure of robustness.

In Chapter 5 we mention in passing that micro-level constraints on the neural networks inside the VAE could also provide robustness. We achieve smoothness in this work by considering statistical and information-theoretic quantities relating to the latent space, but enforced Lipschitz continuity of the encoder and decoder networks may achieve much the same thing. In Barrett et al. (2022) we have tugged at some of these ideas, but still there is much to be done.

Finally, and more speculatively, perhaps through advances in functional analysis of deep learning models that gives insights into various regularisation terms (Adler & Lunz, 2018; Camuto et al., 2020) there may be connections between the noise in a VAE's latent space and its underlying, hitherto unmeasured, Lipschitz constant.

POSTSCRIPT Since writing this thesis, I have had the pleasure to work on projects that build further on the ideas, models and methods presented here.

On the topic of the multiple ways that a given dataset might be clustered, with collaborators we have proposed a new way to do this using VLAE-like models (Zhao et al., 2017) with VaDE-like clustering (Jiang et al., 2017) within each layer of latents. We also correct some mistakes in the proofs in Jiang et al. (2017). The resulting work, Falck et al. (2021), leads to an effective model for simulataneous learning of multiple clusterings of train and test data.

Regarding non-linear ICA, as we say above it has been shown theoretically that suitable side-information can lead to identifiable latents, when the model has infinite capacity, infinite data is available, the true posterior is in the variational family and the global optimum is found (Khemakhem et al., 2020a). With my collaborator Brooks Paige we have recently found, however, that unsupervised clustering VAEs, which learn a clustering variable alongside the continuous latent, learn representations that are empirically identifiable—to the same degree as (or greater than) models that rely on side information (Willetts & Paige, 2021). We hope that this phenomena helps to stimulate further work on the identifiability of models.

Finally, with my collaborator Alexander Camuto we have found new theoretical results that use harmonic theory to understand VAEs (Camuto & Willetts, 2022). We view the latent space of a VAE as a Gauss Space, a kind of measure space. We show that the encoder's variance controls the spectral properties of the functions parameterised by both the encoder and decoder networks. This analysis also shows that the variance of the latent noise induces a soft Lipschitz constraint on the decoder.

All of these advance along the lines of equiry of this thesis, and they leave open, or indeed open up, many interesting research questions about the properties, theoretical and empirical, of deep generative models.

January 2022

Appendices

A

Appendix for Semi-Unsupervised Learning

A.1 Model Implementation and Data Preprocessing

The distributions $p_\theta(\mathbf{x}|\cdot)$, $q_\phi(y|\mathbf{x})$ and $q_\phi(\mathbf{z}|\mathbf{x}, y)$ have their parameters parameterised by neural networks. All networks are small MLPs. $q_\phi(\mathbf{z}|y, \mathbf{x})$ in the variational posterior outputs the mean and (log) diagonal covariance for a Gaussian distribution. Both in SSVAEs and GM-DGMs, the mean and variance networks have the same layers up to the second hidden layer, each having its own output layer. For GM-DGMs, $p_\theta(\mathbf{z}|y)$ is a look-up table: it maps a given setting of y to its $\boldsymbol{\mu}_\theta(y)$ and $\log \boldsymbol{\Sigma}_\theta(y)$. Between models, identical network architectures are used for components with the same purpose.

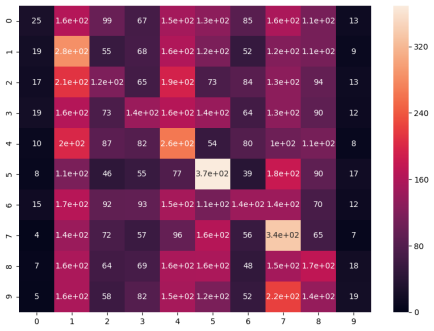
Table A.1: Per-dataset hyperparameters for networks in: $p_\theta(\mathbf{x}|\cdot)$, $q_\phi(y|\mathbf{x})$ and $q_\phi(\mathbf{z}|\mathbf{x}, y)$.

Dataset	Dim $\mathbf{z}$	Units	Batch Size	LR
MNIST	5	200	4	0.001
Fashion-MNIST	10	500	64	0.0015
HAR	15	500	64	0.005

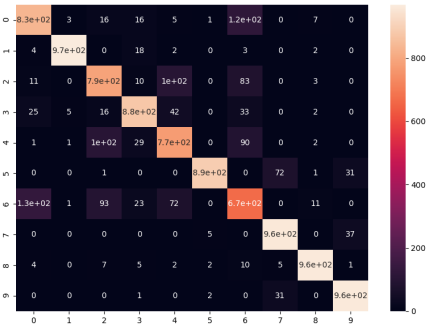
Kernel initialisation is from a Gaussian distribution with standard deviation of 0.001.

Biases are initialised with zeros. We perform stochastic gradient ascent to maximise the ELBO in each case. We use Adam (Kingma & Lei Ba, 2015), with default moment parameters. For the objectives of these models we must approximate various expectations wrt $q_\phi(\mathbf{z}|\mathbf{x}, y)$ and $q_\phi(y|\mathbf{x})$, which we do using the Reparameterisation Trick and Gumbel-Softmax Trick/Concrete Sampling respectively, taking 1 sample in each case per datapoint in the batch. We trained for up to 400 epochs, with cosine decay of the learning rate. For both MNIST and Fashion-MNIST we kept only dimensions of the data with a standard deviation greater than 0.1, and our likelihood function is a set of Bernoulli distributions. We binarise the input image data during training, taking the greyscale values as the probabilities of pixels being set to one, taking a draw to represent each image in a batch. For the Human Activity Recogntion dataset we study, HAR (Stisen et al., 2015), we used a Gaussian likelihood with fixed diagonal $\Sigma = \sigma^2\mathbb{I}$ with $\sigma = 0.01$.

A.2 Additional Confusion Matrices



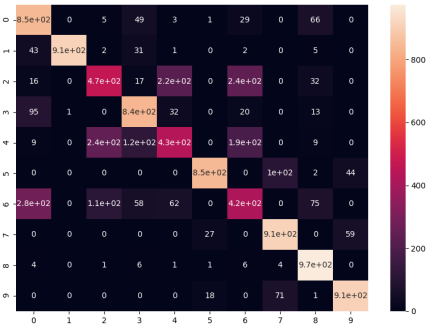
(a) SSVAE unsupervised on Fashion-MNIST



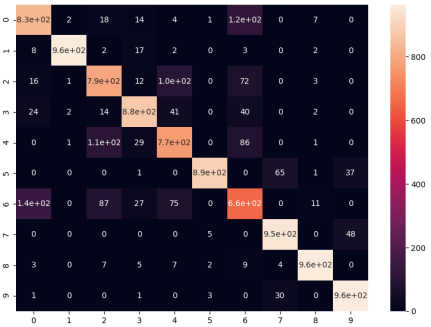
(b) SSVAE semi-supervised on Fashion-MNIST

(c) SSVAE semi-*un*supervised on Fashion-MNIST

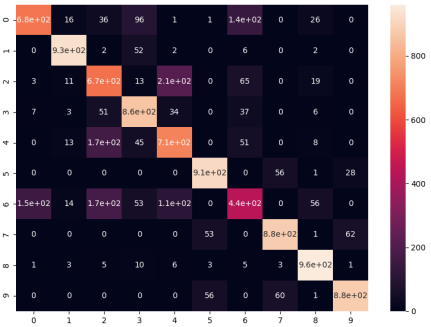
Figure A.1: Example confusion matrix from Fashion-MNIST test set for SSVAEs. a) clearly shows that this model struggles to learn to partition the data into clusters corresponding to the ground truth classes. b) reiterates that these models do perform vanilla semi-supervised learning well. c) shows how on the unsupervised subproblem within semi-unsupervised learning this model also struggles. Recall that for c) classes 5-9 were entirely unlabelled in the training set.



(a) GM-DGM unsupervised on Fashion-MNIST



(b) GM-DGM semi-supervised on Fashion-MNIST



(c) GM-DGM semi-*un*supervised on Fashion-MNIST

Figure A.2: Example confusion matrix for Fashion-MNIST test set from GM-DGMs. In all learning regimes, be it: a) unsupervised; b) semi-supervised; or c) semi-unsupervised this method is able to learn to separate the ground truth classes as well or better than SSVAEs. Recall that for c) classes 5-9 were entirely unlabelled in the training set.

B

Appendix for Relaxed-Responsibility Hierarchical Discrete VAEs

B.1 Relaxed Responsibility Vector Quantisation from a Mixture Model

To gain more insight into VQ-derived models, we can take a hierarchical discrete VAE and within it promote the embedding outputs $\vec{\mathbf{e}} = \{\mathbf{e}_1, .., \mathbf{e}_L\}$ to probabilistic variables. In doing this we obtain a hierarchical Gaussian mixture model, where each layer is itself a set of Gaussian mixture latent variables:

$$p_\theta(\mathbf{x}, \vec{\mathbf{z}}, \vec{\mathbf{e}}) = p_\theta(\mathbf{x}|\vec{\mathbf{e}})p_\theta(\vec{\mathbf{e}}, \vec{\mathbf{z}}) = p_\theta(\mathbf{x}|\vec{\mathbf{e}}) \prod_{\ell=1}^{L-1} [p(\mathbf{e}_\ell|\mathbf{z}_\ell)p_\theta(\mathbf{z}_\ell|\mathbf{e}_{>\ell})]p(\mathbf{z}_L) \tag{B.1}$$

where

$$p(\mathbf{e}_\ell|\mathbf{z}_\ell) = \prod_{m=1}^{M} \mathcal{N}(\mathbf{e}_\ell^m|\boldsymbol{\mu} = \mathbf{E}_{\mu,\ell}\mathbf{z}_\ell^m, \boldsymbol{\Sigma} = \mathbf{E}_{\Sigma,\ell}\mathbf{z}_\ell^m) \tag{B.2}$$

and

$$p_\theta(\mathbf{z}_\ell|\mathbf{e}_{>\ell}) = \prod_{m=1}^{M} \mathrm{Cat}(\mathbf{z}_m|\pi_\theta^m(\mathbf{e}_{>\ell})). \tag{B.3}$$

The posterior is given by

$$q_\phi(\vec{\mathbf{z}}, \vec{\mathbf{e}}|\mathbf{x}) = q_\phi(\mathbf{z}_L|\mathbf{x})q_\phi(\mathbf{e}_L|\mathbf{z}_L) \prod_{\ell=1}^{L-1} q_\phi(\mathbf{z}_\ell|\mathbf{e}_{>\ell}, \mathbf{x})q_\phi(\mathbf{e}_\ell|\mathbf{z}_\ell). \tag{B.4}$$

We can obtain our model as a restricted version of this. Our intent is to bottleneck our representations through a set of discrete latent variables. Thus we choose $q_\phi(\mathbf{e}_\ell|\mathbf{z}_\ell) = p_\theta(\mathbf{e}_\ell|\mathbf{z}_\ell) = \delta(\mathbf{e}_\ell - \mathbf{E}_{\mu,\ell}\mathbf{z}_\ell)$, where $\delta(\cdot)$ is the Dirac delta function. This gives us an ELBO of the form

$$\begin{aligned}\mathcal{L}(\mathbf{x}) = &\mathbb{E}_{\vec{\mathbf{z}}\sim q} \log p_\theta(\mathbf{x}|\vec{\mathbf{z}}) - \sum_{\ell=1}^{L-1} \mathbb{E}_{\mathbf{z}_{>\ell}\sim q}\, D_{\mathrm{KL}}(q_\phi(\mathbf{z}_\ell|\mathbf{z}_{>\ell}, \mathbf{x})||p_\theta(\mathbf{z}_\ell|\mathbf{z}_{>\ell})) \\ &- D_{\mathrm{KL}}(q_\phi(\mathbf{z}_L|\mathbf{x})||p(\mathbf{z}_L)),\end{aligned} \tag{B.5}$$

where we have changed the likelihood to depend on $\vec{\mathbf{z}}$, as the $\vec{\mathbf{e}}$ it depended on is now deterministic given $\vec{\mathbf{z}}$. This mirrors our original notation, where one writes $p_\theta(\mathbf{x}|\mathbf{z})$ and $\mathbf{z}$ implicitly looks-up the codebook embeddings inside the likelihood. If we then we choose Eqs (3.9, 3.12) to parameterise the inference and generative models of each $\mathbf{z}$, we thus obtain our RRVQ–VAE.

B.2 Interpreting Discrete Hierarchical VAEs as Learning a Series of Reconstructions

We can expand each D_{KL} in Eq (3.8) as a cross entropy and an entropy: $D_{\mathrm{KL}}(q||p) = \mathcal{H}(q||p) - \mathcal{H}(q)$. The ELBO for this model can then be written as

$$\begin{aligned}\mathcal{L}(\mathbf{x}) = &\mathbb{E}_{\vec{\mathbf{z}}\sim q}\left[\,\mathcal{H}(q(\mathbf{x})||p_\theta(\mathbf{x}|\vec{\mathbf{z}}))\right] \\ &- \sum_{\ell=1}^{L-1} \mathbb{E}_{\mathbf{z}_{>\ell}\sim q}\left[\,\mathcal{H}(q_\phi(\mathbf{z}_\ell|\mathbf{z}_{>\ell}, \mathbf{x})||p_\theta(\mathbf{z}_\ell|\mathbf{z}_{>\ell})) - \mathcal{H}(q_\phi(\mathbf{z}_\ell|\mathbf{z}_{>\ell}, \mathbf{x}))\right] \\ &- \mathcal{H}(q_\phi(\mathbf{z}_L|\mathbf{x})||p(\mathbf{z}_L)) + \mathcal{H}(q_\phi(\mathbf{z}_L|\mathbf{x})),\end{aligned} \tag{B.6}$$

where $q(\mathbf{x})$ is the per-datapoint empirical distribution (we view a datapoint as a set sub-pixels) of one-hot discrete distributions.

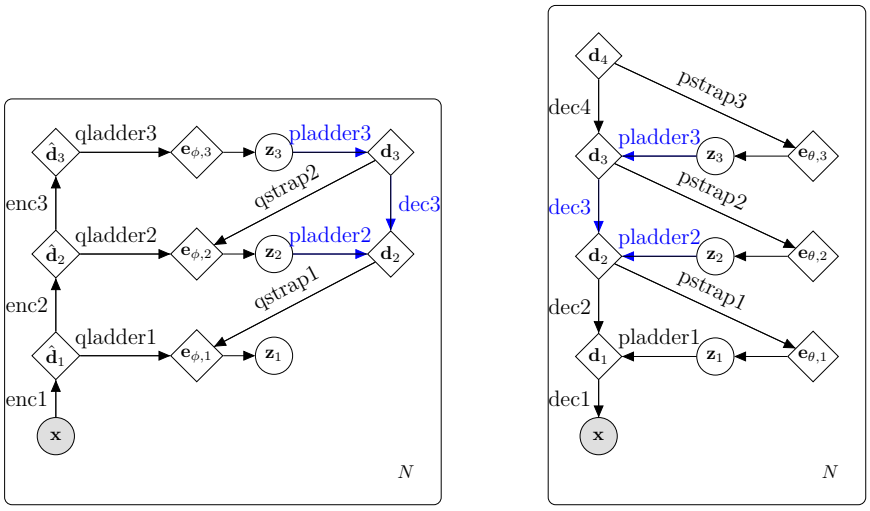
If the likelihood $p_\theta(\mathbf{x}|\vec{\mathbf{z}})$ is itself a set of discrete distributions, then in a hierarchical discrete VAE the latent layers and the likelihood term all provide to the ELBO cross-entropy terms between discrete distributions, with then the entropy of each latent posterior acting as regularisers. If that is that case, then during training we are, in effect, requiring our model to build a series of representations $\vec{\mathbf{z}}$, all

of which are scored under local objectives of the same form as how we score the reconstruction of our datapoint under our likelihood.

One might expect that the embedding of images when plotted as an image looks somewhat like a compressed version of the input data, up to the arbitrary indexing of the discrete latents. This is a weak effect, but can be seen somewhat in Fig B.1 below for two datapoints from CelebA. For each, the background is being encoded mostly using a single codebook index, which means that the person can be seen segmented out spatially in the latent representation in the first layer.

Figure B.1: For two input images from CelebA we plot them and their $\mathbf{z}_1$ representations from a RRVQ–VAE, colouring the indexes using the norm of the corresponding codebook mean.

B.3 Details of Model Architecture



(a) Variational Posterior **(b)** Gen. Model

Figure B.2: RRVQ–VAE with $L = 3$ as an example. (a) the variational posterior and (b) generative model, as defined in Eq (3.8). Blue arrows indicate shared networks. For simplicity the codebooks are not represented. Each labelled arrow corresponds to a network, described below.

The basic structure our network implementation is that of a ResNet VAE (Kingma et al., 2016). Now we describe the structure of each variety of network inside our model. enc1/dec1 are convolutions/transposed convolutions that down/upscale their inputs using a stride of 2. All the other subnetworks of the enc/dec deterministic backbones are each implemented as a single resnet block – each dec_ using a transposed convolution internally. When the mappings between two layers of latent variables requires a resizing, the identity path of the network performs a differentiable rescaling operation.

The networks qladder_ map from the backbone of encoders to the embedding space, and pladder_ map from the embedding space to the backbone of decoders. Each of these are each implemented as a single convolutional layer. The networks qstrap_ and pstrap_ too are each implemented as a single convolutional layer, and carry out upscaling using a stride of 2. They output in the embedding space. The embeddings used to define each layer's posterior distribution are the sum of the outputs of that layer's qstrap_ and qladder_ networks, and the embeddings used for the generative model's internal conditional probabilities are simply the outputs of each pstrap_.

For the $L = 32$ runs the backbones have 256 channels, and the $\mathbf{e}$ representations are 32 dimensional. Each layer's codebooks each hold 256 embeddings. The likelihood function is the same discretised logistic likelihood as in Kingma et al. (2016). As in Kingma et al. (2016), we use weight normalisation, ELU activations and free-bits regularisation.

The top-most latent variable in the generative model can be set to be uniform over embeddings, or can be parameterised by a similar procedure as for the rest via a $\mathbf{d}_L$ that is a learnable parameter (rather than itself the output of a network). See Fig B.2 for a representation of this – here for $L = 3$ we have $\mathbf{d}_4$ in the generative model parameterising $p_\theta(\mathbf{z}_3)$.

When training with just $L = 5$ layers, as opposed to 32, it is as if we remove the corresponding intermediate latent variables along with their ladder_, strap_ networks, so now the enc_ and dec_ networks are composed of 4 resnet blocks

between latents. We also promote the remaining ladder_, strap_ networks to themselves be composed of 4 resnet blocks.

We train using AdaMax with batch size 64 and an initial learning rate that we decay on plateau, multiplying by 0.8 when there has been no decrease in the test set ELBO for 20 (SVHN + CIFAR-10)/5 (CelebA) epochs, down to a minimum of 5×10^{-5}. The initial learning rate is 2×10^{-3}. We train with for up to 500 (SVHN + CIFAR-10)/160 (CelebA) epochs or until convergence. We used Azure VMs with NVIDIA M60 GPUs to train our models – using a single M60 to train a model takes ≈ 1 week for SVHN and CIFAR10. For the CelebA multi-GPU training is necessary.

B.4 Worst-Case Entropy of rVQ and Softmax-parameterised Discrete Distributions

B.4.1 Proof of Theorem 1

Proof. Our distribution of interest is a rVQ distribution, i.e. Eq (3.4), where we have the worst possible arrangement of our K-member codebooks – the arrangement that leads to the minimum possible entropy, and we also assume the worst possible positions of the embedding vector $\mathbf{e}$. The arrangement that leads to this is having all but one of the codebook vectors at one point and a single codebook separated a distance δ from them, with the embedding vector $\mathbf{e}$ lying along the line defined by those two positions a distance d from the outlier codebook vector and $d + \delta$ from the remaining $K - 1$ codebook vectors. We note that this arrangement is closely related to that considered in Beyer et al. (1999), § 3.5.2.

This gives us a distribution $p(\mathbf{z}|\boldsymbol{\pi})$, where

$$\pi^i = \begin{cases} \frac{1}{Z} \exp\left(-\frac{1}{2}d^2\right) & \text{if } i = 1 \\ \frac{1}{Z} \exp\left(-\frac{1}{2}(d+\delta)^2\right) & \text{otherwise.} \end{cases} \tag{B.7}$$

and

$$Z = \exp\left(-\frac{1}{2}d^2\right) + (K-1)\exp\left(-\frac{1}{2}(d+\delta)^2\right). \tag{B.8}$$

The entropy of this discrete distribution is thus:

$$\mathcal{H}_{\text{rVQ}} = -\sum_{i=1}^{K} \pi^i \log \pi^i \tag{B.9}$$

$$= -\frac{\exp\left(-\frac{1}{2}d^2\right)}{Z} \log\left(\frac{\exp\left(-\frac{1}{2}d^2\right)}{Z}\right) - (K-1)\frac{\exp\left(-\frac{1}{2}(d+\delta)^2\right)}{Z} \log\left(\frac{\exp\left(-\frac{1}{2}(d+\delta)^2\right)}{Z}\right) \tag{B.10}$$

$$= -\frac{\exp\left(-\frac{1}{2}d^2\right)}{Z}\left(-\frac{1}{2}d^2 - \log Z + (K-1)\exp\left(-\frac{1}{2}\delta^2 - \delta d\right)\left(-\frac{1}{2}\left(d+\delta\right)^2 - \log Z\right)\right). \tag{B.11}$$

Now let us consider the value of this in the limit of large d, $d \gg \delta$. First, let us expand $\frac{\exp\left(-\frac{1}{2}d^2\right)}{Z}$ using the first order expansion $(1+x)^{-1} \approx 1 - x$ for $|x| \ll 1$.

$$\frac{\exp\left(-\frac{1}{2}d^2\right)}{Z} = \frac{\exp\left(-\frac{1}{2}d^2\right)}{\exp\left(-\frac{1}{2}d^2\right) + (K-1)\exp\left(-\frac{1}{2}(d+\delta)^2\right)} \tag{B.12}$$

$$= \frac{1}{1 + (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)} \tag{B.13}$$

$$= 1 - (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right) + O\left(\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)^2\right). \tag{B.14}$$

Second, let us expand $\log Z$ using the first order expansion $\log(1+x) \approx x$ for $|x| \ll 1$.

$$\log Z = \log\left(\exp\left(-\frac{1}{2}d^2\right) + (K-1)\exp\left(-\frac{1}{2}(d+\delta)^2\right)\right) \tag{B.15}$$

$$= \log\left(\exp\left(-\frac{1}{2}d^2\right)\left(1 + (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)\right)\right) \tag{B.16}$$

$$= -\frac{1}{2}d^2 + \log\left(1 + (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)\right) \tag{B.17}$$

$$= -\frac{1}{2}d^2 + (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right) + O\left(\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)^2\right). \tag{B.18}$$

Taking Eqs (B.14,B.18) and subbing back into Eq (B.11), we get

$$\mathcal{H}_{\text{rVQ}} = \left[1 - (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)\right]\left[(K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)\left(1 + \frac{1}{2}(\delta + d)^2 - \frac{1}{2}d^2\right) + O\left(\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)^2\right)\right] \tag{B.19}$$

$$= (K-1)\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)\left(1 + \frac{1}{2}(\delta^2 + 2\delta d)\right) + O\left(\exp\left(-\frac{1}{2}(\delta^2 + 2\delta d)\right)^2\right). \tag{B.20}$$

Giving us, to first order in $\exp\left(-\frac{1}{2}(\delta^2+2\delta d)\right)$,

$$\mathcal{H}_{\text{rVQ}} \approx (K-1)\exp\left(-\frac{1}{2}(\delta^2+2\delta d)\right)\left(1+\frac{1}{2}\left(\delta^2+2\delta d\right)\right) \tag{B.21}$$

as required. □

B.4.2 Proof of Theorem 2

Proof. Our distribution of interest is a discrete distribution defined as a softmax of K raw logits, where we have the worst possible arrangement of the logit outputs – the arrangement that leads to the minimum possible entropy. The arrangement that leads to this is having all but one of the logits take one value c and a single logit taking the value $c+\ell$, $\ell > 0$.

This gives us a distribution $p(\mathbf{z}|\boldsymbol{\pi})$, where

$$\pi^i = \begin{cases} \frac{1}{Z}\exp(c+\ell) & \text{if } i=1 \\ \frac{1}{Z}\exp(c) & \text{otherwise} \end{cases} \tag{B.22}$$

and

$$Z = \exp(c+\ell) + (K-1)\exp(c). \tag{B.23}$$

The entropy of this discrete distribution is thus:

$$\mathcal{H}_{\text{softmax}} = -\sum_{i=1}^{K}\pi^i\log\pi^i \tag{B.24}$$

$$= -\frac{\exp(\ell+c)}{Z}(\ell+c-\log Z) - (K-1)\frac{\exp(c)}{Z}(c-\log Z). \tag{B.25}$$

Now let us consider the value of this in the limit of large ℓ, $\ell \gg c$. First, let us expand $\frac{1}{Z}$ using the first order expansion $(1+x)^{-1} \approx 1-x$ for $|x| \ll 1$.

$$\frac{1}{Z} = \frac{1}{\exp(c+\ell)+(K-1)\exp(c)} \tag{B.26}$$

$$= \frac{1}{\exp(\ell+c)}\frac{1}{1+(K-1)\exp(-\ell)} \tag{B.27}$$

$$= \exp(-\ell-c)\left(1-(K-1)\exp(-\ell)+O(\exp(-\ell)^2)\right). \tag{B.28}$$

Second, let us expand $\log Z$ using the first order expansion $\log(1+x) \approx x$ for $|x| \ll 1$.

$$\log Z = \log\left(\exp\left(c+\ell\right) + (K-1)\exp\left(c\right)\right) \tag{B.29}$$

$$= \log\left(\exp\left(c+\ell\right)(1 + (K-1)\exp\left(-\ell\right)\right) \tag{B.30}$$

$$= c + \ell + \log\left(1 + (K-1)\exp\left(-\ell\right)\right) \tag{B.31}$$

$$= c + \ell + (K-1)\exp\left(-\ell\right) + O\left(\exp\left(-\ell\right)^2\right). \tag{B.32}$$

Taking Eqs (B.28,B.32) and subbing back into Eq (B.25), keeping terms to first order in $\exp\left(-\ell\right)$ we get

$$\mathcal{H}_{\text{softmax}} \approx (K-1)\exp\left(-\ell\right)(1+\ell) \tag{B.33}$$

as required.

□

B.4.3 Experimental Evaluation

In order to empirically verify the bounds above, we compare the exact entropy to these first-order approximations for both methods' worst-case scenarios. We find the approximation to be highly accurate for inputs > 10, with proportional error $\approx 10^{-6}$ for each.

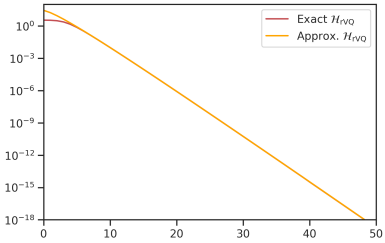


Figure B.3: rVQ worst-case entropy as a function of d, calculated exactly and using Eq (B.21), for $\delta = 1$. Note this is a logarithmic plot.

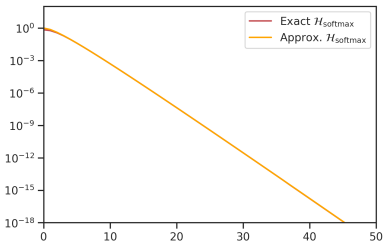


Figure B.4: Softmax worst-case entropy as a function of d, calculated exactly and using Eq (B.33), for $c = 0$. Note this is a logarithmic plot.

As an additional check on the rVQ results, we create random codebooks of embeddings uniformly distributed over the hypersphere with radius 0.5 and calculate $\mathcal{H}$ as a function of d. We do this for 20,000 sampled codebooks per value of d, each of 256 entries, in an embedding space with $d_e = 32$. The entropy we get from simulation shows an entirely different trend from the 'worst-case' calculations. This is reasonable as the worst-possible arrangement is very unlikely to occur.

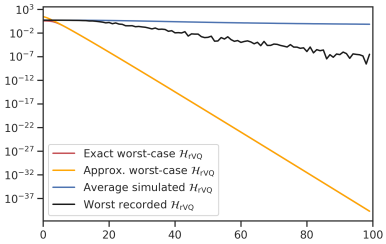


Figure B.5: rVQ entropy as a function of d, calculated for the worst case both exactly and using Eq (B.21), for $\delta = 1$, along with the average entropy from simulated codebooks with codebook embeddings uniform over the radius 0.5 hypersphere and the worst recorded entropy from that simulation procedure at each distance. Note this is a logarithmic plot.

Figure B.6: Reconstructions: We demonstrate our approach provides high quality reconstructions, for CIFAR-10, SVHN and CelebA. In each pair, left is the reconstruction, right the original.

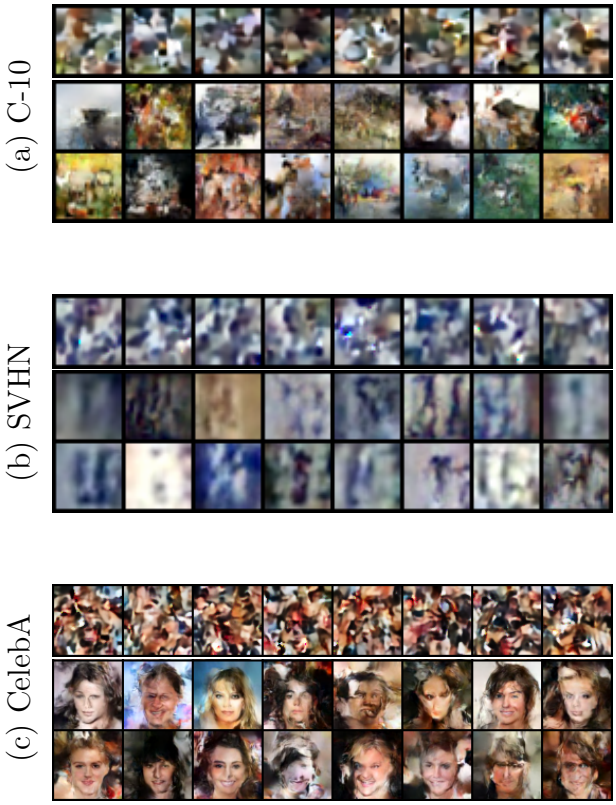


Figure B.7: Sampling: we perform ancestral sampling for single-layer rVQ–VAE baselines (top row) and our $L = 32$ models (middle and bottom), for CIFAR-10, SVHN and CelebA.

B.5 Compression using RRVQ models

For our $L = 5$ models, our latents $\vec{\mathbf{z}}$ are in 5 layers of size $\mathbf{M} = \{16 \times 16, 8 \times 8, 4 \times 4, 2 \times 2, 1 \times 1\}$. $\{\mathbf{E}_{\mu,\ell}, \mathbf{E}_{\Sigma,\ell}\}$. For CIFAR-10 and SVHN these each containing $K = 256$ codebook values $\in \mathbb{R}^{d_e}$, $d_e = 128$, per layer. For CelebA, we taper the number of embeddings per layer so $\mathbf{K} = \{128, 64, 32, 16, 8\}$, $d_e = 32$, and have networks layer-to-layer with fewer channels, for reasons of compute capacity.

In Fig B.8 we compress (top) CelebA images using (middle) our $L = 5$ model and (bottom) using JPEG to the same compression ratio (CR) [same experimental protocol as Gregor et al. (2016). We are compressing 64×64 images into 2275 bits, a CR of $\frac{98304}{2275} \approx 43$. Our approach outperforms JPEG, maintaining more visual information. Unlike JPEG, ours does not introduce blocky artefacts.

Figure B.8: *Top*: Original image, *Middle*: RRVQ $L = 5$ compression, *Bottom*: JPEG at same compression ratio. Best viewed zoomed in.

B.6 MLP rVQ-VAEs

For completeness, in Fig B.9 we train an MLP rVQ-VAE on our colour swatch data, to demonstrate that samples from such a model show consistent colour cast (further, samples show new colours beyond the training set). That is, ancestral samples look like the training data (i.e. with consistent colour) unlike single-latent-layer convolutional models.

Figure B.9: MLP-rVQ-VAE samples, trained on toy colour-swatch dateset.

C

Appendix for Learning Bijective Feature Maps for Linear ICA

C.1 Correlated and Dependent Sources

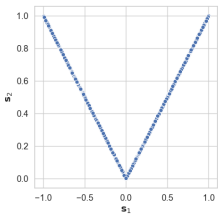


Figure C.1: Sources can be uncorrelated *and* dependent. Consider our first source $\mathbf{s}_1$ to be uniformly distributed on the interval $[-1, 1]$. If $\mathbf{s}_1 \leq 0$, then $\mathbf{s}_2 = -\mathbf{s}_1$, else $\mathbf{s}_2 = \mathbf{s}_1$. In this case the variables are uncorrelated, $\mathbb{E}[\mathbf{s}_1\mathbf{s}_2] = 0$, but the joint distribution of $\mathbf{s}_1$ and $\mathbf{s}_2$ is not uniform on the rectangle $[-1, 1] \times [0, 1]$, as it would be if they were independent. See plot to the left for an illustration of this.

C.2 Proof of optimality

Definition: We say a matrix $\mathbf{B}'$ *is strictly more orthogonal than a matrix* $\mathbf{B}$ *if* $\|\mathbf{B'}^T\mathbf{B'} - \mathbf{I}\| < \|\mathbf{B}^T\mathbf{B} - \mathbf{I}\|$.

Theorem C.1. *As the Frobenius norm* $\|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0$*, where* $\mathbf{G} \in \mathbb{R}^{r\times c}$ *and* $\tilde{\mathbf{G}}$ *is the projection of* $\mathbf{G}$ *onto* $\mathcal{V}(r, c)$ *,* $\|\mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T - \mathbf{D}\| \to 0$*, where* $\mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T$ *is the cross-correlation of the projection of data* $\mathbf{X}$ *by* $\mathbf{G}$*, and* $\mathbf{D}$ *is some diagonal matrix.*

Proof. The Stiefel manifold is given by (assuming $r < c$):

$$\mathcal{V}(r, c) = \{\mathbf{G} \in \mathbb{R}^{r\times c} : \mathbf{G}\mathbf{G}^T = \mathbf{I}\} \tag{C.1}$$

The unique projection $\tilde{\mathbf{G}}$ onto this manifold of a matrix $\mathbf{G} \in \mathbb{R}^{d_s\times d_x}$, with polar decomposition $\mathbf{UP}$, is simply $\mathbf{U}$ (Absil & Malick, 2012). $\mathbf{P}$ denotes a $c \times c$ positive-semidefinite Hermitian matrix and $\mathbf{U}$ is a $r \times c$ orthogonal matrix, i.e $\mathbf{U} \in \mathcal{V}(r, c)$ and has a conjugate transpose denoted $\mathbf{U}^*$ such that $\mathbf{U}\mathbf{U}^* = \mathbf{I}$. As such given any matrix $\mathbf{G}$ we have a polar decomposition $\mathbf{G} = \tilde{\mathbf{G}}\mathbf{P}$, where $\tilde{\mathbf{G}}$, a linear decorrelating matrix, is the projection onto $\mathcal{V}(r, c)$ and $\mathbf{P}$ denotes a $r \times c$ positive-semidefinite Hermitian matrix.

Let $\mathbf{G} \in \mathbb{R}^{r\times c}$ be some projection matrix of data $\mathbf{X}$. We have $\mathbf{S} = \mathbf{G}\mathbf{X}$. The cross-correlation is expressed as $\mathbf{S}\mathbf{S}^\mathbf{T} = \mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T$. In the case where $\mathbf{S}$ is perfectly decorrelated, we have: $\mathbf{S}\mathbf{S}^\mathbf{T} = \mathbf{D}^2$ where $\mathbf{D}$ is a diagonal matrix. We know that the Stiefel manifold $\mathcal{V}(r, c)$ (defined in Eq (C.1)) holds the set of all whitening matrices (Everson & Roberts, 1999), up to the diagonal matrix $\mathbf{D}^{-1}$.

For any matrix $\mathbf{G}$ and its projection $\tilde{\mathbf{G}}$ onto $\mathcal{V}(r, c)$, we have $\tilde{\mathbf{G}}\mathbf{X}\mathbf{X}^T\tilde{\mathbf{G}}^* = \mathbf{D}^2$, where $\tilde{\mathbf{G}}^*$ is the complex conjugate of $\tilde{\mathbf{G}}$. Consequently, given that the Frobenius norm is unitary invariant and the fact that $\tilde{\mathbf{G}}$ is unitary:

$$\begin{aligned}\|\mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T - \mathbf{D}^2\| &= \|\tilde{\mathbf{G}}\mathbf{P}\mathbf{X}\mathbf{X}^T\mathbf{P}^\mathbf{T}\tilde{\mathbf{G}}^* - \mathbf{D}^2\| \\ &= \|\tilde{\mathbf{G}}\mathbf{P}\mathbf{X}\mathbf{X}^T\mathbf{P}^T\tilde{\mathbf{G}}^* - \tilde{\mathbf{G}}\mathbf{X}\mathbf{X}^T\tilde{\mathbf{G}}^*\| \\ &= \|\mathbf{P}\mathbf{X}\mathbf{X}^T\mathbf{P}^T - \mathbf{X}\mathbf{X}^T\| \\ &\leq \|\mathbf{P}^2 - \mathbf{I}\|\ \|\mathbf{X}\mathbf{X}^T\|\end{aligned}$$

The last line comes from the triangle inequality and that $\mathbf{P}$ is Hermitian so $\mathbf{P}\mathbf{P}^T = \mathbf{P}^2$. As $\|\mathbf{X}\mathbf{X}^T\|$ is a constant, $\|\mathbf{P}^2 - \mathbf{I}\|\ \|\mathbf{X}\mathbf{X}^T\| \to 0$ as $\|\mathbf{P}^2 - \mathbf{I}\| \to 0$.

Proposition C.1. *Let* $\mathbf{G} \in \mathbb{R}^{r\times c}$ *and* $\tilde{\mathbf{G}}$ *is the projection of* $\mathbf{G}$ *onto* $\mathcal{V}(r, c)$*. Further let* $\mathbf{G} = \tilde{\mathbf{G}}\mathbf{P}$*. Then* $\|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0 \Leftrightarrow \|\mathbf{P}^2 - \mathbf{I}\| \to 0$*.*

Proof. As the Frobenius norm is invariant to unitary transformations, and because $\tilde{\mathbf{G}}$ and $\mathbf{U}$ are unitary matrices, the norm between $\tilde{\mathbf{G}}$ and $\mathbf{G}$ is

$$\begin{aligned}\|\mathbf{G} - \tilde{\mathbf{G}}\| &= \|\mathbf{G}\tilde{\mathbf{G}}^* - \mathbf{I}\| \\ &= \|\mathbf{U}\mathbf{P}\mathbf{U}^* - \mathbf{I}\| \\ &= \|\mathbf{U}(\mathbf{P} - \mathbf{I})\mathbf{U}^*\| \\ &= \|\mathbf{P} - \mathbf{I}\|.\end{aligned}$$

$\mathbf{P}$ is positive-semidefinite and hence $\|\mathbf{P}^2 - \mathbf{I}\| \to 0$ implies $\|\mathbf{P} - \mathbf{I}\| \to 0$ and the distance between $\mathbf{G}$ and $\mathcal{V}(|\mathbf{s}|, |\mathbf{z}|)$ strictly decreases, i.e $\|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0$. Generally:

$$\|\mathbf{P}^2 - \mathbf{I}\| \to 0 \Leftrightarrow \|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0 \tag{C.2}$$

This ends the proof. □

Thus by Proposition C.1 we have that:

$$\|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0 \Leftrightarrow \|\mathbf{G}\mathbf{X}\mathbf{X}^T\mathbf{G}^T - \mathbf{D}\| \to 0. \tag{C.3}$$

This ends the proof.

□

C.3 Proof of closeness

Theorem C.2. *Let* $\mathbf{G} \in \mathbb{R}^{d_s,D}$ *and let* $\tilde{\mathbf{G}}$ *be its projection onto* $\mathcal{V}(d_s, d_x)$*. As the Frobenius norm* $\|\mathbf{G}\mathbf{G}^T - \mathbf{I}\| \to 0$*, we also have* $\|\tilde{\mathbf{G}} - \mathbf{G}\| \to 0$*.*

Proof. Let $\mathbf{G} \in \mathbb{R}^{d_s,D}$ and let $\tilde{\mathbf{G}}$ be its projection onto $\mathcal{V}(d_s, d_x)$. By Proposition C.1 in Appendix C.2 we have that

$$\|\mathbf{P}^2 - \mathbf{I}\| \to 0 \Leftrightarrow \|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0$$

$\mathbf{G} \in \mathbb{R}^{d_s \times d_x}$, has polar decomposition $\mathbf{UP}$. Note that $\mathbf{G}\mathbf{G}^* = \mathbf{U}\mathbf{P}\mathbf{P}^T\mathbf{U}^* = \mathbf{U}\mathbf{P}^2\mathbf{U}^*$ because $\mathbf{P}$ is Hermitian. Recall that the Frobenius norm is invariant to unitary transformations, such as $\mathbf{U}$:

$$\begin{aligned}\|\mathbf{G}\mathbf{G}^* - \mathbf{I}\| &= \|\mathbf{U}\mathbf{P}^2\mathbf{U}^* - \mathbf{I}\| \\ &= \|\mathbf{P}^2 - \mathbf{I}\|.\end{aligned}$$

As such:

$$\|\mathbf{G}\mathbf{G}^* - \mathbf{I}\| \to 0 \Leftrightarrow \|\mathbf{G} - \tilde{\mathbf{G}}\| \to 0 \tag{C.4}$$

Note that we can trivially show this for $\mathbf{G}^*\mathbf{G} = \mathbf{P}^2$.

This ends the proof. □

C.4 Coupling Layers in Flows

Coupling layers in flows are designed to produce lower triangular Jacobians for ease of calculation of determinants. Rational Quadratic Spline (RQS) flows are defined by K monotonically increasing *knots*, which are the coordinate pairs through which the function passes: $\{(x_k, y_k)\}_{k=0}^{K}$. We can interpolate values between each of the K knots using the equation for the RQS transformation (Durkan et al., 2019). The resulting function is a highly flexible non-linear transformation, such that RQS flows require fewer composed mappings to achieve good performance relative to other coupling layers. The knots themselves are trainable, and parameterised by deep

neural networks. These can then be composed with other tractable transformations, including permutations and multiplication by triangular matrices. More specifically these layers can be defined as (Durkan et al., 2019):

1. Given an input $\mathbf{x}$, split $\mathbf{x}$ into two parts $\mathbf{x}_{1:d-1}$ and $\mathbf{x}_{d:D}$;
2. Using a neural network, compute parameters for a bijective function f using one half of $\mathbf{x}$: $\theta_{d:D} = \mathrm{NN}(\mathbf{x}_{1:d-1})$; parameters $\theta_{1:d-1}$ are learnable parameters that do *not* depend on the input;
3. The output $\mathbf{y}$ of the layer is then $y_i = f_{\theta_i}(x_i)$ for $i = 1, ..., d_x$.

These coupling transforms thus act elementwise on their inputs.

C.5 Network Architectures and Hyperparameters

C.5.1 RQS flows and Bijecta

Within all Rational Quadratic Spline (RQS) flows we parameterise 4 knots for each spline transform. The hyper-parameters of the knots were as in the reference implementation from Durkan et al. (2019), available at github.com/bayesiains/nsf: we set the minimum parameterised gradient of each knot to be 0.001, the minimum bin width between each encoded knot and the origin to be 0.001, and the minimum height between each knot and the origin to be 0.001.

Unlike in Durkan et al. (2019), where they view a single RQS 'layer' as composed of a sequence of numerous coupling layers, in this paper the number of layers we describe a model as having is exactly the number of coupling layers present. So for our 4-layer models there are four rational quadratic splines. Each layer in our flows are composed of: an actnorm layer, an invertible 1x1 convolution, an RQS coupling transform and a final 1x1 invertible convolution.

The parameters of the knots were themselves parameterised using ResNets nets, as used in RealNVP (Dinh et al., 2017), for each of which we used 3 residual blocks and batch normalisation. As in Dinh et al. (2017) we factor-out after each layer.

All training was done using ADAM (Kingma & Lei Ba, 2015), with default β_1, β_2, a learning rate of 0.0005 and a batch size of 512. We perform cosine decay on the learning rate during training, training for 25,000 steps.

Data was rescaled to 5-bit integers and we used RealNVP affine pre-processing so our input data was in the range $[\epsilon, 1-\epsilon]$ with $\epsilon = 0.05$.

C.5.2 VAEs

Affine For the affine experiments we used fully convolutional encoders and decoders, each made out of 5 resnet block. Each block also down/up-scales their input by a factor of 2 along the spatial input dimensions. As our input images are 32×32, this means that 5 such scalings map to 1×1 representations.

As we pass through the encoder we double the number of features for each block, from an initial number of 16. Thus the final residual output has $32 \times 16 = 512$ filters. A 1×1 convolutional layer then maps this to the posterior. The decoder performs these same operations in reverse order: an initial 1×1 convolutional layer maps the sampled value of the latent variable into a $1 \times 1 \times 512$ hidden representation, a chain of 5 upscaling resnet blocks map this to a $32 \times 32 \times 16$ representation. Finally a 1×1 convolution maps this to the $32 \times 32 \times 3$ sub-pixel means of the likelihood function.

CelebA For CelebA we used the same networks as used by Chen et al. (2018a) for their CelebA experiments. The encoder and decoder are composed for 5 convolutions/transposed convolutions, with batchnorm layers in between. The number of filters increases as we go up the encoder: $64 \to 64 \to 128 \to 128 \to 512$ before being mapped to the posterior distribution's parameters by a 1×1 convolution. The decoder has the same sequence of filter sizes in its transposed convolutions but in reverse, with a final convolution to the $32 \times 32 \times 3$ sub-pixel means of the likelihood function.

C.6 Reconstructions, Latent Traversals, and Samples

C.6.1 Numerical Results

Table C.1: Here we evaluate the source-separation and reconstruction quality of non-square ICA models. We evaluate source separation by evaluating the mean log probability of the validation set embeddings in $\mathcal{S}$ under our heavy-tailed prior, normalised by the dimensionality of $\mathcal{S}$ space: $\log p(\mathbf{s})/d_s$ ($d_s = 64$). As our base distribution is heavy-tailed, this metric evaluates the axis-alignment, the *independence*, of learnt factors. We also give the validation set bits-per-dim (bpd), a scaled ELBO and thus a measure of model quality (lower better). Bijecta models consistently have lower bpd than the linear models. We evaluate the quality of low-dimensional representations by measuring the L_1 reconstruction error in $\mathcal{X}$. A better representation encodes more information in $\mathcal{S}$, making it easier for the model to then reconstruct in $\mathcal{Z}$ and subsequently in $\mathcal{X}$. Most striking is the improvement across all metrics when introducing a *single* bijective mapping. Our 4-layer model further improves the quality of the compressed representations as seen by the lower reconstruction errors.

		CIFAR-10	MNIST	fashion-MNIST	CelebA
Linear-ICA	$\log p(\mathbf{s})/d_s$	-4.8	-4.29	-3.0	-9.0
	bits-per-dim	6.3	8.6	6.6	6.2
	L_1 reconstruction error in $\mathcal{X}$	3.0	3.1	2.9	2.9
1-layer Bijecta	$\log p(\mathbf{s})/d_s$	**-4.2**	**-2.8**	**-2.2**	**-7.3**
	bits-per-dim	3.8	3.2	3.8	3.4
	L_1 reconstruction error in $\mathcal{X}$	2.0	1.0	1.6	1.9
4-layer Bijecta	$\log p(\mathbf{s})/d_s$	-4.7	-3.0	**-2.2**	-7.8
	bits-per-dim	**3.2**	**2.1**	**3.1**	**3.2**
	L_1 reconstruction error in $\mathcal{X}$	**1.9**	**0.6**	**1.2**	**1.4**

D

Appendix to Improving the Robustness of VAEs to Adversarial Attack

Contents

D.1 Total-Correlation Decomposition of ELBO

Proof of Theorem 5.1

Here we prove that the ELBO for a hierarchical VAE with forward model as in Eq (5.3) and amortised variational posterior as in Eq (5.4) can be decomposed to reveal a total-correlation in the top-most latent variable.

Specifically, now considering the ELBO for the whole dataset and using $q(\mathbf{x})$ to indicate the empirical data distribution, we will obtain, denoting $\mathbf{z}^0 = \mathbf{x}$:

$$\begin{aligned}\mathcal{L}\left(\theta, \phi; \mathcal{D}\right) = &\mathbb{E}_{q_\phi(\vec{\mathbf{z}}, \mathbf{x})}\left[\log p_\theta(\mathbf{x}|\vec{\mathbf{z}})\right] - \mathbb{E}_{q_\phi(\vec{\mathbf{z}}|\mathbf{x})q(\mathbf{x})}\left[\sum_{i=1}^{L-1} D_{\mathrm{KL}}(q_\phi(\mathbf{z}^i|\mathbf{z}^{i-1}, \mathbf{x})||p_\theta(\mathbf{z}^i|\mathbf{z}^{i+1}))\right] \\ &- \mathbb{E}_{q_\phi(\mathbf{z}^{L-1})} D_{\mathrm{KL}}(q_\phi(\mathbf{z}^L, \mathbf{x}|\mathbf{z}^{L-1})||q_\phi(\mathbf{z}^L)q(\mathbf{x})) \\ &- \textstyle\sum_j D_{\mathrm{KL}}(q_\phi(\mathbf{z}_j^L)||p(\mathbf{z}_j^L)) - \beta D_{\mathrm{KL}}\Big(q_\phi(\mathbf{z}^L)||\prod_j q_\phi(\mathbf{z}_j^L)\Big) \end{aligned} \tag{D.1}$$

We start with the forms of p and q given in Theorem 5.1. The likelihood is conditioned on all $\mathbf{z}$ layers: $p_\theta(\mathbf{x}|\vec{\mathbf{z}})$.

$$\mathcal{L}(\theta, \phi; \mathcal{D}) = \mathbb{E}_{q_\phi(\vec{\mathbf{z}}, \mathbf{x})} \log \frac{p_\theta(\mathbf{x}, \vec{\mathbf{z}})}{q_\phi(\vec{\mathbf{z}}, \mathbf{x})} \tag{D.2}$$

$$= \mathbb{E}_{q_\phi(\vec{\mathbf{z}}, \mathbf{x})}\left[\log p_\theta(\mathbf{x}|\vec{\mathbf{z}})\right] - \mathbb{E}_{q(\mathbf{x})}\left[D_{\mathrm{KL}}(q_\phi(\vec{\mathbf{z}}, \mathbf{x})||p_\theta(\vec{\mathbf{z}}))\right] \tag{D.3}$$

$$= \mathbb{E}_{q(\vec{\mathbf{z}}, \mathbf{x})} \log p_\theta(\mathbf{x}|\vec{\mathbf{z}}) - \mathbb{E}_{q(\mathbf{x})} \log q(\mathbf{x}) + \mathbb{E}_{q(\vec{\mathbf{z}}, \mathbf{x})} \log \frac{p_\theta(\vec{\mathbf{z}})}{q(\vec{\mathbf{z}}|\mathbf{x})} \tag{D.4}$$

$$= \mathbb{E}_{q(\vec{\mathbf{z}}, \mathbf{x})} \log p_\theta(\mathbf{x}|\vec{\mathbf{z}}) + \mathcal{H}(q(\mathbf{x})) \tag{D.5}$$

$$+ \underbrace{\int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}^1 \prod_{i=2}^{L} (\mathrm{d}\mathbf{z}^i q_\phi(\mathbf{z}^i|\mathbf{z}^{i-1}, \mathbf{x})) q_\phi(\mathbf{z}^1|\mathbf{x}) q(\mathbf{x}) \log \frac{p(\mathbf{z}^L) \prod_{k=1}^{L-1} p_\theta(\mathbf{z}^k|\mathbf{z}^{k+1})}{q_\phi(\mathbf{z}^1|\mathbf{x}) \prod_{m=1}^{L-1} q_\phi(\mathbf{z}^{m+1}|\mathbf{z}^m, \mathbf{x})}}_{\text{(W)}}$$

So here we have three terms: an expectation over the data likelihood, the entropy of the empirical data distribution (a constant) and (W). We now can expand (W) to a term involving the prior for the latent $\mathbf{z}^L$ and a term involving the conditional distributions from the generative model for the remaining components of $\vec{\mathbf{z}}$:

$$\begin{aligned}\text{(W)} = &\underbrace{\int \mathrm{d}\mathbf{x} \prod_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\vec{\mathbf{z}}|\mathbf{x}) q(\mathbf{x}) \log \frac{\prod_{k=1}^{L-1} p_\theta(\mathbf{z}^k|\mathbf{z}^{k+1})}{q_\phi(\mathbf{z}^1|\mathbf{x}) \prod_{m=1}^{L-2} q_\phi(\mathbf{z}^{m+1}|\mathbf{z}^m, \mathbf{x})}}_{\text{(R)}} \\ &+ \underbrace{\int \mathrm{d}\mathbf{x} \prod_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\vec{\mathbf{z}}|\mathbf{x}) q(\mathbf{x}) \log \frac{p(\mathbf{z}^L)}{q_\phi(\mathbf{z}^L|\mathbf{z}^{L-1}, \mathbf{x})}}_{\text{(S)}}\end{aligned} \tag{D.6}$$

The first part (R), it that part of (W) not involving the prior for 'top-most' latent variable $\mathbf{z}^L$, is the first subject of our attention. We split out the part of (R) involving the generative and posterior terms for the latent variable closest to

the data, $\mathbf{z}^1$ and the rest:

$$\textcircled{\text{R}} = \underbrace{\int \mathrm{d}\mathbf{x} \prod_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\vec{\mathbf{z}}|\mathbf{x}) q(\mathbf{x}) \log \frac{p_\theta(\mathbf{z}^1|\mathbf{z}^2)}{q_\phi(\mathbf{z}^1|\mathbf{x})}}_{\textcircled{\text{R}_a}} + \underbrace{\sum_{m=2}^{L-1} \int \mathrm{d}\mathbf{x} \prod_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\vec{\mathbf{z}}|\mathbf{x}) q(\mathbf{x}) \log \frac{p_\theta(\mathbf{z}^m|\mathbf{z}^{m+1})}{q_\phi(\mathbf{z}^m|\mathbf{z}^{m-1}, \mathbf{x})}}_{\textcircled{\text{R}_b}}.$$

The first of these terms $\textcircled{\text{R}_a}$ is an expectation over a D_{KL}:

$$\textcircled{\text{R}_a} = -\,\mathbb{E}_{q_\phi(\mathbf{z}^2, \mathbf{x})}\, D_{\mathrm{KL}}(q_\phi(\mathbf{z}^1|\mathbf{x})||p_\theta(\mathbf{z}^1|\mathbf{z}^2)). \tag{D.7}$$

And the rest, $\textcircled{\text{R}_b}$, provides the D_{KL} divergences in the ELBO for all latent variables other than $\mathbf{z}^L$ and $\mathbf{z}^1$. It reduces to a sum of expectations over D_{KL} divergences, one per latent variable.

$$\textcircled{\text{R}_b} = \sum_{m=2}^{L-1} \int \mathrm{d}\mathbf{x} \prod_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\mathbf{z}^1|\mathbf{x}) q(\mathbf{x}) \prod_{k=1, \neq m}^{L-1} (q_\phi(\mathbf{z}^{k+1}|\mathbf{z}^k, \mathbf{x})) q_\phi(\mathbf{z}^m|\mathbf{z}^{m-1}, \mathbf{x}) \log \frac{p_\theta(\mathbf{z}^m|\mathbf{z}^{m+1})}{q_\phi(\mathbf{z}^m|\mathbf{z}^{m-1}, \mathbf{x})} \tag{D.8}$$

$$= -\sum_{m=2}^{L-1} \int \mathrm{d}\mathbf{x} \prod_{i=1}^{L} (\mathrm{d}\mathbf{z}^i) q_\phi(\mathbf{z}^1|\mathbf{x}) q(\mathbf{x}) \prod_{k=1, \neq m}^{L-1} (q_\phi(\mathbf{z}^{k+1}|\mathbf{z}^k, \mathbf{x})) D_{\mathrm{KL}}(q_\phi(\mathbf{z}^m|\mathbf{z}^{m-1}, \mathbf{x})||p_\theta(\mathbf{z}^m|\mathbf{z}^{m+1})) \tag{D.9}$$

$$= -\sum_{m=2}^{L-1} \mathbb{E}_{q_\phi(\mathbf{z}^{m+1}, \mathbf{z}^{m-1}, \mathbf{x})}\, D_{\mathrm{KL}}(q_\phi(\mathbf{z}^m|\mathbf{z}^{m-1}, \mathbf{x})||p_\theta(\mathbf{z}^m|\mathbf{z}^{m+1})). \tag{D.10}$$

Now we have:

$$\mathcal{L}(\theta, \phi; \mathcal{D}) = \mathbb{E}_{q(\vec{\mathbf{z}}, \mathbf{x})} \log p_\theta(\mathbf{x}|\vec{\mathbf{z}}) + \mathcal{H}(q(\mathbf{x})) + \textcircled{\text{R}_a} + \textcircled{\text{R}_b} + \textcircled{\text{S}} \tag{D.11}$$

We wish to apply TC decomposition to the top-most latent variable $\mathbf{z}^L$. $\textcircled{\text{S}}$ is an expectation over the D_{KL} divergence between $q_\phi(\mathbf{z}^L|\mathbf{z}^{L-1}, \mathbf{x})$ and $p(\mathbf{z}^L)$

$$\textcircled{\text{S}} = -\,\mathbb{E}_{q_\phi(\mathbf{z}^{L-1}, \mathbf{x})}\, D_{\mathrm{KL}}(q_\phi(\mathbf{z}^L|\mathbf{z}^{L-1}, \mathbf{x})||p(\mathbf{z}^L)) \tag{D.12}$$

Applying the decomposition, with j indexes over units in $\mathbf{z}^L$.

$$
\begin{aligned}
\textcircled{S} =& - \mathbb{E}_{q_\phi(\mathbf{z}^L, \mathbf{z}^{L-1}, \mathbf{x})} \Big[\log q_\phi(\mathbf{z}^L | \mathbf{z}^{L-1}, \mathbf{x}) - \log p(\mathbf{z}^L) + \log q_\phi(\mathbf{z}^L) \\
& \qquad - \log q_\phi(\mathbf{z}^L) + \log \prod_j q_\phi(\mathbf{z}_j^L) - \log \prod_j q_\phi(\mathbf{z}_j^L) \Big] \\
=& - \mathbb{E}_{q_\phi(\mathbf{z}^L, \mathbf{z}^{L-1}, \mathbf{x})} \left[\log \frac{q_\phi(\mathbf{z}^L | \mathbf{z}^{L-1}, \mathbf{x})}{q_\phi(\mathbf{z}^L)} \right] - \mathbb{E}_{q_\phi(\mathbf{z}^L)} \left[\log \frac{q_\phi(\mathbf{z}^L)}{\prod_j q_\phi(\mathbf{z}_j^L)} \right] \\
& - \mathbb{E}_{q_\phi(\mathbf{z}^L)} \left[\log \frac{\prod_j q_\phi(\mathbf{z}_j^L)}{p(\mathbf{z}^L)} \right] \\
=& - \mathbb{E}_{q_\phi(\mathbf{z}^L, \mathbf{z}^{L-1}, \mathbf{x})} \left[\log \frac{q_\phi(\mathbf{z}^L | \mathbf{z}^{L-1}) q(\mathbf{x})}{q_\phi(\mathbf{z}^L) q(\mathbf{x})} \right] - \mathbb{E}_{q_\phi(\mathbf{z}^L)} \left[\log \frac{q_\phi(\mathbf{z}^L)}{\prod_j q_\phi(\mathbf{z}_j^L)} \right] \\
& - \sum_j \mathbb{E}_{q_\phi(\mathbf{z}^L)} \left[\log \frac{q_\phi(\mathbf{z}_j^L)}{p(\mathbf{z}_j^L)} \right] \\
=& \underbrace{- \mathbb{E}_{q_\phi(\mathbf{z}^{L-1})) D_{\mathrm{KL}}(q_\phi(\mathbf{z}^L, \mathbf{x} | \mathbf{z}^{L-1}) || q_\phi(\mathbf{z}^L) q(\mathbf{x}))}}_{\textcircled{S_a}} \\
& \underbrace{- \sum_j D_{\mathrm{KL}}(q_\phi(\mathbf{z}_j^L) || p(\mathbf{z}_j^L))}_{\textcircled{S_b}} \underbrace{- D_{\mathrm{KL}}(q_\phi(\mathbf{z}^L) || \prod_j q_\phi(\mathbf{z}_j^L))}_{\textcircled{S_c}}
\end{aligned}
$$

Where we have used $p(\mathbf{z}^L) = \prod_j p(\mathbf{z}_j^L)$ for our chosen generative model, a product of independent unit-variance Gaussian distributions.

$$
\mathcal{L}(\theta, \phi; \mathcal{D}) = \mathbb{E}_{q(\vec{\mathbf{z}}, \mathbf{x})} \log p_\theta(\mathbf{x} | \vec{\mathbf{z}}) + \mathcal{H}(q(\mathbf{x})) + \textcircled{R_a} + \textcircled{R_b} + \textcircled{S_a} + \textcircled{S_b} + \textcircled{S_c} \quad \text{(D.13)}
$$

Giving us a decomposition of the evidence lower bound that reveals the TC-term in $\mathbf{z}^L$, as required. Multiplying this with a chosen prefactor β gives us the required form. $\square$

D.2 Minibatch Weighted Sampling

As in Chen et al. (2018a), applying β-TC decomposition requires us to calculate terms of the form:

$$\mathbb{E}_{q_\phi(\mathbf{z}^i)} \log q_\phi(\mathbf{z}^i) \tag{D.14}$$

The $i = 1$ case is covered in the appendix of Chen et al. (2018a). First we will repeat the argument for $i = 1$ as made in Chen et al. (2018a), but in our notation, and then we cover the case $i > 1$ for models with factorisation of $q_\phi(\vec{\mathbf{z}}|\mathbf{x})$ of Seatbelt VAEs.

D.2.1 MWS for β-TCVAEs

We denote $\mathcal{B}_M = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M\}$, a minibatch of datapoints drawn uniformly iid from $q(\mathbf{x}) = 1/N \sum_{n=1}^{N} \delta(\mathbf{x} - \mathbf{x}_n)$. For any minibatch we have $p(\mathcal{B}_M) = \frac{1}{N}^M$. Chen et al. (2018a) introduce $r(\mathcal{B}_M|\mathbf{x})$, the probability of a sampled minibatch given that one member is x and the remaining $M - 1$ points are sampled iid from $q(\mathbf{x})$, so $r(\mathcal{B}_M|\mathbf{x}) = \frac{1}{N}^{M-1}$.

$$\begin{aligned}
\mathbb{E}_{q_\phi(\mathbf{z}^1)} \log q_\phi(\mathbf{z}^1) &= \mathbb{E}_{q_\phi(\mathbf{z}^1,\mathbf{x})} [\log \mathbb{E}_{q(\mathbf{x})} [q_\phi(\mathbf{z}^1|\mathbf{x})]] && \text{(D.15)}\\
&= \mathbb{E}_{q_\phi(\mathbf{z}^1,\mathbf{x})} [\log \mathbb{E}_{p(\mathcal{B}_M)} [\frac{1}{M} \sum_{m=1}^{M} q_\phi(\mathbf{z}^1|\mathbf{x}_m)]] && \text{(D.16)}\\
&\geq \mathbb{E}_{q_\phi(\mathbf{z}^1,\mathbf{x})} [\log \mathbb{E}_{r(\mathcal{B}_M|\mathbf{x})} [\frac{p(\mathcal{B}_M)}{r(\mathcal{B}_M|\mathbf{x})} \frac{1}{M} \sum_{m=1}^{M} q_\phi(\mathbf{z}^1|\mathbf{x}_m)]] && \text{(D.17)}\\
&= \mathbb{E}_{q_\phi(\mathbf{z}^1,\mathbf{x})} [\log \mathbb{E}_{r(\mathcal{B}_M|\mathbf{x})} [\frac{1}{NM} \sum_{m=1}^{M} q_\phi(\mathbf{z}^1|\mathbf{x}_m)]] && \text{(D.18)}
\end{aligned}$$

So then during training, one samples a minibatch $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M\}$ and can estimate $\mathbb{E}_{q_\phi(\mathbf{z}^1)} \log q_\phi(\mathbf{z}^1)$ as:

$$\mathbb{E}_{q_\phi(\mathbf{z}^1)} \log q_\phi(\mathbf{z}^1) \approx \frac{1}{M} \sum_{i=1}^{M} [\log \sum_{j=1}^{M} q_\phi(\mathbf{z}_i^1|\mathbf{x}_j) - \log NM] \tag{D.19}$$

and $\mathbf{z}_i^1$ is a sample from $q_\phi(\mathbf{z}^1|\mathbf{x}_i)$.

D.2.2 Minibatch Weighted Sampling for Seatbelt-VAEs

Here we have that $q(\vec{\mathbf{z}}, \mathbf{x}) = \prod_{l=2}^{L} [q_\phi(\mathbf{z}^l|\mathbf{z}^{l-1}, \mathbf{x})] q_\phi(\mathbf{z}^1|\mathbf{x}) q(\mathbf{x})$. Now instead of having a minibatch of datapoints, we have a minibatch of draws of $\mathbf{z}^{i-1}$: $\mathcal{B}_M^{i-1} = \{\mathbf{z}_1^{i-1}, \mathbf{z}_2^{i-1}, ..., \mathbf{z}_M^{i-1}\}$. Each member of which is the result of sequentially sampling along a chain, starting with some particular datapoint $\mathbf{x}_m \sim q(\mathbf{x})$.

For $i > 2$, members of $\mathcal{B}_M^{i-1}$ are drawn:

$$\mathbf{z}_j^{i-1} \sim q_\phi(\mathbf{z}^{i-1}|\mathbf{z}_j^{i-2}, \mathbf{x}_j) \tag{D.20}$$

and for $i = 2$:

$$\mathbf{z}_j^{1} \sim q_\phi(\mathbf{z}^{1}|\mathbf{x}_j) \tag{D.21}$$

Thus each member of this batch $\mathcal{B}_M^{i-1}$ is the descendant of a particular datapoint that was sampled in an iid minibatch $\mathcal{B}_M$ as defined above. We similarly define $r(\mathcal{B}_M^{i-1}|\mathbf{z}^{i-1}, \mathbf{x})$ as the probability of selecting a particular minibatch $\mathcal{B}_M^{i-1}$ of these values out from our set $\{(\mathbf{x}_n, \mathbf{z}_n^{i-1})\}$ (of cardinality N) given that we have selected into our minibatch one particular pair of values $(\mathbf{x}, \mathbf{z}^{i-1})$ from these N values. Like above, $r(\mathcal{B}_M^{i-1}|\mathbf{z}^{i-1}, \mathbf{x}) = \frac{1}{N}^{M-1}$

Now we can consider $\mathbb{E}_{q_\phi(\mathbf{z}^i)} \log q_\phi(\mathbf{z}^i)$ for $i > 1$:

$$\begin{aligned}
\mathbb{E}_{q_\phi(\mathbf{z}^i)} \log q_\phi(\mathbf{z}^i) &= \mathbb{E}_{q_\phi(\mathbf{z}^i, \mathbf{z}^{i-1}, \mathbf{x})} \left[\log \mathbb{E}_{q_\phi(\mathbf{z}^{i-1}, \mathbf{x})} \left[q_\phi(\mathbf{z}^i|\mathbf{z}^{i-1}, \mathbf{x}) \right] \right] && \text{(D.22)} \\
&= \mathbb{E}_{q_\phi(\mathbf{z}^i, \mathbf{z}^{i-1}, \mathbf{x})} \left[\log \mathbb{E}_{p(\mathcal{B}_M^{i-1})} \left[\frac{1}{M} \sum_{m=1}^{M} q_\phi(\mathbf{z}^i|\mathbf{z}_m^{i-1}, \mathbf{x}_m) \right] \right] && \text{(D.23)} \\
&\geq \mathbb{E}_{q_\phi(\mathbf{z}^i, \mathbf{z}^{i-1}, \mathbf{x})} \left[\log \mathbb{E}_{r(\mathcal{B}_M^{i-1}|\mathbf{z}^{i-1}, \mathbf{x})} \left[\frac{p(\mathcal{B}_M^{i-1})}{r(\mathcal{B}_M^{i-1}|\mathbf{z}^{i-1}, \mathbf{x})} \frac{1}{M} \sum_{m=1}^{M} q_\phi(\mathbf{z}^i|\mathbf{z}_m^{i-1}, \mathbf{x}_m) \right] \right] && \text{(D.24)} \\
&= \mathbb{E}_{q_\phi(\mathbf{z}^i, \mathbf{z}^{i-1}, \mathbf{x})} \left[\log \mathbb{E}_{r(\mathcal{B}_M^{i-1}|\mathbf{z}^{i-1}, \mathbf{x})} \left[\frac{1}{NM} \sum_{m=1}^{M} q_\phi(\mathbf{z}^i|\mathbf{z}_m^{i-1}, \mathbf{x}_m) \right] \right] && \text{(D.25)}
\end{aligned}$$

Where we have followed the same steps as in the previous subsection.

During training, one samples a minibatch $\{\mathbf{z}_1^{i-1}, \mathbf{z}_2^{i-1}, ..., \mathbf{z}_M^{i-1}\}$, where each is constructed by sampling ancestrally. Then one can estimate $\mathbb{E}_{q_\phi(\mathbf{z}^i)} \log q_\phi(\mathbf{z}^i)$ as:

$$\mathbb{E}_{q_\phi(\mathbf{z}^i)} \log q_\phi(\mathbf{z}^i) \approx \frac{1}{M} \sum_{k=1}^{M} \left[\log \sum_{j=1}^{M} q_\phi(\mathbf{z}_k^i|\mathbf{z}_j^{i-1}, \mathbf{x}_j) - \log NM \right] \tag{D.26}$$

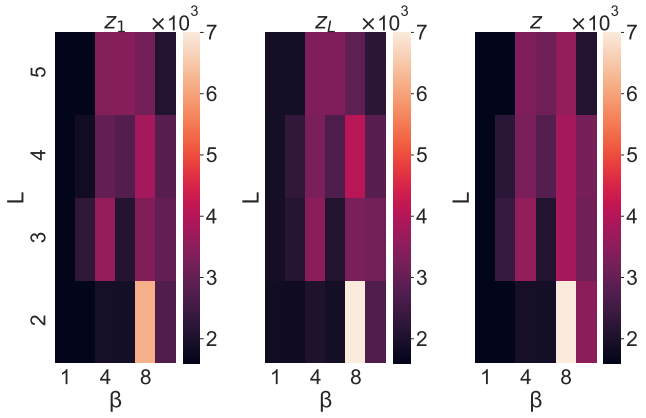
and $\mathbf{z}_k^i$ is a sample from $q_\phi(\mathbf{z}^i|\mathbf{z}_k^{i-1}, \mathbf{x}_k)$. In our approach we only need terms of this form for $i = L$, so we have:

$$\mathbb{E}_{q_\phi(\mathbf{z}^L)} \log q_\phi(\mathbf{z}^L) \approx \frac{1}{M} \sum_{k=1}^{M} [\log \sum_{j=1}^{M} q_\phi(\mathbf{z}_k^L|\mathbf{z}_j^{L-1}, \mathbf{x}_j) - \log NM] \tag{D.27}$$

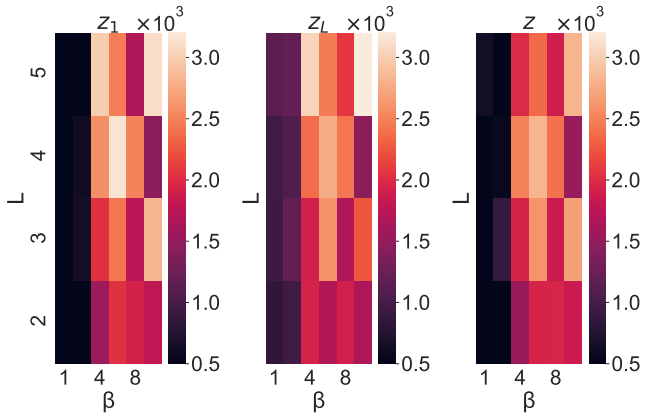
and $\mathbf{z}_k^L$ is a sample from $q_\phi(\mathbf{z}^L|\mathbf{z}_k^{L-1}, \mathbf{x}_k)$.

D.3 Seatbelt-VAE Results

D.3.1 Seatbelt-VAE layerwise attacks



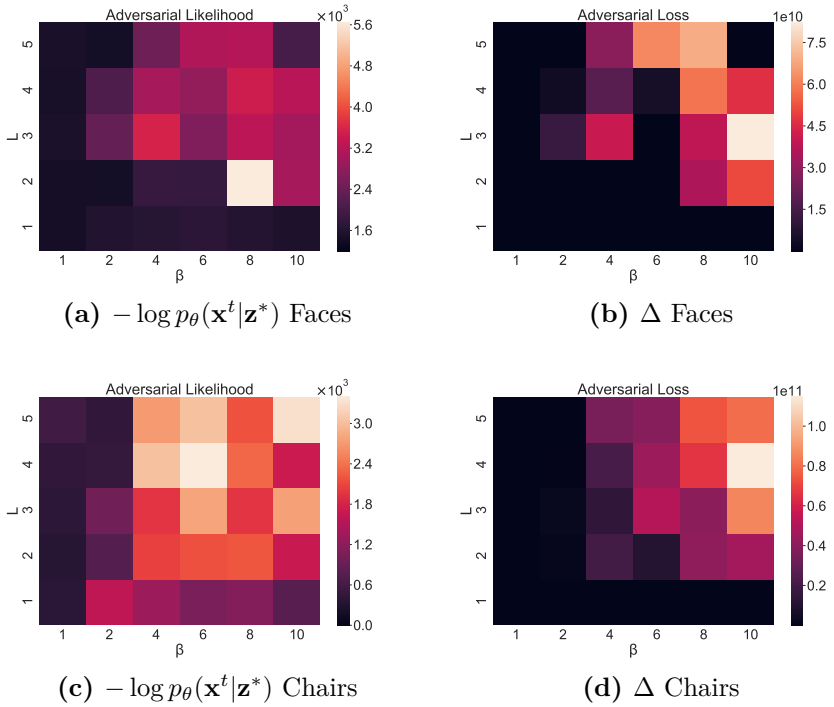
(a) 3D Faces



(b) Chairs

Figure D.1: $-\log p_\theta(\mathbf{x}^t|\tilde{\mathbf{z}}), \tilde{z} \sim q(\mathbf{z}|\mathbf{x}+d)$ where d is some adversarial distortion, for Seatbelt-VAEs trained on (a) 3D Faces and (b) Chairs; over β and L values for *latent* attacks. We attack the bottom layer ($\mathbf{z}^1$), the top layer ($\mathbf{z}^L$), and finally show the effect when attacking all layers ($\mathbf{z}$). Larger values of $-\log p_\theta(\mathbf{x}^t|\tilde{\mathbf{z}})$ correspond to less successful adversarial attacks. Generally attacking all layers seems to give the attacker a slight advantage (as seen by the slightly lower $-\log p_\theta(\mathbf{x}^t|\tilde{\mathbf{z}})$ values for Faces and Chairs).

D.3.2 Seatbelt-VAE attacks by model depth and β

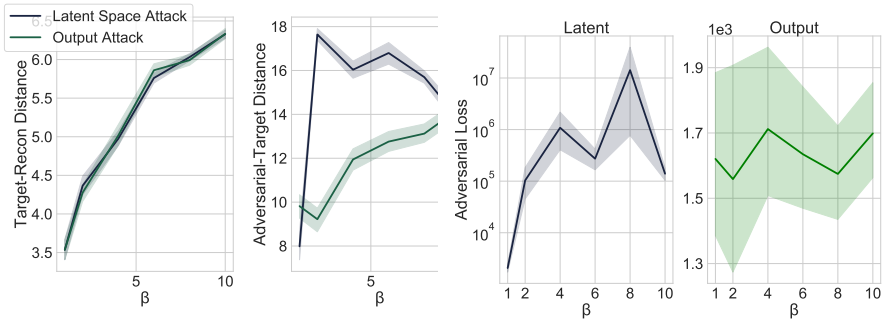


(a) $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ Faces **(b)** Δ Faces

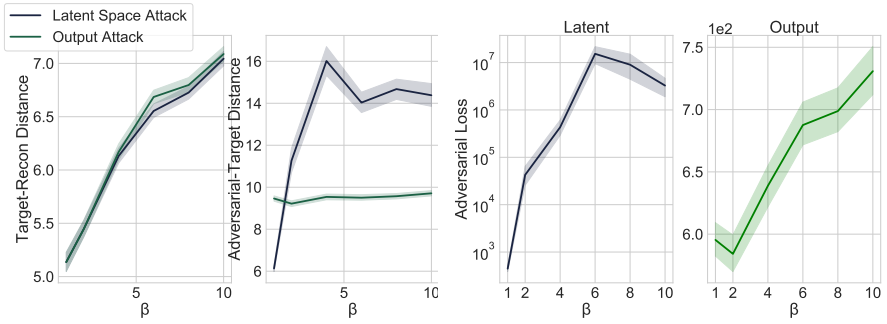
(c) $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ Chairs **(d)** Δ Chairs

Figure D.2: Here we measure the robustness of TC-penalised models numerically. Sub-figures (a) and (c) show $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$, the adversarial likelihood of a target image $\mathbf{x}^t$ given an attacked latent representation $\mathbf{z}^*$ for Seatbelt-VAEs for Chairs and 3D Faces. Larger likelihood values correspond to less successful adversarial attacks. Sub-figures (b) and (d) show adversarial loss Δ for Seatbelt-VAEs for Chairs and 3D Faces. We show these likelihood and loss values over β and L (total number of stochastic layers) values for attacks. Note that the bottom rows of all figures have $L = 1$, and thus correspond to β-TCVAEs. The leftmost column corresponds to models with $\beta = 1$, which are vanilla VAEs and hierarchical VAEs. As we go to the largest values of β and L for both Chairs and 3D Faces, Δ grows by a factor of $\approx 10^7$ and $-\log p_\theta(\mathbf{x}^t|\mathbf{z}^*)$ doubles. These results tell us that depth and TC-penalisation together, i.e Seatbelt-VAE, can offer immense protection from the adversarial attacks studied.

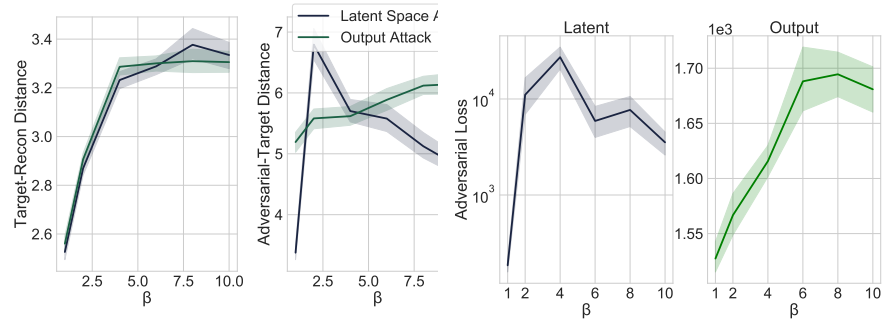
D.4 Aggregate Analysis of Adversarial Attack



(a) dSprites Distances **(b)** dSprites Losses



(c) Chairs Distances **(d)** Chairs Losses



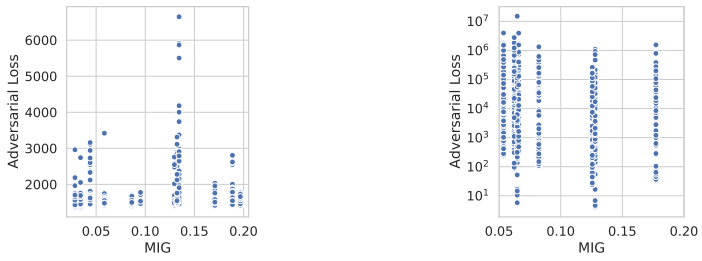
(e) 3D Faces Distances **(f)** 3D Faces Losses

Figure D.3: Plots showing the effect of varying β in a β-TCVAE trained on dSprites (a,b), Chairs (c,d), and 3D Faces (d,e) on: the L_2 distance from the adversarial target x^t to its reconstruction when given as input (target-recon distance) and the L_2 distance between the adversarial input x^* and x^t (adversarial-target distance); and the adversarial objectives Δ. For latent attacks the adversarial-target L_2 distance grows more rapidly than the target-recon distance (i.e the degradation of reconstruction quality) as we increase β. This effect is much less clear for output attacks. This makes it apparent that the robustness we see in β-TCVAE to latent space adversarial attacks is not due the degradation in reconstruction quality we see as β increases. It is also apparent that increasing β increases the adversarial loss for latent attacks and output attacks.

D.4.1 Disentangling and Robustness?

Although we are using regularisation methods that were initially proposed to encourage disentangled representations, we are interested here in their effect on robustness *not* whether the representations we learn are in fact disentangled. This is not least due to the questions that have arisen about the hyperparameter tuning required for disentangled representations (Locatello et al., 2019; Rolinek et al., 2019). For us the β prefactor is just the degree of regularisation imposed.

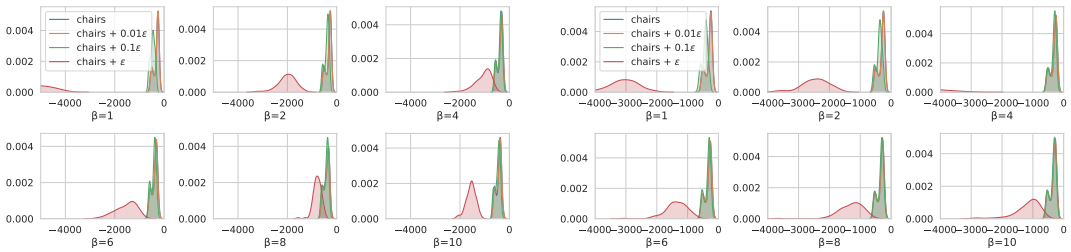
However, it may be of interest to see what relationship, if any, exists between the ease of attacking of a model and how disentangled it is. Here we show the MIG score (Chen et al., 2018a) against the achieved adversarial loss on the Faces data for β-TCVAEs. MIG measures the degree to which representations are disentangled and larger adversarial losses correspond to a less successful attack. Shading is over the range of β and d_z values. There does not seem to be any simple correspondence between increased MIG and increases in adversarial loss, indicative of a less successful attack.



(a) Faces **(b)** Chairs

Figure D.4: Adversarial attack loss reached vs MIG score for β-TCVAEs trained on Faces and Chairs presented for a range of $\beta = \{1, 2, 4, 6, 8, 10\}$ and $d_z = \{8, 32\}$ values.

D.5 Robustness to Noise



(a) Chairs β-TCVAE $\log p_\theta(\mathbf{x}|\mathbf{z})$ **(b)** Chairs Seatbelt-VAE $\log p_\theta(\mathbf{x}|\mathbf{z})$

Figure D.5: Here we measure the robustness of both β-TCVAE and Seatbelt-VAE when Gaussian noise is added to Chairs. Within each plot a range of β values are shown. We evaluate each model's ability to decode a noisy embedding to the original non-noised data $\mathbf{x}$ by measuring the distribution of $\log p_\theta(\mathbf{x}|\mathbf{z})$ when $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x} + a\boldsymbol{\epsilon})$ (a some scaling factor taking values in $\{0.1, 0.5, 1\}$ and $\epsilon \sim \mathcal{N}(0, 1)$) for which higher values indicate better denoising. We show these likelihood values as density plots for the β-TCVAE in (a) and for the Seatbelt-VAE with $L = 4$ in (b), taking $\beta \in \{1, 2, 4, 6, 8, 10\}$. Note the axis scalings are different for each subplot. We see that for both models using $\beta > 1$ produces autoencoders that are better at denoising their inputs. Namely, the mean of the density, i.e. $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}+\boldsymbol{\epsilon})}[\log p_\theta(\mathbf{x}|\mathbf{z})]$, shifts dramatically to higher values for $\beta > 1$ relative to $\beta = 1$. In other words, for both these models, the likelihood of the dataset in the noisy setting is much closer to the non-noisy dataset when $\beta > 1$ across all noise scales (0.1ϵ, 0.5ϵ, ϵ).

D.6 Implementation Details

All runs were done on the Azure cloud system on NC6 GPU machines.

D.6.1 Encoder and Decoder Architectures

We used the same convolutional network architectures as Chen et al. (2018a). For the encoders of all our models ($q(\cdot|\mathbf{x})$) we used purely convolutional networks with 5 convolutional layers. When training on single-channel (binary/greyscale) datasets such as dSprites, 3D Faces, or Chairs the 5 layers took the following number of filters in order: $\{32, 32, 64, 64, 512\}$. For more complex RGB datasets, such as CelebA, the layers had the following number of filters in order: $\{64, 64, 128, 128, 512\}$. The mean and variance of the amortised posteriors are the output of dense layers acting on the output of the purely convolutional network, where the number of neurons in these layers is equal to the dimensionality of the latent space $\mathcal{Z}$.

Similarly, for the decoders ($p(\mathbf{x}|\mathbf{z})$) of all our models we also used purely convolutional networks with 6 deconvolutional layers. When training on single-channel (binary/greyscale) datasets, dSprites, 3D Faces, or Chairs, the 6 layers took the following number of filters in order: $\{512, 64, 64, 32, 32, 1\}$. For CelebA the layers had the following number of filters in order: $\{512, 128, 128, 64, 64, 3\}$. The mean of the likelihood $p(\mathbf{x}|\cdot)$ was directly encoded by the final de-convolutional layer. The variance of the decoder, σ, was fixed to 0.1.

For β-TCVAE the range of $d_{\mathbf{z}}$ values used was $\{4, 6, 8, 16, 32, 64, 128\}$. For Seatbelt-VAEs the number of units in each layer $\mathbf{z}^i$ decreases sequentially. There is a list z_sizes for each dataset, and for a model of L layers that the last L entries to give $d_{\mathbf{z},i}, i \in \{1, ..., L\}$.

$$\{d_{\mathbf{z}}\}^{\text{dSprites}} = \{96, 48, 24, 12, 6\} \tag{D.28}$$

$$\{d_{\mathbf{z}}\}^{\text{Chairs}} = \{96, 48, 24, 12, 6\} \tag{D.29}$$

$$\{d_{\mathbf{z}}\}^{\text{3DFaces}} = \{96, 48, 24, 12, 6\} \tag{D.30}$$

$$\{d_{\mathbf{z}}\}^{\text{CelebA}} = \{256, 128, 64, 32\} \tag{D.31}$$

For Seatbelt-VAEs we also have the mappings $q_\phi(\mathbf{z}^{i+1}|\mathbf{z}^i, \mathbf{x})$ and $p_\theta(\mathbf{z}^i|\mathbf{z}^{i+1})$. These are amortised as MLPs with 2 hidden layers with batchnorm and Leaky-ReLU activation. The dimensionality of the hidden layers also decreases as a function of layer index i:

$$d_{\mathbf{h}}(q_\phi(\mathbf{z}^{i+1}|\mathbf{z}^i, \mathbf{x})) = \text{h}_{\text{sizes}}[i] \tag{D.32}$$

$$d_{\mathbf{h}}(p_\theta(\mathbf{z}^i|\mathbf{z}^{i+1})) = \text{h}_{\text{sizes}}[i] \tag{D.33}$$

$$\text{h}_{\text{sizes}} = [1024, 512, 256, 128, 64] \tag{D.34}$$

To train the model we used ADAM (Kingma & Lei Ba, 2015) with default parameters, a cosine decaying learning rate of 0.001, and a batch size of 1024. All data was preprocessed to fall on the interval -1 to 1. CelebA and Chairs were both downsampled and cropped as in Chen et al. (2018a) and Kulkarni et al. (2015) respectively. We find that using *free-bits* regularisation (Kingma et al., 2016) greatly ameliorates the optimisation challenges associated with DLGMs.

E

Appendix for Towards a Theoretical Understanding of the Robustness of Variational Autoencoders

E.1 Choosing r for r-robustness

Proposition E.1. *For any input and perturbation, a necessary requirement for a VAE with a Gaussian encoder to satisfy r-robustness is that*

$$r > \sqrt{2\mathrm{Tr}(\boldsymbol{\Sigma}(\mathbf{x}))} + \mathcal{O}(\boldsymbol{\varepsilon}) \tag{E.1}$$

where $\boldsymbol{\Sigma}(\mathbf{x}) = \mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))\boldsymbol{\sigma}^2_\phi(\mathbf{x})\mathbf{J}^T_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$, $(\mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))_{i,j} = \partial g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))_i / \partial(\boldsymbol{\mu}_\phi(\mathbf{x}))_j$, *and* $\mathcal{O}(\boldsymbol{\varepsilon})$ *represents higher order terms that tend to zero in the limit* $\boldsymbol{\sigma}_\phi(\mathbf{x}) \to \mathbf{0}$.

We provide empirical confirmations in Appendix E.4.1.1 that show that the r for r-robustness scales with the encoder variance.

Proof. Let $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$ be the result of mapping to the encoder mean ($\boldsymbol{\mu}_\phi$) and then decoding to the likelihood mean (g_θ), and $\Delta(\mathbf{x}) = g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x})) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$ we want to find a bound for r for which:

$$p(||\Delta(\mathbf{x})||_2 \leq r) > p(||\Delta(\mathbf{x})||_2 > r) \tag{E.2}$$

where as before $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Here we can invoke Taylor's theorem on $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x}))$ around the deterministic mapping $\boldsymbol{\mu}_\phi(\mathbf{x})$. Namely, if we assume that all terms in Hessian of $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x})$ are finite (i.e. $|\partial^2 g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x})_i / \partial \boldsymbol{\mu}_\phi(\mathbf{x})_j \boldsymbol{\mu}_\phi(\mathbf{x})_k| < \infty \, \forall i, j, k$), then we have:

$$g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\epsilon}) = g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x})) + \mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))(\boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x})) + \mathcal{O}(\boldsymbol{\varepsilon}) \tag{E.3}$$

where $\mathcal{O}(\varepsilon)$ represents asymptotically dominated higher order terms that go to zero in the limit of small $\boldsymbol{\sigma}_\phi(\mathbf{x}))$ and $\mathbf{J}_\theta$ is defined element-wise as:

$$\mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))_{i,j} = \frac{\partial g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))_i}{\partial (\boldsymbol{\mu}_\phi(\mathbf{x}))_j} \tag{E.4}$$

Note that $\mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))(\boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x}))$ is distributed according to the multivariate Gaussian

$$\mathcal{N}(0, \mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))\boldsymbol{\sigma}_\phi^2(\mathbf{x})\mathbf{J}_\theta^T(\boldsymbol{\mu}_\phi(\mathbf{x})))$$

Given these definitions

$$p(||\Delta(\mathbf{x})||_2 \leq r) > p(||\Delta(\mathbf{x})||_2 > r) \tag{E.5}$$

$$\Leftrightarrow p(||\Delta(\mathbf{x})||_2 \leq r) > 0.5 \tag{E.6}$$

$$\Leftrightarrow p(||\mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))(\boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x})) + \mathcal{O}(\boldsymbol{\varepsilon})||_2 < r) > 0.5 \tag{E.7}$$

$$\Leftrightarrow p(||\mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))(\boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x})) + \mathcal{O}(\boldsymbol{\varepsilon})||_2^2 < r^2) > 0.5 \tag{E.8}$$

We must now consider the distribution of the square norm of $\boldsymbol{\mathcal{E}}(\mathbf{x}) = \mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))(\boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x}))$. Let

$$Q(\boldsymbol{\mathcal{E}}(\mathbf{x})) = ||\boldsymbol{\mathcal{E}}(\mathbf{x})||_2^2 = \boldsymbol{\mathcal{E}}(\mathbf{x})^T \boldsymbol{\mathcal{E}}(\mathbf{x}) \tag{E.9}$$

$$\boldsymbol{\Sigma}(\mathbf{x}) = \mathbf{J}_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))\boldsymbol{\sigma}_\phi^2(\mathbf{x})\mathbf{J}_\theta^T(\boldsymbol{\mu}_\phi(\mathbf{x})) \tag{E.10}$$

$$\mathbf{Y}(\mathbf{x}) = \boldsymbol{\Sigma}(\mathbf{x})^{-\frac{1}{2}} \boldsymbol{\mathcal{E}}(\mathbf{x}) \tag{E.11}$$

Given that we restrict ourselves to positive activation functions, $\mathbf{J}_\theta$ is positive and $\boldsymbol{\Sigma}(\mathbf{x})$ will be positive semi definite and is invertible. As such we have $Q(\boldsymbol{\mathcal{E}}) = \mathbf{Y}^T(\mathbf{x})\boldsymbol{\Sigma}(\mathbf{x})\mathbf{Y}(\mathbf{x})$.

Using the spectral decomposition theorem we can write that $\mathbf{\Sigma}(\mathbf{x}) = \mathbf{P}^T(\mathbf{x})\mathbf{\Lambda}(\mathbf{x})\mathbf{P}(\mathbf{x})$ where $\mathbf{P}^T(\mathbf{x})\mathbf{P}(\mathbf{x}) = \mathbf{I}$ and $\mathbf{\Lambda}(\mathbf{x})$ is the diagonal matrix of the eigenvalues of $\mathbf{\Sigma}(\mathbf{x})$, $\lambda_1, \ldots, \lambda_{d_\mathcal{X}}$, where $d_\mathcal{X}$ is the dimensionality of the data-space. Given that $\mathbf{\Sigma}(\mathbf{x})$ is positive semi definite $\mathbf{\Lambda}(\mathbf{x})$ will only have positive values.

Let $\mathbf{U}(\mathbf{x}) = \mathbf{P}(\mathbf{x})\mathbf{Y}(\mathbf{x}) = \mathbf{P}(\mathbf{x})\mathbf{\Sigma}(\mathbf{x})^{-\frac{1}{2}}\boldsymbol{\mathcal{E}}(\mathbf{x})$, which is multivariate Gaussian with identity matrix and zero mean. We have that:

$$Q(\boldsymbol{\mathcal{E}}) = \mathbf{Y}^T(\mathbf{x})\mathbf{\Sigma}(\mathbf{x})\mathbf{Y}(\mathbf{x}) \tag{E.12}$$

$$= \mathbf{Y}^T(\mathbf{x})\mathbf{P}^T(\mathbf{x})\mathbf{\Lambda}(\mathbf{x})\mathbf{P}(\mathbf{x})\mathbf{Y}(\mathbf{x}) \tag{E.13}$$

$$= \mathbf{U}^T(\mathbf{x})\mathbf{\Lambda}(\mathbf{x})\mathbf{U}(\mathbf{x}) \tag{E.14}$$

As such:

$$\sum_{i=1}^{d_\mathcal{X}}(\boldsymbol{\mathcal{E}}_i)^2 = \mathbf{U}^T(\mathbf{x})\mathbf{\Lambda}(\mathbf{x})\mathbf{U}(\mathbf{x}) = \sum_{i=1}^{d_\mathcal{X}}\lambda_i(\mathbf{U}_i(\mathbf{x}))^2, \quad \lambda_i(\mathbf{U}_i(\mathbf{x}))^2 \sim \Gamma\left(\frac{1}{2}, 2\lambda_i\right) \tag{E.15}$$

This comes from the fact that for $\lambda_i\mathbf{X}, \mathbf{X} \sim \Gamma\left(\frac{1}{2}, 2\right)$ we have that $\lambda_i\mathbf{X} \sim \Gamma\left(\frac{1}{2}, 2\lambda_i\right)$.

To establish a lower bound on r, we use Markov's inequality which states that:

$$p(||\boldsymbol{\mathcal{E}}(\mathbf{x}) + \mathcal{O}(\boldsymbol{\varepsilon})||_2^2 > r^2) < \frac{\mathbb{E}\,||\boldsymbol{\mathcal{E}}(\mathbf{x}) + \mathcal{O}(\boldsymbol{\varepsilon})||_2^2}{r^2} \tag{E.16}$$

Here $\mathbb{E}\,||\boldsymbol{\mathcal{E}}(\mathbf{x})||_2^2 = \mathbb{E}\sum_{i=1}^{d_\mathcal{X}}(\boldsymbol{\mathcal{E}}_i(\mathbf{x}))^2 = \mathbb{E}\sum_{i=1}^{d_\mathcal{X}}(\lambda_i(\mathbf{U}_i)^2(\mathbf{x}))^2$, which is simply $\sum_{i=1}^{d_\mathcal{X}}\lambda_i$.

Recall that we want: $p(||\boldsymbol{\mathcal{E}}(\mathbf{x}) + \mathcal{O}(\boldsymbol{\varepsilon})||_2^2 > r^2) < 0.5$. As such

$$r > \sqrt{2\sum_{i=1}^{d_\mathcal{X}}\lambda_i} + \mathcal{O}(\boldsymbol{\varepsilon}) = \sqrt{2\mathrm{Tr}(\mathbf{\Sigma}(\mathbf{x}))} + \mathcal{O}(\boldsymbol{\varepsilon}) \tag{E.17}$$

□

E.2 Margin for r-robustness in $\mathcal{X}$

Theorem 6.1. *Consider a VAE with a diagonal-variance Gaussian encoder, an input* $\mathbf{x}$*, and an output margin* $r \in \mathbb{R}$ *such that the VAE is* r*-robust to the stochasticity of the encoder when the* $\mathbf{x}$ *is unperturbed as per* (6.2)*. Assuming standard regularity assumptions (discussed in the proof) hold for* $\boldsymbol{\mu}_\phi(\mathbf{x})$*, then*

$$R^r_{\mathcal{X}}(\mathbf{x}) \geq \frac{(\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i)\Phi^{-1}(p(||\Delta(\mathbf{x})||_2 \leq r))}{||\mathbf{J}^\mu_\phi(\mathbf{x})||_F} + \mathcal{O}(\boldsymbol{\varepsilon}) \tag{6.4}$$

where $\mathcal{O}(\boldsymbol{\varepsilon})$ *represents higher order dominated terms that disappear in the limit of small perturbations,* Φ^{-1} *is the probit function,* $\mathbf{J}^\mu_\phi(\mathbf{x})_{i,j} = \partial \boldsymbol{\mu}_\phi(\mathbf{x})_i / \partial \mathbf{x}_j$ *is the Jacobian of* $\boldsymbol{\mu}_\phi(\mathbf{x})$*, and* $||\cdot||_F$ *is the Frobenius norm.*

Proof. Suppose we have an r for which r-robustness is satisfied before any perturbation is added to the VAE input. First we want to establish a margin in the latent space $\mathcal{Z}$ for which our model is robust given a perturbation in the latent space.

To do this, we first define

$$\Delta_e(\mathbf{y}) = g_\theta(\mathbf{y}) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x})), \quad \mathbf{y} \in \mathcal{Z} \tag{E.18}$$

where g_θ is the decoder network and $\mathbf{y}$ is an arbitrary realization of the latents. Note here that there is an implicit dependency on $\mathbf{x}$, but as this input is fixed we will ignore this dependency throughout. Let $\mathbf{z} = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\eta} \circ \boldsymbol{\sigma}_\phi(\mathbf{x})$ be the random variable produced by the embedding, i.e. the latent sampled by the encoder. We want to find a bound R^r_e for which:

$$\|\boldsymbol{\delta}_z\|_2 \leq R^r_e \quad \Leftrightarrow \quad p(||\Delta_e(\mathbf{z} + \boldsymbol{\delta}_z)||_2 \leq r) > p(||\Delta_e(\mathbf{z} + \boldsymbol{\delta}_z)||_2 > r) \tag{E.19}$$

such that r-robustness is satisfied on the decoder output when we apply a deterministic a perturbation $\boldsymbol{\delta}_z$ of maximum size R^r_e to the random variable $\mathbf{z}$. Note that all the stochasticity is contained in $\boldsymbol{\eta}$.

Let A^r denote the set of $\boldsymbol{\delta}_z$ for which (E.19) holds and conversely let B^r be the set of $\boldsymbol{\delta}_z$ for which it does not. By assumption in the Theorem, then $\mathbf{0} \in A^r$ as

the unperturbed input satisfies r-robustness. Moreover, we also have that this unperturbed input $\boldsymbol{\delta}_z$ has a probability $p_\Delta(\mathbf{0}) := p(||\Delta_e(\mathbf{z})||_2 \leq r) = p(||\Delta(\mathbf{x})||_2 \leq r) > 0.5$ of returning a reconstruction with r of $g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))$.

Now we know that $\mathbf{z}$ is a Gaussian random variable and so regardless of form of the decoder, $p_\Delta(\boldsymbol{\delta}_z) := p(||\Delta_e(\mathbf{z} + \boldsymbol{\delta}_z)||_2 \leq r)$ must vary smoothly as we change $\boldsymbol{\delta}_z$. In essence, as we increase the size of the perturbation $\boldsymbol{\delta}_z$ slowly from zero, the distribution of $\mathbf{z} + \boldsymbol{\delta}_z$ will still have most of its mass of the same region $\mathbf{z}$. When coupled with the fact that we have some "excess probability" $p_\Delta(\mathbf{0}) - 0.5$ beyond what it is needed for r-robustness, there must be at certain degree to which we can increase $\boldsymbol{\delta}_z$ before all this excess probability is "used up". We can then use this to construct a bound for R_e^r by considering the minimum $\boldsymbol{\delta}_z$ to break r-robustness in the "worst-case" setting for the boundary between A^r and B^r.

Intuitively as shown in Figure E.1, and also more formally using the Neyman-Pearson lemma (Neyman & Pearson, 1933) by analogy to the approach of Cohen et al. (2019), this worst case setting will occur when the boundary between A^r and B^r is a straight line perpendicular to the direction of lowest variance for $\mathbf{z}$ (remembering that this is Gaussian distributed) and $\boldsymbol{\delta}_z$ is increased in this direction of lowest variance. In essence, this is the setup where our excess probability is used up most quickly for a given $\|\boldsymbol{\delta}_z\|_2$. By assumption in the theorem statement, we are using a diagonal covariance encoder and so this direction of lowest variance is the latent variable corresponding to $\arg\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i$. Further, by noting that we need only consider the marginal distribution in this dimension, it is straightforward to see that the bound is reached when

$$\|\boldsymbol{\delta}_z\|_2 = \left(\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i\right) \Phi^{-1}\left(p_\Delta(\mathbf{0})\right) = \left(\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i\right) \Phi^{-1}\left(p(||\Delta(\mathbf{x})||_2 \leq r)\right) \quad \text{(E.20)}$$

where Φ^{-1} is the inverse cumulative distribution function for a unit Gaussian, i.e. the probit function. Note that this yields $\|\boldsymbol{\delta}_z\|_2 = 0$ if $p(||\Delta(\mathbf{x})||_2 \leq r) = 0$, such we get the excepted result that our margin is zero is r-robustness only just holds without an input perturbation.

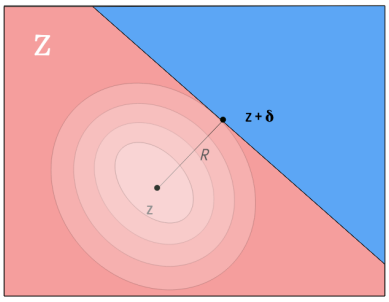


Figure E.1: Illustration of the boundary R we are measuring in $\mathcal{Z}$. Red represents spaces where A^r is satisfied. Blue represent spaces where B^r is satisfied. The concentric ellipsoids centered on $\mathbf{z}$ are the contours of $\mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}^2_\phi(\mathbf{x}))$. R is the minimum distance δ for which A^r is satisfied. The line dividing the two spaces represent the Neyman-Pearson "worst-case" model and is along the direction of minimum *variance*, $\min_i \boldsymbol{\sigma}^2_\phi(\mathbf{x})_i$.

Next we need to relate $\|\boldsymbol{\delta}_z\|_2$ to $\|\boldsymbol{\delta}_x\|_2$. Here we can straightforwardly invoke Taylor's theorem on $\boldsymbol{\mu}_\phi(\mathbf{x} + \boldsymbol{\delta}_x)$ around the original input $\mathbf{x}$. Namely, if we assume that all terms in Hessian of $\mu_\phi(\mathbf{x})$ are finite (i.e. $|\partial^2 \boldsymbol{\mu}_\phi(\mathbf{x})_i / \partial \mathbf{x}_j \mathbf{x}_k| < \infty \, \forall i, j, k$), then we have

$$\boldsymbol{\delta}_z = \boldsymbol{\mu}_\phi(\mathbf{x} + \boldsymbol{\delta}_x) - \boldsymbol{\mu}_\phi(\mathbf{x}) = \mathbf{J}^\mu_\phi(\mathbf{x})\boldsymbol{\delta}_x + \mathcal{O}(\varepsilon) \tag{E.21}$$

where $\mathcal{O}(\varepsilon)$ represents asymptotically dominated higher order terms that go to zero in the limit of small $\boldsymbol{\delta}_x$. We thus have

$$\|\boldsymbol{\delta}_x\|_2 \leq \frac{\|\boldsymbol{\delta}_z\|_2}{\|\mathbf{J}^\mu_\phi(\mathbf{x})\|_F} + \mathcal{O}(\varepsilon) \tag{E.22}$$

where $\mathcal{O}(\varepsilon)$ again represents asymptotically dominated higher order terms (note though these are not the same terms as in (E.21)). To complete the proof we now simply combine this with (E.20) to give the $\|\boldsymbol{\delta}_x\|_2$ at which the bound is reached and thus the $R^r_{\mathcal{X}}(\mathbf{x})$ quoted in the theorem, namely

$$R^r_{\mathcal{X}}(\mathbf{x}) \geq \frac{(\min_i \boldsymbol{\sigma}_\phi(\mathbf{x})_i)\Phi^{-1}(p(||\Delta(\mathbf{x})||_2 \leq r))}{||\mathbf{J}^\mu_\phi(\mathbf{x})||_F} + \mathcal{O}(\varepsilon) \tag{E.23}$$

where the inequality comes from the fact that the $\boldsymbol{\delta}_z$ we derived was the worst possible case (i.e. smallest $\boldsymbol{\delta}_z$ which might reach the bound).

□

E.3 β-VAE Optimal Posterior

Theorem 6.2. *For a β-VAE, the optimum posterior is:*

$$q_\phi(\mathbf{z}|\mathbf{x}) \propto p(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z})^{1/\beta}.$$

Proof. Here we use calculus of variations to obtain optimal posteriors for β-VAEs. The objectives we are optimising are over the whole dataset $\mathcal{D} = \{\mathbf{x}_i\}, i = 1, .., N$, with empirical data density $\rho(\mathbf{x}) = \frac{1}{N}\sum_i^N \delta(\mathbf{x} - \mathbf{x}_i)$.

The evidence lower bound for a β-VAE is

$$\mathcal{L}_\beta(\mathcal{D}; \theta, \phi) = \mathbb{E}_{\rho(\mathbf{x})}\Big[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\Big]. \tag{E.24}$$

This is easier to work with written explicitly as integrals. Note that as we are going to be finding the optimal $q_\phi(\mathbf{z}|\mathbf{x})$ we must add a constraint so that it integrates to 1.

$$\begin{aligned}\mathcal{L}_\beta(\mathcal{D}; \theta, \phi) = \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x})\Big[q_\phi(\mathbf{z}|\mathbf{x})[\log p_\theta(\mathbf{x}|\mathbf{z}) - \beta \log q_\phi(\mathbf{z}|\mathbf{x}) + \beta \log p(\mathbf{z})] \\ + \lambda(\mathbf{x})(q_\phi(\mathbf{z}|\mathbf{x}) - 1)\Big]\end{aligned} \tag{E.25}$$

For brevity, going forward $p_\theta(\mathbf{x}|\mathbf{z}) = p$, $p(\mathbf{z}) = \pi$, $q_\phi(\mathbf{z}|\mathbf{x}) = q$. We also view $\mathcal{L}$ as depending on q, p directly.

To proceed with calculus of variations, we substitute $q \to q + \epsilon$, where ϵ is a small function that goes to zero appropriately fast for large $\mathbf{x}, \mathbf{z}$. Thus we expand $\mathcal{L}$ to first order in q to find $\frac{\delta \mathcal{L}}{\delta q}$. The form of q for which this gradient is zero gives us the optimum q for this functional.

$$\mathcal{L}_\beta(q+\epsilon) = \int \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{z}\rho(\mathbf{x})\Big[(q+\epsilon)[\log p - \beta \log(q+\epsilon) + \beta \log \pi] + \lambda(\mathbf{x})(q+\epsilon-1)\Big] \tag{E.26}$$

Recall that $\log(1+x) \approx x$ to first order. Thus $\log(q+\epsilon) \approx \log q + \frac{\epsilon}{q}$ to first order. So,

$$\mathcal{L}_\beta(q+\epsilon) = \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) q \Big[\log p - \beta \log q + \beta \log \pi - \beta \frac{\epsilon}{q} \Big] \tag{E.27}$$

$$+ \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) \epsilon \Big[\log p - \beta \log q + \beta \log \pi - \beta \frac{\epsilon}{q} \Big] \tag{E.28}$$

$$+ \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) \lambda(\mathbf{x})(q-1) + \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) \lambda(\mathbf{x}) \epsilon + \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) O(\epsilon^2). \tag{E.29}$$

Rearranging we find

$$\begin{aligned} \mathcal{L}_\beta(q+\epsilon) &= \mathcal{L}_\beta(q) + \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) \epsilon \Big[\log p - \beta \log q + \beta \log \pi - \beta + \lambda(\mathbf{x}) \Big] \\ &\qquad + \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) O(\epsilon^2) \qquad \text{(E.30)} \\ &= \mathcal{L}_\beta(q) + \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \frac{\delta \mathcal{L}_\beta}{\delta q} \epsilon + \int \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{z}\, \rho(\mathbf{x}) O(\epsilon^2) \qquad \text{(E.31)} \end{aligned}$$

At the optimum value of q the functional will have vanishing functional derivative $\frac{\delta \mathcal{L}_\beta}{\delta q}$, so

$$\log p - \beta \log q + \beta \log \pi - \beta + \lambda(\mathbf{x}) = 0, \tag{E.32}$$

$$\log q = \frac{1}{\beta} \log p + \log \pi + C(\mathbf{x}). \tag{E.33}$$

Exponentiating we find the optimal q to be

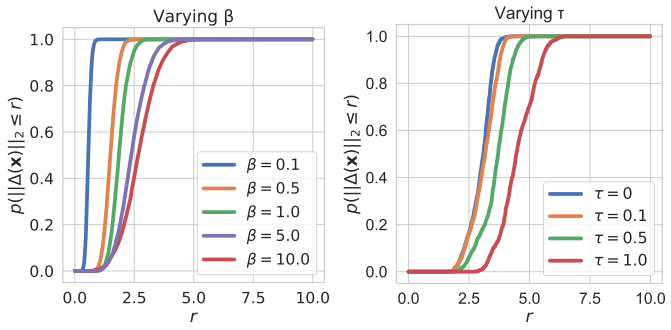
$$q_\phi(\mathbf{z}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} p_\theta(\mathbf{x}|\mathbf{z})^{\frac{1}{\beta}} p(\mathbf{z}), \tag{E.34}$$

where Z is an appropriate normalising constant. This completes the proof. □

E.4 Empirical Calculation of the Bounds

E.4.1 Estimating the minimum r

E.4.1.1 Results



(a) **(b)**

Figure E.2: Here we show that the minimum r for which $p(||\Delta(\mathbf{x})||_2 \leq r) = 0.5$ increases with β and τ, where β is the penalty applied to the D_{KL} in β-VAEs and τ is an offset added to the encoder standard deviation $\boldsymbol{\sigma}_\phi(\mathbf{x})$. This probability, estimated as detailed below in Appendix E.4.1.2, increases with r, but increases more slowly for large β (a) and large τ (b). In such models the encoding process has higher variance resulting in a greater spread of reconstructions, confirming Proposition E.1 in Appendix A that the minimum r for r-robustness increases with the encoder variance.

E.4.1.2 Algorithm

Algorithm E.1: Estimating r

Result: r such that $p(||\Delta(\mathbf{x})||_2 \leq r) > 0.5$
m, *step*, *samples*, $\mathbf{x}$, $r \leftarrow 0$, $p(||\Delta(\mathbf{x})||_2 \leq r) \leftarrow 0$;
while $p(||\Delta(\mathbf{x})||_2 \leq r) < m$ **do**
  $d \leftarrow \{\}$;
  for $i \leftarrow 1$ **to** *samples* **by** 1 **do**
    $\mathbf{s} \sim \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}_\phi(\mathbf{x}))$;
    $s_d \leftarrow ||g_\theta(\mathbf{s}) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))||_2$;
    d.insert(s_d);
  end
  $r \leftarrow r + step$;
  $p(||\Delta(\mathbf{x})||_2 \leq r) \leftarrow \frac{\mathrm{Sum}(d<r)}{nsamples}$;
end

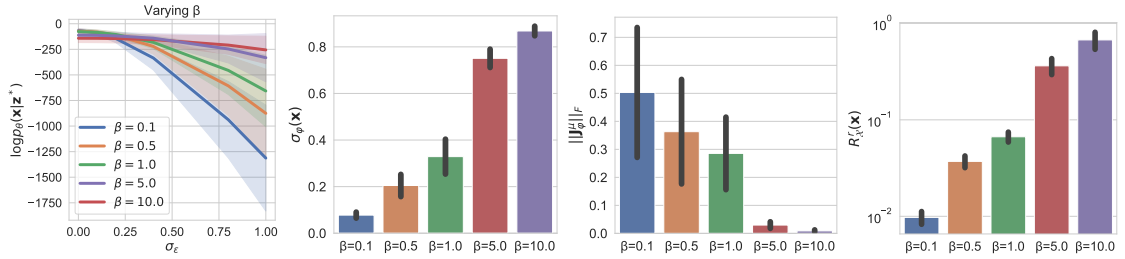
E.4.2 Estimating $R^r_{\mathcal{X}}(\mathbf{x})$

Algorithm E.2: Estimating $R^r_{\mathcal{X}}(\mathbf{x})$

Result: $R^r_{\mathcal{X}}(\mathbf{x})$ such that $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r) > 0.5$
step, *samples*, $\mathbf{x}$, r, $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r) \leftarrow 0$, $R^r_{\mathcal{X}}(\mathbf{x}) \leftarrow 10$, *restarts* $\leftarrow 5$;
while $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r) < 0.5$ **do**
  for $j \leftarrow 1$ **to** *restarts* **by** 1 **do**
    $d \leftarrow \{\}$;
    for $i \leftarrow 1$ **to** *samples* **by** 1 **do**
      $\boldsymbol{\delta}_x \leftarrow$ max damage attack constrained to the norm $R^r_{\mathcal{X}}(\mathbf{x})$;
      $\mathbf{s} \sim \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x} + \boldsymbol{\delta}_x), \boldsymbol{\sigma}_\phi(\mathbf{x} + \boldsymbol{\delta}_x))$
      $s_d \leftarrow ||g_\theta(\mathbf{s}) - g_\theta(\boldsymbol{\mu}_\phi(\mathbf{x}))||_2$;
      d.insert(s_d);
    end
    $R^r_{\mathcal{X}}(\mathbf{x}) \leftarrow R^r_{\mathcal{X}}(\mathbf{x}) - step$;
    $p(||\Delta(\mathbf{x}, \boldsymbol{\delta}_x)||_2 \leq r) \leftarrow \frac{\text{Sum}(d<r)}{nsamples}$;
  end
end

E.5 β-VAE Sensitivity Experiments

(a) $\beta = 0.1$ **(b)** $\beta = 0.5$ **(c)** $\beta = 1$ **(d)** $\beta = 5$ **(e)** $\beta = 10$

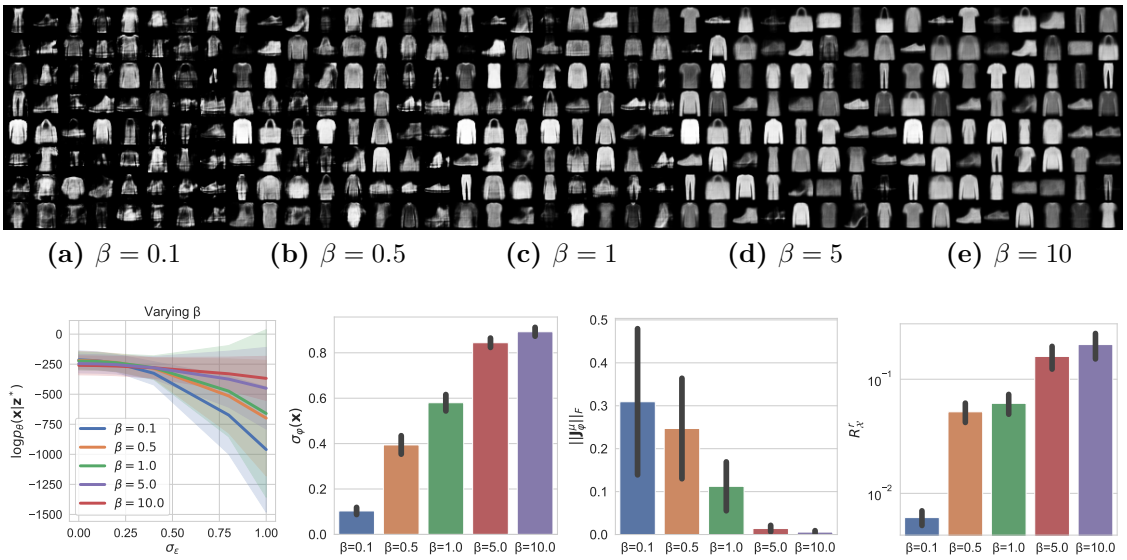


(f) VAE Sensitivity **(g)** Encoder Variance **(h)** Encoder Jacobian **(i)** $R^r_{\mathcal{X}}(\mathbf{x})$ bound

Figure E.3: Here we illustrate that β-VAEs, trained MNIST, with higher β penalties generalise better and are less sensitive to input perturbations.

In Figure E.3, the first row (a)-(e) shows samples drawn from the latent space prior that are then fed through the VAE decoder. It is clear that as β increases, so too does the quality of generated samples. (f) shows the sensitivity of the VAE to input perturbations. We add zero-mean Gaussian noise of variance σ_ϵ^2 to the VAE input to form a noisy input $\mathbf{x}^*$ and embedding $\mathbf{z}*$. We then measure the likelihood of the original point $\mathbf{x}$ under this noisy embedding. σ_ϵ^2 is thus an approximation of the margin of robustness of the VAE, if the VAE's likelihood does not change even for high variance noise, it must have a large margin of robustness ($R^r_{\mathcal{X}}(\mathbf{x})$). The likelihood of $\mathbf{x}$ is quasi constant, under increasing noise variance, for high values of β. This supports our analysis that such models have higher $R^r_{\mathcal{X}}(\mathbf{x})$. Figures (g) and (h) show that the encoder variance and that the norm of the encoder Jacobian ($||\mathbf{J}^{\mu}_{\phi}(\mathbf{x})||_F$) increase as β increases, supporting our analysis that the changes in these values underpin the robustness observed. In (i) we calculate the bound for $R^r_{\mathcal{X}}(\mathbf{x})$ from Theorem 6.1 where we ignore higher order terms. We select r such that $p_{A^r}(\mathbf{x}) = 0.9$, which is a relatively strict metric for robustness. In (f-i) confidence

intervals correspond to the standard deviations of values over the entire MNIST dataset. Taken as a whole these experiments support our analysis that the margin $R^r_{\mathcal{X}}(\mathbf{x})$ increases with β as in Theorem 6.2, in conjuction with the norm of the encoder Jacobian and the encoder variance, supporting Theorem 6.1.



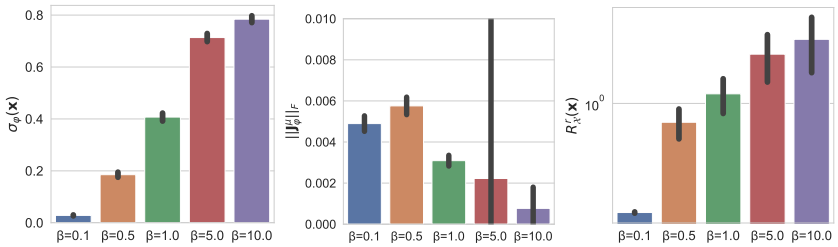
(a) $\beta = 0.1$ **(b)** $\beta = 0.5$ **(c)** $\beta = 1$ **(d)** $\beta = 5$ **(e)** $\beta = 10$

(f) VAE Sensitivity **(g)** Encoder Variance **(h)** Encoder Jacobian **(i)** $R^r_{\mathcal{X}}(\mathbf{x})$ bound

Figure E.4: Here we illustrate that β-VAEs, trained on fashion-MNIST, with higher β penalties generalise better and are less sensitive to input perturbations.

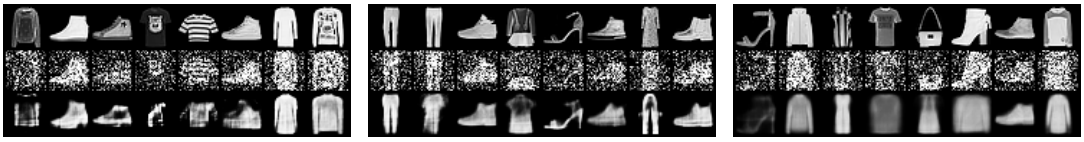
In Figure E.4, the first row in (a)-(e) shows samples drawn from the latent space prior that are then fed through the VAE decoder. It is clear that as β increases, so too does the quality of generated samples. (f) shows the sensitivity of the VAE to input perturbations. We add zero-mean Gaussian noise of variance σ^2_ϵ to the VAE input to form a noisy input $\mathbf{x}^*$ and embedding $\mathbf{z}*$. We then measure the likelihood of the original point $\mathbf{x}$ under this noisy embedding. σ^2_ϵ is thus an approximation of the margin of robustness of the VAE, if the VAE's likelihood does not change even for high variance noise, it must have a large margin of robustness ($R^r_{\mathcal{X}}(\mathbf{x})$). The likelihood of $\mathbf{x}$ is quasi constant, under increasing noise variance, for high values of β. This supports our analysis that such models have higher $R^r_{\mathcal{X}}(\mathbf{x})$. Figures (g) and (h) show that the encoder variance and that the encoder Jacobian norm ($||\mathbf{J}^{\mu}_{\phi}(\mathbf{x})||_F$) increase as β increases, supporting our analysis that the changes in

these values underpin the robustness observed. In (i) we calculate the bound for $R^r_{\mathcal{X}}(\mathbf{x})$ from Theorem 6.1 where we ignore higher order terms. We select r such that $p_{A^r}(\mathbf{x}) = 0.9$, which is a relatively strict metric for robustness. In (f-i) confidence intervals correspond to the standard deviations of values over the entire fashion-MNIST dataset. Taken as a whole these experiments support our analysis that the margin $R^r_{\mathcal{X}}(\mathbf{x})$ increases with β as in Theorem 6.2, in conjuction with the norm of the encoder Jacobian and the encoder variance, supporting Theorem 6.1.



(a) Encoder Variance **(b)** Encoder Jacobian **(c)** $R^r_{\mathcal{X}}(\mathbf{x})$ bound

Figure E.5: Here we illustrate that β-VAEs, trained on CIFAR10, with higher β have larger margins of robustness. Figures (a) and (b) show that the encoder variance and that the encoder Jacobian norm ($||\mathbf{J}^\mu_\phi(\mathbf{x})||_F$) increase as β increases, supporting our analysis that the changes in these values underpin the robustness observed. In (i) we calculate the bound for $R^r_{\mathcal{X}}(\mathbf{x})$ from Theorem 6.1 where we ignore higher order terms. We select r such that $p_{A^r}(\mathbf{x}) = 0.9$, which is a relatively strict metric for robustness. In (a-c) confidence intervals correspond to the standard deviations of values over the entire dataset. Taken as a whole these experiments support our analysis that the margin $R^r_{\mathcal{X}}(\mathbf{x})$ increases with β as in Theorem 6.2, in conjuction with the norm of the encoder Jacobian and the encoder variance, supporting Theorem 6.1.



(a) $\beta = 0.1$ **(b)** $\beta = 1$ **(c)** $\beta = 10$

Figure E.6: We show reconstructions of noisy data for VAEs trained with $\beta \in \{0.1, 1, 10\}$ on Fashion-MNIST. The first row corresponds to the original image, the second to noised a image $\mathbf{x} + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, (0.5^2)\mathbf{I})$. Clearly larger β models are less sensitive to noise, supporting our analysis that increasing β increases the margin of robustness to perturbations.

E.6 Network Hyperparameters

All networks used the same hyperparameters. Namely networks were trained for 100 epochs with the Adam optimizer, with a learning rate of 0.001 and a batch size of 512.

For MNIST and fashion-MNIST networks for the encoder variance and encoder mean were two hidden layer multi-layer perceptrons (MLPs) with 400 units per layer, which shared their first layer. Similarly the decoder was a two layer MLP with 400 units per layer. For these datasets we used a latent space size of 20.

For CIFAR10 we used 4-layer MLPs with 400 units per layer for the encoder and decoder networks and used a 64-dimensional latent space.

Bibliography

Absil, P. A. & Malick, J. (2012). Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22(1), 135–158.

Achlioptas, D. (2003). Database-friendly random projections: Johnson-Lindenstrauss with binary coins. In *Journal of Computer and System Sciences*, volume 66 (pp. 671–687).

Adler, J. & Lunz, S. (2018). Banach Wasserstein GAN. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Akhtar, N. & Mian, A. (2018). Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. *IEEE Access*, 6, 14410–14430.

Alemi, A. A., Fischer, I., Dillon, J. V., & Murphy, K. (2017). Deep Variational Information Bottleneck. In *International Conference on Learning Representations (ICLR)*.

Alemi, A. A., Poole, B., Fischer, I., Dillon, J. V., Saurous, R. A., & Murphy, K. (2018). Fixing a Broken ELBO. In *International Conference on Machine Learning (ICML)*.

Almeida, L. B. (2003). MISEP – Linear and Nonlinear ICA Based on Mutual Information. *Journal of Machine Learning Research*, 4, 1297–1318.

Aubry, M., Maturana, D., Efros, A. A., Russell, B. C., & Sivic, J. (2014). Seeing 3D chairs: Exemplar part-based 2D-3D alignment using a large dataset of CAD models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 3762–3769).

Bachman, P., Devon Hjelm, R., & Buchwalter, W. (2019). Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Bakir, G. H., Gretton, A., Franz, M., & Schölkopf, B. (2004). Multivariate regression via Stiefel manifold constraints. In *Joint Pattern Recognition Symposium* (pp. 262–269).

Barrett, B., Camuto, A., Willetts, M., & Rainforth, T. (2022). Certifiably Robust Variational Autoencoders. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Beal, M. J. (2003). *Variational algorithms for approximate bayesian inference*. PhD thesis, University of Cambridge.

Bell, A. J. & Sejnowski, T. J. (1995). An information-maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7(6), 1004–1034.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 1798–1828.

Beyer, K., Goldstein, J., Ramakrishnan, R., & Shaft, U. (1999). When is "Nearest Neighbor" Meaningful? In *Proceedings of the 7th International Conference on Database Theory*, volume 1540 (pp. 217–235).

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York.

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518), 859–877.

Bornschein, J., Mnih, A., Zoran, D., & Rezende, D. J. (2017). Variational Memory Addressing in Generative Models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., & Bengio, S. (2016). Generating sentences from a continuous space. In *CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., & Amodei, D. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Burda, Y., Grosse, R., & Salakhutdinov, R. (2016). Importance Weighted Autoencoders. In *International Conference on Learning Representations (ICLR)*.

Burel, G. (1992). Blind separation of sources: A nonlinear neural algorithm. *Neural Networks*, 5(6), 937–947.

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., Lerchner, A., & London, D. (2017). Understanding disentangling in beta-VAE. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM J. Sci. Comput.*, 16(5), 1190–1208.

Camuto, A. & Willetts, M. (2022). Variational Autoencoders: A Harmonic Perspective. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Camuto, A., Willetts, M., Paige, B., Holmes, C., & Roberts, S. (2021a). Learning Bijective Feature Maps for Linear ICA. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Camuto, A., Willetts, M., Roberts, S., Holmes, C., & Rainforth, T. (2021b). Towards a Theoretical Understanding of the Robustness of Variational Autoencoders. *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Camuto, A., Willetts, M., Şimşekli, U., Roberts, S., & Holmes, C. (2020). Explicit Regularisation in Gaussian Noise Injections. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Cardoso, J. F. (1989a). Blind identification of independent components with higher-order statistics. In *IEEE Workshop on Higher-Order Spectral Analysis*.

Cardoso, J. F. (1989b). Source separation using higher order moments. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, volume 4 (pp. 2109–2112).

Cardoso, J. F. (1997). Infomax and Maximum Likelihood for Blind Source Separation. *IEEE Letters on Signal Processing*, 4, 112–114.

Cardoso, J. F. & Laheld, B. H. (1996). Equivariant adaptive source separation. *IEEE Transactions on Signal Processing*, 44(12), 3017–3030.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., & Joulin, A. (2020). Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Cayley, A. (1846). Sur quelques propriétés des déterminants gauches. *Journal für die reine und angewandte Mathematik*, 32, 119–123.

Chapelle, O., Schlokopf, B., & Zien, A. (2010). *Semi-Supervised Learning*. MIT Press, 1st edition.

Chen, R., Li, X., Grosse, R., & Duvenaud, D. (2018a). Isolating Sources of Disentanglement in Variational Autoencoders. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020a). A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*.

Chen, X., Fan, H., Girshick, R., & He, K. (2020b). Improved Baselines with Momentum Contrastive Learning. *arXiv preprint*.

Chen, X., Mishra, N., Rohaninejad, M., & Abbeel, P. (2018b). PixelSNAIL: An improved autoregressive generative model. In *International Conference on Machine Learning (ICML)*.

Child, R. (2021). Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. In *International Conference on Learning Representations (ICLR)*.

Choromanski, K., Cheikhi, D., Davis, J., Likhosherstov, V., Nazaret, A., Bahamou, A., Song, X., Akarte, M., Parker-Holder, J., Bergquist, J., Gao, Y., Pacchiano, A., Sarlos, T., Weller, A., & Sindhwani, V. (2020). Stochastic Flows and Geometric Optimization on the Orthogonal Group. In *International Conference on Machine Learning (ICML)*.

Choudrey, R. (2000). *Variational Methods for Bayesian Independent Component Analysis*. PhD thesis, University of Oxford.

Cohen, J., Rosenfeld, E., & Kolter, J. Z. (2019). Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*.

Comon, P. (1994). Independent component analysis, A new concept? *Signal Processing*, 36(3), 287–314.

Corduneanu, A. & Jaakkola, T. (2002). Continuation Methods for Mixing Heterogenous Sources. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

Dai, B. & Wipf, D. (2019). Diagnosing and enhancing VAE models. In *International Conference on Learning Representations (ICLR)*.

Deco, G. & Brauer, W. (1995). Higher Order Statistical Decorrelation without Information Loss. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Dilokthanakul, N., Mediano, P. A. M., Garnelo, M., Lee, M. C. H., Salimbeni, H., Arulkumaran, K., & Shanahan, M. (2017). Deep Unsupervised Clustering with Gaussian Mixture VAE. *arXiv preprint*.

Dinh, L., Krueger, D., & Bengio, Y. (2015). NICE: Non-linear Independent Components Estimation. In *International Conference on Learning Representations (ICLR)*.

Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2017). Density estimation using Real NVP. In *International Conference on Learning Representations (ICLR)*.

Doersch, C. (2016). *Tutorial on Variational Autoencoders*. Technical report, Carnegie Mellon University.

Durkan, C., Bekasov, A., Murray, I., & Papamakarios, G. (2019). Neural Spline Flows. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Esmaeili, B., Wu, H., Jain, S., Bozkurt, A., Siddharth, N., Paige, B., Brooks, D. H., Dy, J., & van de Meent, J.-W. (2019). Structured Disentangled Representations. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Everson, R. & Roberts, S. J. (1999). Independent Component Analysis: A Flexible Nonlinearity and Decorrelating Manifold Approach. *Neural Computation*, 11(8), 1957–83.

Everson, R. & Roberts, S. J. (2001). *Independent Component Analysis*. Cambridge University Press.

Falck, F., Zhang, H., Willetts, M., Nicholson, G., Yah, C., & Holmes, C. (2021). Multi-Facet Clustering Variational Autoencoders. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Fortuin, V., Hüser, M., Locatello, F., Strathmann, H., & Rätsch, G. (2019). SOM-VAE: Interpretable discrete representation learning on time series. In *International Conference on Learning Representations (ICLR)*.

Frey, B. J. & Hinton, G. E. (1996). Free energy coding. In *Data Compression Conference* (pp. 73–81).

Fu, M. C. (2006). Stochastic Gradient Estimation. In S. G. Henderson & B. L. Nelson (Eds.), *Handbooks in Operations Research and Management Science: Simulation*, volume 13 chapter 17.

Ghosh, P., Losalka, A., & Black, M. J. (2019). Resisting Adversarial Attacks Using Gaussian Mixture Variational Autoencoders. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Ghosh, P., Sajjadi, M. S. M., Vergari, A., Black, M., & Schölkopf, B. (2020). From Variational to Deterministic Autoencoders. In *International Conference on Learning Representations (ICLR)*.

Gilmer, J., Adams, R. P., Goodfellow, I., Andersen, D., & Dahl, G. E. (2018). Motivating the Rules of the Game for Adversarial Example Research. *arXiv preprint*.

Golinski, A., Rainforth, T., & Lezcano-Casado, M. (2019). Improving Normalizing Flows via Better Orthogonal Parameterizations. In *ICML Workshop on Invertible Neural Networks and Normalizing Flows*.

Gondim-Ribeiro, G., Tabacof, P., & Valle, E. (2018). Adversarial Attacks on Variational Autoencoders. *arXiv preprint*.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., & Duvenaud, D. (2018). Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations (ICLR)*.

Gregor, K., Besse, F., Rezende, D. J., Danihelka, I., & Wierstra, D. (2016). Towards conceptual compression. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Gresele, L., Rubenstein, P. K., Mehrjou, A., Locatello, F., & Schölkopf, B. (2020). The Incomplete Rosetta Stone Problem: Identifiability results for multi-view nonlinear ICA. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, volume 3.

Ha, D. & Schmidhuber, J. (2018). World Models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Han, K., Vedaldi, A., & Zisserman, A. (2019). Learning to Discover Novel Visual Categories via Deep Transfer Clustering. In *Proceedings of the IEEE International Conference on Computer Vision*.

Harandi, M. & Fernando, B. (2016). Generalized BackPropagation, Etude De Cas: Orthogonality. *arXiv preprint*.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum Contrast for Unsupervised Visual Representation Learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Henaff, O. J., Srinivas, A., Fauw, J. D., Razavi, A., Doersch, C., Eslami, S. M., & Eslami, A. V. O. (2020). Data-Efficient image recognition with contrastive predictive coding. *International Conference on Machine Learning (ICML)*.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., & Lerchner, A. (2017a). β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations (ICLR)*.

Higgins, I., Pal, A., Rusu, A., Matthey, L., Burgess, C., Pritzel, A., Botvinick, M., Blundell, C., & Lerchner, A. (2017b). DARLA: Improving Zero-Shot Transfer in Reinforcement Learning. In *International Conference on Machine Learning (ICML)*.

Hinton, G. (2018). Deep learning-a technology with the potential to transform health care. *Journal of the American Medical Association*, 320(11), 1101–1102.

Hinton, G. E. & Zemel, R. S. (1994). Autoencoders, Minimum Description Length, and Helmholtz Free Energy. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Hoffman, M., Blei, D. M., Wang, C., & Paisley, J. (2012). Stochastic Variational Inference. *Journal of Machine Learning Research*, 14, 1303–1347.

Holmes, C. C. & Walker, S. G. (2017). Assigning a value to a power likelihood in a general Bayesian model. *Biometrika*, 104(2), 497–503.

Hoogeboom, E., Peters, J. W. T., van den Berg, R., & Welling, M. (2019). Integer Discrete Flows and Lossless Compression. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Hsu, Y. C., Lv, Z., & Kira, Z. (2018). Learning to cluster in order to transfer across domains and tasks. In *International Conference on Learning Representations (ICLR)*.

Hsu, Y. C., Lv, Z., Schlosser, J., Odom, P., & Kira, Z. (2019). Multi-class classification without multi-class labels. In *International Conference on Learning Representations (ICLR)*.

Hyvärinen, A., Karhunen, J., & Oja, E. (2001). *Independent Component Analysis*. John Wiley.

Hyvärinen, A. & Oja, E. (1997). A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7), 1483–1492.

Hyvärinen, A. & Pajunen, P. (1999). Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks*, 12(3), 429–439.

Hyvarinen, A., Sasaki, H., & Turner, R. E. (2019). Nonlinear ICA Using Auxiliary Variables and Generalized Contrastive Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1), 79–87.

Jakubovitz, D. & Giryes, R. (2018). Improving DNN robustness to adversarial attacks using jacobian regularization. In *Lecture Notes in Computer Science*, volume 11216 LNCS (pp. 525–541).

Jang, E., Gu, S., & Poole, B. (2017). Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations (ICLR)*.

Jaynes, E. T. (2002). *Probability theory: the logic of science*. Cambridge University Press.

Jiang, Z., Zheng, Y., Tan, H., Tang, B., & Zhou, H. (2017). Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Johnson, W. B. & Lindenstrauss, J. (1984). Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(1), 189–206.

Jutten, C. & Herault, J. (1991). Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24(1), 1–10.

Karhunen, J. (2001). Nonlinear Independent Component Analysis. In R. Everson & S. J. Roberts (Eds.), *ICA: Principles and Practive* (pp. 113–134). Cambridge University Press.

Khemakhem, I., Kingma, D. P., Monti, R. P., & Hyvärinen, A. (2020a). Variational Autoencoders and Nonlinear ICA: A Unifying Framework. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Khemakhem, I., Monti, R. P., Kingma, D. P., & Hyvärinen, A. (2020b). ICE-BeeM: Identifiable conditional energy-based deep models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kilinc, O. & Uysal, I. (2018). Learning Latent Representations in Neural Networks for Clustering Through Pseudo Supervision and Graph-based activity Regularization. In *International Conference on Learning Representations (ICLR)*.

Kim, H. & Mnih, A. (2018). Disentangling by Factorising. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kingma, D. P. & Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kingma, D. P. & Lei Ba, J. (2015). Adam: A Method for Stochastic Optimisation. In *International Conference on Learning Representations (ICLR)*.

Kingma, D. P., Rezende, D. J., Mohamed, S., & Welling, M. (2014). Semi-Supervised Learning with Deep Generative Models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., & Welling, M. (2016). Improved Variational Inference with Inverse Autoregressive Flow. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kingma, D. P. & Welling, M. (2014). Auto-encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*.

Kos, J., Fischer, I., & Song, D. (2018). Adversarial Examples for Generative Models. In *IEEE Security and Privacy Workshops* (pp. 36–42).

Kulkarni, T. D., Whitney, W., Kohli, P., & Tenenbaum, J. B. (2015). Deep Convolutional Inverse Graphics Network. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Kullback, S. & Leibler, R. A. (1951). On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1), 79–86.

Kumar, A. & Poole, B. (2020). On Implicit Regularization in β-VAE. In *International Conference on Machine Learning (ICML)*.

Kusner, M. J., Paige, B., & Miguel Hernández-Lobato, J. (2017). Grammar Variational Autoencoder. In *International Conference on Machine Learning (ICML)*.

Lappalainen, H. & Honkela, A. (2000). Bayesian Non-Linear Independent Component Analysis by Multi-Layer Perceptrons. In M. Girolami (Ed.), *Advances in Independent Component Analysis* (pp. 93–121). Springer.

Lawrence, N. D. & Bishop, C. M. (2000). *Variational Bayesian Independent Component Analysis*. Technical report, University of Cambridge.

LeCun, Y. & Cortes, C. (2010). MNIST handwritten digit database.

Lee, T.-W., Girolami, N., Bell, A. J., & Sejnowski, T. J. (2000). A Unifying Information-Theoretic Framework for Independent Component Analysis. *Computers & Mathematics with Applications*, 39(11), 1–21.

Lee, T. W. & Koehler, B. U. (1997). Blind source separation of nonlinear mixing models. *Neural Networks for Signal Processing - Proceedings of the IEEE Workshop*, (pp. 406–415).

Li, J., Li, F., & Todorovic, S. (2020a). Efficient Riemannian Optimization on the Stiefel Manifold via the Cayley Transform. In *International Conference on Learning Representations (ICLR)*.

Li, J., Zhou, P., Xiong, C., & Hoi, S. C. H. (2020b). Prototypical Contrastive Learning of Unsupervised Representations. In *International Conference on Learning Representations (ICLR)*.

Li, X., Chen, Z., Poon, L. K., & Zhang, N. L. (2019). Learning latent superstructures in variational autoencoders for deep multidimensional clustering. In *International Conference on Learning Representations (ICLR)*.

Liévin, V., Dittadi, A., Maaløe, L., & Winther, O. (2019). Towards Hierarchical Discrete Variational Autoencoders. In *Advances in Approximate Bayesian Inference*.

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Locatello, F., Bauer, S., Lucie, M., Rätsch, G., Gelly, S., Schölkopf, B., & Bachem, O. (2019). Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning (ICML)*, volume 2019-June.

Lucas, J., Tucker, G., Grosse, R., & Norouzi, M. (2019). Don't Blame the ELBO! A Linear VAE Perspective on Posterior Collapse. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Maaløe, L., Fraccaro, M., Liévin, V., & Winther, O. (2019). BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Maaløe, L., Fraccaro, M., & Winther, O. (2017). Semi-Supervised Generation with Cluster-aware Generative Models. *arXiv preprint*.

Maaløe, L., Sønderby, C. K., Sønderby, S. K., & Winther, O. (2016). Auxiliary Deep Generative Models. In *International Conference on Machine Learning (ICML)*.

Mackay, D. J. C. (1995). Developments in Probabilistic Modelling with Neural Networks – Ensemble Learning. In *Proceedings of the 3rd Annual Symposium on Neural Networks* (pp. 191–198). Nijmegen, Netherlands.

Mackay, D. J. C. (1996). *Maximum Likelihood and Covariant Algorithms for Independent Component Analysis*. Technical report, University of Cambridge.

MacKay, D. J. C. (2005). *Information Theory, Inference, and Learning Algorithms*. Cambridge, UK: CUP, 7.2 edition.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics* (pp. 281–297).: University of California Press.

Maddison, C. J., Mnih, A., & Teh, Y. W. (2017). The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations (ICLR)*.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., & Frey, B. (2016). Adversarial Autoencoders. In *International Conference on Learning Representations (ICLR)*.

Mathieu, E., Rainforth, T., Siddharth, N., & Teh, Y. W. (2019). Disentangling Disentanglement in Variational Autoencoders. In *International Conference on Machine Learning (ICML)*.

Matthey, L., Higgins, I., Hassabis, D., & Lerchner, A. (2017). dSprites: Disentanglement testing Sprites dataset.

Miller, J. W. & Dunson, D. B. (2019). Robust Bayesian Inference via Coarsening. *Journal of the American Statistical Association*, 114(527), 1113–1125.

Misra, I. & van der Maaten, L. (2020). Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 6706–6716).

Moosavi-Dezfooli, S. M., Fawzi, A., Fawzi, O., & Frossard, P. (2017). Universal adversarial perturbations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*.

Nalisnick, E., Hertel, L., & Smyth, P. (2016). Approximate Inference for Deep Latent Gaussian Mixtures. In *NeurIPS Bayesian Deep Learning Workshop*.

Nalisnick, E., Matsukawa, A., Teh, Y. W., & Lakshminarayanan, B. (2019). Detecting Out-of-Distribution Inputs to Deep Generative Models Using Typicality. *arXiv preprint*.

Neyman, J. & Pearson, E. S. (1933). On the Problem of the Most Efficient Tests of Statistical Hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231, 289–337.

Nguyen, A. T. & Raff, E. (2019). Adversarial Attacks, Regression, and Numerical Stability Regularization. In *The AAAI Workshop on Engineering Dependable and Secure Machine Learning Systems*.

Nigam, K., Mccallum, A. K., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2), 103–134.

Nordhaus, W. D. (2007). Two centuries of productivity growth in computing. *Journal of Economic History*, 67(1), 128–159.

Noroozi, M. & Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9910 LNCS, 69–84.

Oliver, A., Odena, A., Raffel, C., Cubuk, E. D., & Goodfellow, I. J. (2018). Realistic Evaluation of Deep Semi-Supervised Learning Algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Owen, A. B. (2013). *Monte Carlo theory, methods and examples*.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2019). *Normalizing Flows for Probabilistic Modeling and Inference*. Technical report, DeepMind, London, UK.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017). Practical black-box attacks against machine learning. In *ASIA CCS 2017 - Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security*.

Parra, L., Deco, G., & Miesbach, S. (1995). Redundancy reduction with information-preserving nonlinear maps. *Network: Computation in Neural Systems*, 6(1), 61–72.

Parra, L., Deco, G., & Miesbach, S. (1996). Statistical Independence and Novelty Detection with Information Preserving Nonlinear Maps. *Neural Computation*, 8(2), 260–269.

Paysan, P., Knothe, R., Amberg, B., Romdhani, S., & Vetter, T. (2009). A 3D face model for pose and illumination invariant face recognition. In *6th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2009* (pp. 296–301).

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*.

Pervez, A., Cohen, T., & Gavves, E. (2020). Low Bias Low Variance Gradient Estimates for Hierarchical Boolean Stochastic Networks. In *International Conference on Machine Learning (ICML)*.

Rainforth, T., Cornish, R., Yang, H., Warrington, A., & Wood, F. (2018). On nesting Monte Carlo estimators. In *International Conference on Machine Learning (ICML)*.

Rao, D., Visin, F., Rusu, A. A., Teh, Y. W., Pascanu, R., & Hadsell, R. (2019). Continual Unsupervised Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Razavi, A., van den Oord, A., & Vinyals, O. (2019a). Generating Diverse High-Fidelity Images with VQ-VAE-2. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Razavi, A., Vinyals, O., Van Den Oord, A., & Poole, B. (2019b). Preventing posterior collapse with δ-VAES. In *International Conference on Learning Representations (ICLR)*.

Rezende, D. J. & Mohamed, S. (2015). Variational Inference with Normalizing Flows. In *International Conference on Machine Learning (ICML)*.

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conference on Machine Learning (ICML)*.

Rezende, D. J. & Viola, F. (2018). Taming VAEs. *arXiv preprint*.

Roberts, S. J. (1998). Independent Component Analysis: Source Assessment & Separation, a Bayesian Approach. *IEEE Proceedings-Vision, Image and Signal Processing*, 145(3), 149–154.

Roeder, G., Metz, L., & Kingma, D. P. (2021). On Linear Identifiability of Learned Representations. In *International Conference on Machine Learning (ICML)*.

Rolinek, M., Zietlow, D., & Martius, G. (2019). Variational autoencoders pursue pca directions (by accident). In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June (pp. 12398–12407).

Sack, J. R. & Urrutia, J., Eds. (2000). *Handbook of Computational Geometry*. NLD: North-Holland Publishing Co.

Salimans, T., Karpathy, A., Chen, X., & Kingma, D. P. (2017). PixelCNN++: Improving the PixelCNN with discretized logistic mixture likelihood and other modifications. In *International Conference on Learning Representations (ICLR)*.

Schott, L., Rauber, J., Bethge, M., & Brendel, W. (2019). Toward the First Adversarially Robust Neural Network Model on MNIST. In *International Conference on Learning Representations (ICLR)*.

Shamir, A., Safran, I., Ronen, E., & Dunkelman, O. (2019). A Simple Explanation for the Existence of Adversarial Examples with Small Hamming Distance. *arXiv preprint*.

Shu, R. (2016). Gaussian Mixture VAE: Lessons in Variational Inference, Generative Models, and Deep Nets.

Siddharth, N., Paige, B., Van De Meent, J. W., Desmaison, A., Goodman, N. D., Kohli, P., Wood, F., & Torr, P. H. (2017). Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Siegel, J. W. (2021). Accelerated Optimization With Orthogonality Constraints. *Journal of Computational Mathematics*, 39(2), 207–226.

Sohn, K., Berthelot, D., Li, C. L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., & Raffel, C. (2020). FixMatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Sokolić, J., Giryes, R., Sapiro, G., & Rodrigues, M. R. (2017). Generalization error of invariant classifiers. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Sønderby, C. K., Poole, B., & Mnih, A. (2017). Continuous Relaxation Training of Discrete Latent Variable Image Models. In *NeurIPS Bayesian Deep Learning Workshop*.

Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., & Winther, O. (2016). Ladder Variational Autoencoders. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Springenberg, J. T. (2016). Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks. In *International Conference on Learning Representations (ICLR)*.

Stiefel, E. (1935). Richtungsfelder und Fernparallelismus in n-dimensionalen Mannigfaltigkeiten. *Commentarii mathematici Helvetici*, 8, 305–353.

Stisen, A., Blunck, H., Bhattacharya, S., Prentow, T. S., Kjærgaard, M. B., Dey, A., Sonne, T., & Jensen, M. M. (2015). Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems* (pp. 127–140).

Stühmer, J., Turner, R. E., & Nowozin, S. (2020). Independent Subspace Analysis for Unsupervised Learning of Disentangled Representations. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Sutton, R. S. (2019). The Bitter Lesson.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. *arXiv preprint*.

Tabacof, P., Tavares, J., & Valle, E. (2016). Adversarial Images for Variational Autoencoders. In *NeurIPS Workshop on Adversarial Training*.

Taleb, A. (2002). A generic framework for blind source separation in structured nonlinear models. *IEEE Transactions on Signal Processing*, 50(8), 1819–1830.

Theis, L., Shi, W., Cunningham, A., & Huszár, F. (2017). Lossy Image Compression with Compressive Autoencoders. In *International Conference on Learning Representations (ICLR)*.

Titsias, M. & Lázaro-Gredilla, M. (2014). Doubly Stochastic Variational Bayes for non-Conjugate Inference. *International Conference on Machine Learning (ICML)*.

Townsend, J., Bird, T., & Barber, D. (2019). Practical Lossless Compression with Latent Variables using Bits Back Coding. In *International Conference on Learning Representations (ICLR)*.

Townsend, J., Bird, T., Kunze, J., & Barber, D. (2020). HiLLoC: Lossless Image Compression with Hierarchical Latent Variable Models. In *International Conference on Learning Representations (ICLR)*.

Tran, D., Vafa, K., Agrawal, K. K., Dinh, L., & Poole, B. (2019). Discrete flows: Invertible generative models of discrete data. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Uesato, J., O'Donoghue, B., Van Den Oord, A., & Kohli, P. (2018). Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning (ICML)*, volume 11.

Vahdat, A. & Kautz, J. (2020). NVAE: A Deep Hierarchical Variational Autoencoder. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Valpola, H., Oja, E., Ilin, A., Honkela, A., & Karhunen, J. (2003). Nonlinear blind source separation by variational Bayesian learning. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E86-A(3), 532–541.

Van Den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester normalizing flows for variational inference. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., & Kavukcuoglu, K. (2016). Conditional Image Generation with PixelCNN Decoders. In *Advances in Neural Information Processing Systems (NeurIPS)*.

van den Oord, A., Li, Y., & Vinyals, O. (2018). Representation Learning with Contrastive Predictive Coding. *arXiv preprint*.

van den Oord, A., Vinyals, O., & Kavukcuoglu, K. (2017). Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Watanabe, S. (1960). Information Theoretical Analysis of Multivariate Correlation. *IBM Journal of Research and Development*, 4(1), 66–82.

Waterhouse, S., MacKay, D., & Robinson, A. (1996). Bayesian methods for mixtures of experts. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. D. (2016). A survey of transfer learning. *Journal of Big Data*, 3(1).

Wenzel, F., Roth, K., Veeling, B. S., Świątkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., & Nowozin, S. (2020). How Good is the Bayes Posterior in Deep Neural Networks Really? *arXiv preprint*.

Willetts, M., Camuto, A., Rainforth, T., Roberts, S., & Holmes, C. (2021a). Improving VAEs' Robustness to Adversarial Attack. In *International Conference on Learning Representations (ICLR)*.

Willetts, M., Miscouridou, X., Roberts, S., & Holmes, C. (2021b). Relaxed-Responsibility Hierarchical Discrete VAEs. In *NeurIPS Bayesian Deep Learning Workshop*.

Willetts, M. & Paige, B. (2021). I don't need u: Identifiable non-linear ica without side information. *arXiv preprint*.

Willetts, M., Roberts, S., & Holmes, C. (2019). Disentangling to Cluster: Gaussian Mixture Variational Ladder Autoencoders. In *NeurIPS Bayesian Deep Learning Workshop*.

Willetts, M., Roberts, S., & Holmes, C. (2020). Semi-Unsupervised Learning: Clustering and Classifying using Ultra-Sparse Labels. In *2020 IEEE International Conference on Big Data*.

Williams, W., Ringer, S., Ash, T., Hughes, J., MacLeod, D., & Dougherty, J. (2020). Hierarchical Quantized Autoencoders. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Woodruff, D. P. (2014). Sketching as a Tool for Numerical Linear Algebra. *Foundations and Trends in Theoretical Computer Science*, 10(2), 1–157.

Xiao, H., Rasul, K., & Vollgraf, R. (2017). *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. Technical report.

Xie, Q., Dai, Z., Hovy, E., Luong, M. T., & Le, Q. V. (2020). Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xu, W., Sun, H., Deng, C., & Tan, Y. (2017). Variational Autoencoder for Semi-supervised Text Classification. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Yang, H. H., Amari, S. I., & Cichocki, A. (1998). Information-theoretic approach to blind separation of sources in non-linear mixture. *Signal Processing*, 64(3), 291–300.

Yang, J., Parikh, D., Batra, D., & Tech, V. (2016). Joint Unsupervised Learning of Deep Representations and Image Clusters. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yellott, J. I. (1977). The relationship between luce's choice axiom, thurstone's theory of comparative judgment, and the double exponential distribution. *Journal of Mathematical Psychology*, 15(2), 109–144.

Zhai, X., Oliver, A., Kolesnikov, A., & Beyer, L. (2019). S4L: Self-supervised semi-supervised learning. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1476–1485).

Zhao, S., Song, J., & Ermon, S. (2017). Learning Hierarchical Features from Generative Models. In *International Conference on Machine Learning (ICML)*.