Optimal transport for Latent variable models

Benoit Gaujac

Department of Computer Science

University College London

This dissertation is submitted for the degree of

Doctor of Philosophy

Declaration

I, Benoit Gaujac, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Generative models are probabilistic models which aim at approximating the process by which a given dataset is generated. They are well suited to the unsupervised learning setup and constitute intuitive and powerful models providing an interpretable representation of the data. However, learning even simple generative models can be challenging in the traditional Maximum Likelihood framework, due to the inflexibility of the training objective used. Motivated by its topological properties, we will show in this thesis how methods based on Optimal Transport can overcome these difficulties and offer competitive alternatives. Firstly, we show that training generative models that combine both discrete and continuous latent variables can be significantly more effective when using Optimal Transport methods. Such intuitive models are highly motivated by the structure of many real-world datasets but remain hard to train with the most common likelihood-based method, often resulting in the collapse of the discrete latent variables. Secondly, we propose a novel approach based on Optimal Transport to training models with fully Markovian deep-latent hierarchies. Probabilistic models with deep latent-variable structures have powerful modelling capacity, but common approaches often fail to leverage deep-latent hierarchies without complex inference and optimisation schemes. Our method successfully leverages the whole hierarchy of the models and shows competitive generative performance while learning smooth latent manifolds through every layers of the latent hierarchy. Finally, we introduce a new training objective to improve the learning of interpretable and disentangled representation of the data. Our method achieves competitive disentanglement relative to state-of-the-art techniques whilst improving the reconstruction and generation performances of the models.

Impact statement

The research presented in this thesis has great potential to benefit both the academia world and commercial organizations. Much of the work has been published at major machine learning conferences [Gaujac et al., 2021a,b].

In this thesis, we investigated the advantages of using Optimal-Transport-based methods to train generative models. It contributed to further deepen our understanding of the limits of the classic likelihood-based methods in settings that have recently seen a renew of interest by the academic community. Especially, the methods presented in the subsequent chapters will likely open the doors to alternative approaches in areas such as discrete latent modeling, deep-hierarchical latent-modeling, and representation and disentanglement learning.

Outside academia, a growing number of commercial organisations make use of deep generative modeling to analyse and interpret their data, making the understanding of current methods and the development of alternatives very valuable. The omnipresence of discrete data in real world applications makes the method proposed in Chapter 2 likely beneficial for the industry with a better modeling of the data. Many commercial applications also rely on powerful models learned from huge amounts of data without supervision. These applications will only be successful if they offer competitive performances at low computational costs, highlighting the benefits of the approach introduced in Chapter 3. Alongside the growing adoption of generative models in commercial applications, organisations have seen an increasing scrutiny over the implication of such models in our daily life. In Chapter 4 we developed a method that offers more interpretable representation with more control over the way information is encoded.

Contents

List of Figures

List of Tables

Notations

Notations

- $\mathbb{Z}^\star$: $\mathbb{Z}/\{0\}$. In this thesis, we only consider $\mathbb{Z} = \mathbb{N}$ or $\mathbb{Z} = \mathbb{R}$.
- $\mathbb{R}_+$: Positive real numbers.
- $\mathbf{0}_d$, $\mathbf{1}_d$: null and unit vector on $\mathbb{R}^d$.
- $\mathbf{I}_d$: identity matrix of size $d \times d$.
- $\mathrm{diag}(u)$: $d \times d$ matrix whose diagonal is u and zero elsewhere, where $u \in \mathbb{R}^d$.
- $u^\top$: transpose of u.
- $\cdot$: canonical dot product.
- $\odot$: element-wise multiplication.
- $\|\cdot\|_{L_m}$: L_m norm, $m \in \mathbb{N}^\star$, on the Euclidean space $\mathbb{R}^d$.
- δ_{x_0}: Dirac centered at x_0.
- Ω: probability space.
- ω: element of Ω.
- μ: reference Lebesgue measure.
- Calligraphic letters refer to sets, *e.g.* $\mathcal{X}$
- Capital letters refer to random variable on Ω, *e.g.* $X : \Omega \to \mathcal{X}$.

- $\perp$: independence for random variables.
- Lower case letters refer to the realisation of random variables, *e.g.* $x = X(\omega)$.
- $\mathcal{P}(\mathcal{X})$: set of probability distributions on $\mathcal{X}$.
- $\mathcal{P}(\mathcal{X} \times \mathcal{Y})$: set of join probability distributions on $\mathcal{X} \times \mathcal{Y}$.
- $\mathcal{P}(P, Q)$: set of couplings with marginal $P \in \mathcal{P}(\mathcal{X})$ and $Q \in \mathcal{P}(\mathcal{Y})$ respectively.
- $P(X)$: distribution of X.
- $p(x)$: density of X *w.r.t.* μ.
- $P(X|Z)$: distribution of X conditioned on Z.
- $P(X|Z = z)$: distribution of X conditioned on the realisation $Z = z$. Also denoted as $P(X|z)$.
- $p(x|Z = z)$: density of X conditioned on the realisation $Z = z$. Also denoted as $p(x|z)$.

Introduction

Understanding the intrinsic structure of a given dataset of interest is at the center of the machine learning community effort. Indeed, capturing this information is key to answering most questions that arise when studying datasets composed of random variables, and for tackling challenges that emerge when working with real-world data. Machine learning approaches can be broadly divided in three categories: supervised learning, unsupervised learning and reinforcement learning.

Supervised learning methods use both the data and the desired output to learn the conditional distribution of the outputs given the inputs. Supervised learning methods were the first ones to present human-level performances in common tasks such as classification and regression. Since, they have been used in wide range of applications such as image classification, object detection, machine translation and speech recognition. Two important models for supervised learning are convolutional neural networks (CNNs) and recurrent neural networks (RNNs). CNNs [Krizhevsky et al., 2012, Simonyan and Zisserman, 2015, He et al., 2016] are a type of deep neural networks that are specifically designed for processing data that can be represented on a grid, such as images or videos. They are obtained by stacking convolutional layers, each layer capturing patterns on different scales. RNNs are a type of deep neural network that are specifically designed for processing data that has a sequential structure, such as text or audio. RNNs work by learning to store information from previous inputs in a recurrent hidden state vector h, and then using this information to make predictions about future input sequences. As with CNNs, they are built by stacking specifically designed layers [Hochreiter and Schmidhuber, 1997, Cho et al., 2014] that can learn to extract information from the previously generated inputs and use it to both generate the next set of inputs and pass down the extracted

information to the next layers. Recently, transformer models such as Devlin et al. [2018] have shown human-level performances on a wide variety of tasks.

In reinforcement learning, an agent interacts with a known or unknown environment in order to maximize a cumulative reward. The environment provides a feedback for each action taken by the agent allowing the agent to learn the optimal set of actions for this specific environment to achieve maximal reward. Applications of reinforcement learning algorithms vary and often involve complex interaction between the agent and the environment such as games, robotic control and autonomous driving.We can distinguish 3 main families of reinforcement learning methods. First, policy gradient methods are a type of deep reinforcement learning methods that directly optimize the policy of an agent. The policy of an agent is a function that maps from the internal state of the agent to the action taken by agent. Policy gradient methods Williams [1992] work by estimating the gradient of the expected reward with respect to the policy, and then using this gradient to update the policy. Policy gradient methods are relatively simple to implement, and they can be effective in a variety of domains. However, they can be computationally expensive, and they can be difficult to train for large and complex problems. Value-based methods [Mnih et al., 2013] are another type of deep reinforcement learning method that estimates the value of a state or a state-action pair. The value of a state is the expected reward that an agent can expect to receive from that state, given its policy. Value-based methods work by estimating the value function, and then using this estimate to make decisions. Value-based methods are typically more efficient than policy gradient methods, and they can be more effective for large and complex problems. However, they can be more difficult to train, and they can be less robust to noise. Advantage actor-critic methods [Mnih et al., 2016, Schulman et al., 2017] are a type of deep reinforcement learning methods that combines the advantages of policy gradient methods and value-based methods. Advantage actor-critic methods estimate the advantage function, which is the difference between the value of a state-action pair and the value of the current state. The advantage function is used to update the policy, and the value function is used to make decisions. Advantage actor-critic methods are typically more efficient than policy gradient methods, and they can

be more effective for large and complex problems. They are also more robust to noise than value-based methods. In reinforcement learning, an agent interacts with a known or unknown environment in order to maximize a cumulative reward. The environment provides a feedback for each action taken by the agent allowing the agent to learn the optimal set of actions for this specific environment to achieve maximal reward. Applications of reinforcement learning algorithms are varied and often involve complex interaction between the agent and the environment such as games, robotic control and autonomous driving.We can distinguish 3 main families of reinforcement learning methods. First, policy gradient methods are a type of deep reinforcement learning methods that directly optimize the policy of an agent. The policy of an agent is a function that maps from the internal state of the agent to the action taken by agent. Policy gradient methods Williams [1992] work by estimating the gradient of the expected reward with respect to the policy, and then using this gradient to update the policy. Policy gradient methods are relatively simple to implement, and they can be effective in a variety of domains. However, they can be computationally expensive, and they can be difficult to train for large and complex problems. Value-based methods [Mnih et al., 2013] are another type of deep reinforcement learning method that estimates the value of a state or a state-action pair. The value of a state is the expected reward that an agent can expect to receive from that state, given its policy. Value-based methods work by estimating the value function, and then using this estimate to make decisions. Value-based methods are typically more efficient than policy gradient methods, and they can be more effective for large and complex problems. However, they can be more difficult to train, and they can be less robust to noise. Advantage actor-critic methods [Mnih et al., 2016, Schulman et al., 2017] are a type of deep reinforcement learning methods that combines the advantages of policy gradient methods and value-based methods. Advantage actor-critic methods estimate the advantage function, which is the difference between the value of a state-action pair and the value of the current state. The advantage function is used to update the policy, and the value function is used to make decisions. Advantage actor-critic methods are typically more efficient than policy gradient methods, and they can be more effective for large and complex

problems. They are also more robust to noise than value-based methods.

Finally, and most relevant for this thesis, unsupervised learning methods have only access to the data without any guidance on the desired outputs as opposed to supervised learning. Since no explicit information is given to the model, unsupervised learning tasks are often considered more challenging than their supervised counterparts. Historically, these methods have been popular for tasks involving summarizing and explaining the data such as clustering or density estimation. More recently, using these methods to learn the process by which a set of data has been generated has been at the center of the machine learning community effort. Indeed, it not only allows for the generation of new data but also helps to capture and understand the intrinsic structure of the dataset at hand. The generative models introduced by these methods have been widely used in areas where large unlabelled dataset are available, from Natural Language Processing (NLP) and image generation to image in-painting and representation learning. Amongst the different type of generative models, we can find:

- Variational Autoencoders (VAEs) [Kingma and Welling, 2014, Rezende et al., 2014] are probabilistic models that are trained to efficiently encode the data information into a compressed internal or latent representation. In order to learn the encoder distribution, VAEs also learn to reconstruct the data from the latent representation using a decoder network.

- Generative Adversarial Networks (GANs) [Radford et al., 2016, Brock et al., 2019] consist of two neural networks, a generator and a discriminator. The generator is responsible for creating new data samples, while the discriminator is responsible for distinguishing between real and fake data samples. The two networks are trained adversarially, meaning that they compete against each other. The generator tries to create data samples that are so realistic that the discriminator cannot distinguish them from real data samples. The discriminator, on the other hand, tries to become better at distinguishing between real and fake data samples. While GANs have been shown to be very effective at generating realistic data samples, but they can be difficult to train.

GANs can also be unstable, meaning that they can sometimes generate data samples that are not very realistic.

- Autoregressive models [Van den Oord et al., 2016c,a, Radford et al., 2018] are a type of deep generative model that generates data one sample at a time. Each sample is generated based on the previous samples. Autoregressive models can be used to generate text, music, and other types of data. Autoregressive models can be slow to generate data points and difficult to scale to large datasets.

- Flow-based generative models [Dinh et al., 2014, 2016b, Kingma and Dhariwal, 2018] are a type of deep generative model that uses normalizing flows to transform the latent distribution to the data distribution. The latent distribution is typically a simple distribution, such as a Gaussian distribution and the generative flow transforms the latent distribution into the data distribution a using a series of invertible transformations. While very expressive, Flow-based models can be computationally expensive to train.

- Diffusion-based models [Ho et al., 2020, Song et al., 2021, Dhariwal and Nichol, 2022] are a type of deep generative models that generates data by gradually denoising an initial random sample. Diffusion models work by recursively add noise to the available training data (also known as the forward diffusion process) and then reversing the process (known as denoising or the reverse diffusion process) to recover the data. Diffusion-based models are typically more efficient at large scale than both flow-based and autoregressive models.

In this work, we assume that the data are noisy observations of hidden random variables that live in a low-dimensional space. These models are often referred to as latent-variable models. The support of the hidden variables, or the latent manifold, can be seen as a compact and efficient representation of the observed data. The representations encoded in the latent manifold can be used for many downstream tasks such as semantic classification or data visualisation. However, finding the optimal model that matches the data generation process is hard, if not impossible, and often requires approximations. Maximum likelihood (ML) have been

designed to train parametric latent-variable models by maximizing the likelihood of the observations under the model distribution. Amongst these, VAE [Kingma and Welling, 2014, Rezende et al., 2014] is a popular method that maximizes the Evidence Lower Bound (ELBO) of the model likelihood. Both VAEs and variants have shown impressive results. Yet it remains intrinsically ill-defined and inflexible. Recently, Optimal Transport (OT) has been suggested as a likelihood-free method showing competitive performances with appealing properties that could overcome the difficulties encountered by likelihood-based methods.

The main challenges and problems tackled in this work are central questions in the machine learning community. Training powerful generative models is a common challenge in the unsupervised learning setting as no explicit information on the desired behavior is given to the model. For example, while latent-variable models with discrete latent variable are ones of the most simple models to describe dataset split into a known number of discrete classes, their training is challenging when using the common ML approach. A second aspect we looked at was the models themselves, and especially how to improve the expressiveness and interpretability of generative models. One solution to increase the models expressiveness is to increase the depth of their latent variables. For example, models with deep-latent hierarchies are believed to be able to capture a powerful representation of the data as each latent layer of the hierarchy can encode the information at different scale. Another approach would be to introduce explicit control on the way the latent variables encode the data. For example, encouraging disentanglement of the latent variable has been show to produce more robust and semantically meaningful latent representations.

In Chapter 1, we introduce the foundations of generative modelling. We first present the theory underpinning probabilistic generative modelling, latent-variable models and the popular ML approach, focusing on VAE [Kingma and Welling, 2014, Rezende et al., 2014]. We then give an overview of the OT problem, starting with the discrete case before moving on to the continuous case which is of more interest in this work. Finally, we review the recently introduced Wasserstein Autoencoder (WAE) [Tolstikhin et al., 2018, Bousquet et al., 2017]. This is a likelihood-free method based on the OT framework that shows interesting geometric properties and promising

results when training latent-variable models.

In Chapter 2 we quantitatively show how WAEs are appealing alternatives to VAEs when working with discrete-latent models in a fully unsupervised setting. Generative models with both discrete and continuous latent variables, such as Gaussian mixture latent-variable models (GL-LVMs), have long been of interest for the machine learning community because of the sheer number of real-world applications with discrete classes in the data. However, classic approximate variational-inference methods which rely on gradient descent algorithms are not suited to the discrete setting. For example, it has been well documented that the modes of the discrete-latent variables collapse. Solutions to this include using specially designed models [Jang et al., 2017, Maddison et al., 2017, Van den Oord et al., 2017, Johnson et al., 2016] and training methods [Eslami et al., 2016, Lawson et al., 2018]. Another alternative is to simply add back some supervision by introducing a subset of labelled data [Kingma et al., 2014]. Motivated by the weak topology induced by the Wasserstein distance on the space of the model distributions, we show that WAEs are better suited to learning discrete-latent models than the likelihood-based VAEs. Specifically, we show that GM-LVMs can be trained with WAE in an unsupervised fashion with qualitatively better latent representation, motivating further the value of the OT approach to generative modelling [Gaujac et al., 2021a].

In Chapter 3, we present StWAE, a new training objective specially designed for deep-latent hierarchical generative models. Such models are highly expressive models that aim to encode the data with a latent hierarchy. Their deep-latent structures are believed to provide the practitioner with an intuitive and explainable latent representation while improving at the same time the generative performance of the model. While increasing the depth of the latent hierarchy has long been used as a means to increase the expressiveness of generative models, training latent-variable models with deep latent hierarchy remains challenging. State-of-the-art methods rely on highly tailored inference and generative network designs [Sønderby et al., 2016, Bachman, 2016, Kingma et al., 2016, Vahdat and Kautz, 2020] as well as complex training schemes [Kingma et al., 2016, Maaløe et al., 2019, Yoshida and Miyato, 2017, Vahdat and Kautz, 2020]. Instead, StWAE is an OT-based approach, building

on the WAE of Tolstikhin et al. [2018], Bousquet et al. [2017] for learning expressive deep-latent hierarchical models. StWAE is derived by stacking WAEs in each latent layer, introducing an inference distribution at each level of the hierarchy that maps the information up to the next layer. We quantitatively show that by doing so, StWAE is able to leverage all of its hierarchy and to encode the information up to its deepest latent layer. Moreover, the simple first order Markovian structure of StWAE reduces the computational cost of the model whilst providing a qualitatively more interpretable latent representation.

While StWAE was focused on training deep-latent hierarchical generative models, it showed promising capabilities at learning interpretable latent representations. In Chapter 4, we introduce the Total Correlation Wasserstein Autoencoder (TCWAE), an OT-based method for learning disentangle latent representations. Disentanglement learning seeks to learn a semantically meaningful representation of the data by using statistically independent latent variables, each encoding for a unique underlying generative factor Bengio et al. [2013]. Not only do such methods improve the interpretability of the learned representation but also offer more control over how the model encodes information. Additionally, it is argued that disentangled latent representations improve the performance and robustness of the model Bengio et al. [2013], van Steenkiste et al. [2019]. By leveraging the decomposition of the Kullback-Leibler (KL) divergence, TCWAE presents an explicit dependency on the Total Correlation of the aggregated posterior, which has been shown to play an important role in the disentanglement of the latent representation.

We conclude with a review of the contributions of this thesis and discuss the potential direction for future work.

Chapter 1

Foundations

In this chapter, we lay out the background knowledge and technical concepts needed to understand the remainder of the dissertation. We start by introducing generative modelling and the Maximum Likelihood principle, providing an intuitive method to train such generative models. We then present the Optimal Transport problem and introduce an Optimal Transport based framework for learning generative models that offers a powerful alternative to the likelihood approach.

1.1 Generative models

Generative models are probabilistic models that are trained to replicate the process by which a set of data is generated. To do so, one assumes that the data at hand, or observations, are *i.i.d.* realisations of a random variable X, defined on probability space Ω, living in $\mathcal{X}$, with probability distribution P_X. In most of the cases, it is assumed that P_X is continuous *w.r.t.* the Lebesgue reference measure μ with its density denoted $p_X = \frac{dP_X}{d\mu}$. However, this data distribution is generally unknown, with only a sub-set of variables observable and only a finite set of samples available. The goal is then to learn the data distribution by *approximating* it with a model or likelihood distribution $P \approx P_X$. This is achieved by defining a divergence or loss function, $\mathcal{L}$, on the probability space and finding the optimal element $P^\star$ in the

model distribution space $\mathcal{P}(\mathcal{X})$ that minimizes this loss:

$$P^\star = \arg\min_{P \in \mathcal{P}(\mathcal{X})} \mathcal{L}\big(P_X, P\big) \tag{1.1}$$

Often, the model distributions are parametric functions. We will denote the model parameters by θ for the remainder of this work. Only deterministic parameters are being considered in this thesis, as opposed to the Bayesian approach where parameters are themselves random variables. The objective of generative modelling boils down in this case to learning the parameters $\theta^\star$ that result in the best approximation of the data distribution:

$$\theta^\star = \arg\min_{\theta} \mathcal{L}\big(P_X, P_\theta\big) \tag{1.2}$$

1.1.1 Latent variable models

Latent variable models are a class of probabilistic models where only a sub-set of variables is observed. The likelihood distribution then refers to the marginal of the joint distribution, $P(X, Z)$ over the observed variables X. It is implicitly defined by integrating or *marginalizing* over the latent variables:

$$p(x) = \int_{\mathcal{Z}} p(x, z)\, dz = \int_{\mathcal{Z}} p(x|z)p(z)\, dz \tag{1.3}$$

The distribution $P(Z)$ is referred to as the prior distribution whilst the conditional distribution $P(X|Z)$ as the generative model. Even though the integral in Equation (1.3) is intractable in many cases, it is often easy to sample from the generative model. Assuming one can sample easily from both the prior and the conditional generative model, sampling from the model can be straightforward by first sampling z from the prior and then x from the generative model conditioned on the sampled z.

When working with latent variable models, one can only hope to learn the marginal of the joint model distribution over the observed variables as the ground truth data distribution is only accessible through samples, and in many cases only a finite set of samples.

Often, the observed variables are high dimensional whilst the latent space lies in a low dimension space. The latent variables can be seen as a compact and efficient representation of the high dimensional observations. The generative model $P(X|Z)$ maps the latent variables to the observations whilst the posterior, $P(Z|X)$, provides the inverse mapping. In some applications, latent variables are used to extract semantically meaningful information about the observed data, for example by encoding the factors of variation in a statistically independent manner (see Chapter 4).

1.1.2 Maximum Likelihood

A popular method for training latent variable models is the Maximum Likelihood (ML) approach. In the ML approach, the loss function $\mathcal{L}$ in Equation (1.1) is the Kullback-Liebler (KL) divergence. The optimization problem defined in Equation (1.2) is then equivalent to maximizing the loglikelihood of the model $P(X)$:

$$\begin{aligned}\min_{P\in\mathcal{P}(\mathcal{X})} \mathbf{KL}\big(P_X \parallel P\big) &= \min_{P\in\mathcal{P}(\mathcal{X})} \mathop{\mathbb{E}}_{P_X}\Big(\log\frac{p_X(X)}{p(X)}\Big)\\ &= \min_{P\in\mathcal{P}(\mathcal{X})} \mathop{\mathbb{E}}_{P_X}\Big(\log p_X(X)\Big) - \mathop{\mathbb{E}}_{P_X}\Big(\log p(X)\Big)\\ &= \max_{P\in\mathcal{P}(\mathcal{X})} \mathop{\mathbb{E}}_{P_X}\Big(\log p(X)\Big) \qquad (1.4)\end{aligned}$$

In the case of finite samples $x_1,\dots,x_M \overset{\text{i.i.d}}{\sim} P_X$, with parametric model P_θ, the maximization problem (1.4) becomes:

$$\theta^{ML} \overset{\text{def}}{=} \arg\max_\theta \frac{1}{M}\sum_m \log p_\theta(x_m) \qquad (1.5)$$

where θ^{MLE} is the Maximum Likelihood Estimator (MLE).

In Information Theory, the KL divergence between two distributions P_1 and P_2 represents the average number of extra bits required to encode samples coming from P_1 when using P_2. In other words, the ML approach seeks to find the model distribution that requires the less amount of extra bits to encode the data. An equivalent interpretation of the ML method can be drawn from the last line of Equation (1.4):

the ML maximises the negative cross-entropy of the model distribution under the data distribution. The cross-entropy quantifies the number of bits needed when encoding the data with the model distribution instead of the data distribution [Kraft, 1949, McMillan, 1956].

The ML approach is not always feasible as the integral in Equation (1.4) might not be tractable, making the likelihood impossible or hard to compute. Moreover, even in the case where one can approximate the likelihood term (using sampling methods for example), it is still possible that no closed-form solutions exist for Equation (1.4). In this case, one solution is to optimize a tractable lower-bound of the likelihood. One popular bound is the *variational free energy* function or *Evidence Lower-Bound* (ELBO) defined, for $x \in \mathcal{X}$, by:

$$\mathcal{L}_{ELBO}\big(x, P_X, Q\big) = \int_{\mathcal{Z}} q(z|x) \log \frac{p(x|z)p(z)}{q(z|x)} \, dz \tag{1.6}$$

where p is the model distribution. The density q introduced in Equation 1.6 is often named the *variational posterior* or *variational distribution*. Using, Jensen's inequality, it is easy to show that $\mathcal{L}_{ELBO}\big(x, P_X, Q\big) \leq \log p_x(x)$ for all $x \in \mathcal{X}$ and all densities p_x and q.

Expectation-Maximisation The Expectation-Maximisation (EM) algorithm [Dempster et al., 1977] is an example of such methods. EM iteratively increases the log-likelihood of the model by updating the generative distribution $P(X|Z)$, during the E-step, and the variational distribution $Q(Z|X)$, in the M-step. In the case of parametric models, P_θ, and variational distributions, Q_ϕ, the EM alternates between the E-step and the M-step as follow:

- **E-step**: maximise $\mathcal{L}_{ELBO}(\theta, \phi)$ with respect to ϕ whilst holding θ fixed.
- **M-step**: maximise $\mathcal{L}_{ELBO}(\theta, \phi)$ with respect to θ whilst holding ϕ fixed.

Variational Autoencoder The Variational Autoencoder (VAE) [Kingma and Welling, 2014, Rezende et al., 2014] is a popular method for training generative models that is relatively straightforward with specific choices of variational distributions and

priors. It relies on simple Monte-Carlo estimators (MC) combined with expressive encoder and decoder distributions parameterised by powerful neural networks. In VAEs, the ELBO is decomposed as:

$$\mathcal{L}_{ELBO}(x, \theta, \phi) = \int_{\mathcal{Z}} q_\phi(z|x) \log p_\theta(x|z)\ dz - \mathbf{KL}\big(Q_\phi(Z|x) \parallel P(Z)\big) \tag{1.7}$$

The first term of the *r.h.s.* of Equation(1.7) is referred to as the reconstruction cost whilst the second term as the latent regularizer. The reconstruction term penalizes models that poorly reconstruct the observations whilst the latent regularizer ensures that the posterior does not drift dramatically away from the prior.

Assuming that $\log p_\theta(x|z)$ is well defined and that it is possible to sample z from q_ϕ, we can easily compute a MC estimate of the reconstruction term. Additionally, if we can compute a closed-form expression of the KL term in the *r.h.s.* of Equation(1.7), we can obtain an unbiased estimator of the ELBO. Then, with continuous and differentiable parametrizations of the model and variational distributions, we can use gradient descent methods to maximize the ELBO *w.r.t.* to the parameters θ and ϕ:

$$\nabla_{\theta,\phi}\ \mathcal{L}_{ELBO}(x, \theta, \phi) = \frac{1}{M} \sum_{m=1}^{M} \nabla_{\theta,\phi}\ \log p_\theta(x|z_m) - \nabla_\phi\ \mathbf{KL}\big(Q_\phi(Z|x) \parallel P(Z)\big) \tag{1.8}$$

where $z_m \sim q_\phi(z|x)$ for $m = 1, \ldots, M$. However, it has been shown [Paisley et al., 2012] that the gradient of the MC estimate of the reconstruction term with respect to ϕ can be of high variance.

Kingma and Welling [2014] introduced the reparametrization trick. They show that, under mild conditions on the variational distribution, it is possible to reparametrize the distribution $q_\phi(.|x)$ with a differentiable transformation of a fixed auxiliary noise variable ϵ as follow:

$$\begin{aligned} \epsilon &\sim p(\epsilon) \\ z &= g_\phi(x, \epsilon) \end{aligned} \tag{1.9}$$

where g_ϕ is a differentiable function *w.r.t.* ϕ.

Example 1.1.1 *A common choice suggested by Kingma and Welling [2014] is to parameterize the approximate posterior using a Gaussian distribution with mean* μ_ϕ *and covariance* $\text{diag}(\sigma_\phi^2)$ *where* μ_ϕ *and* σ_ϕ^2 *are neural networks with parameters* ϕ*.* $q_\phi(\cdot|x)$ *can then be reparametrized using a unit normal distribution:*

$$\begin{aligned} \epsilon &\sim \mathcal{N}(\mathbf{0}_{d_\mathcal{Z}}, \mathbf{I}_{d_\mathcal{Z}}) \\ g_\phi(x, \epsilon) &= \mu_\phi(x) + \sigma_\phi^2(x) \odot \epsilon \end{aligned} \tag{1.10}$$

It follows that $z = g_\phi(x, \epsilon) \sim \mathcal{N}\big(\mu_\phi(x), \text{diag}(\sigma_\phi^2(x))\big)$*.*

Combined with stochastic gradient methods, VAEs prove to be relatively simple to train and have shown impressive results in term of samples generation and loglikelihood score. Since its introduction, there have been a growing number of methods, introducing more complex graphical models, more powerful neural networks and tailored loss functions, that gradually closed the performance gap with other approaches, showing impressive sample quality and achieving state-of-the-art likelihood score.

Generative Adversarial Networks Generative Adversarial Networks (GANs) [Goodfellow et al., 2014] are another family of generative models that have shown impressive performance at learning complex data distributions and generating realistic samples. Similarly to VAEs, GANs use a generative distribution or generator to generate samples providing with a mapping from the latent variables to the observations. As in VAEs, the mean parameters of the generator are usually parametrized by neural networks. However, as opposed to VAEs, GANs do not use any encoder network, and thus do not provide the practitioner with the invert mapping from the observation space to the latent space. Instead, a discriminator is trained simultaneously to the generator to distinguish true data from samples generated by the generator. In doing so, the discriminator guides the generator to generator more realistic samples. More formally, the generator G is trained to minimizing the discrimination loss of the discriminator D while the discriminator, D, learns to maximize its discriminative

power. Mathematically, the training objective is a min-max problem defined by:

$$\min_G \max_D \mathbb{E}_{P_X}\big[\log D(x)\big] + \mathbb{E}_{P_Z}\Big[\log\big(1 - D\big(G(z)\big)\big)\Big] \tag{1.11}$$

where again, P_X is the true data distribution while P_Z is the prior distribution on the latent space $\mathcal{Z}$.

GANs have shown good generative performances, with impressive samples quality, often out-performing VAEs. Many follow-up works have built on the original paper, improving the generative performances and introducing modification of the training objective to guide the generation process [Salimans et al., 2016]. However, the lack of implicit likelihood function and encoder to map from the observation to the latent space makes GANs unsuitable in many applications. Moreover, GANs are known to be prone to mode collapsing during training [Salimans et al., 2016], making the use of very powerful networks for the generator and discriminator much more difficult than with VAEs.

Flow-based models Normalizing flows [Dinh et al., 2014] have been proposed to alleviate some of the problems encountered in both VAEs and GANs. Mostly, they try to directly learn the data distribution as opposed and thus optimize directly loglikelihood of the model as opposed to GANs and VAEs whose training objectives are only proxy of the data likelihood. Flow-based models rely on invertible transformations of the model distribution, transforming a simple distribution into complex ones by sequentially applying a sequence of invertible functions to the model distribution. More formally, we can define the normalizing flows f_i, $i = 0, \ldots, N$ as follow:

$$z_0 \sim p_0,\ z_i = f_i(z_{i-1}),\ i = 1, \ldots, N \tag{1.12}$$

where $z_N = x$. Using the change of variable theorem, we can easily compute the density of the intermediate variables z_i from Equation (1.12):

$$p_i(z_i) = p_{i-1}(f_i^{-1}(z_i))\left|det\frac{df_i^{-1}(z_i)}{dz_i}\right| \tag{1.13}$$

Thus, the model loglikelihood can be written as:

$$\log p_X(x) = \log p_0(z_0) - \sum_{n=1}^{N} \log \left| det \frac{df_i(z_{i-1})}{dz_{i-1}} \right| \tag{1.14}$$

In order to make Equation (1.14) tractable, the flows f_i must have a tractable inverse function and their Jacobian must also be tractable. Several works have refined the transformation functions with invertible layers such as additive coupling layers [Dinh et al., 2014] and affine coupling layers [Dinh et al., 2016b] and designed simple invertible operations such as invertible 1x1 convolutions [Kingma and Dhariwal, 2018]. Another family of flow-based models framed the transformations in the normalizing flow as autoregressive generative models, where is dimension of the observations is conditioned on a set of previous dimensions for a given dimension ordering. These models with autoregressive flows have been used in a variety of task, from simple density estimation to image generation [Van den Oord et al., 2016b] and audio signals [Van den Oord et al., 2016a]. Finally, and interesting in the context of this thesis, normalizing flows have been used to improve the flexibility and complexity of the prior distribution in VAEs [Rezende and Mohamed, 2015]. Specifically, the simple prior distribution is sequentially transform into a more complex one by applying a sequence of invertible flows which are designed to make the latent regularizer in Equation (1.7) tractable. While flow-based models have been shown to achieve impressive performance in term of likelihood score and sample generation, their training requires heavy computation power due to the inverse operation at inference time making these method hard to scale to bigger dimensions.

Diffusion based models Diffusion models have emerged recently as a powerful generative modeling technique, building on Denoising Diffusion Probabilistic Models (DDPM) Sohl-Dickstein et al. [2015b]. The key intuition is to recast data generation as a denoising process. Specifically, a Markov chain is used to gradually add Gaussian noise to data samples x_0, diffusing them into pure noise x_T Ho et al. [2020]. The generative model $p_\theta(x_{t-1}|x_t)$ aims to reverse this diffusion by modeling the conditional distribution to denoise x_t back to x_{t-1}.

The model p_θ is trained by maximizing the log likelihood $\log p(x_0)$ under this forward diffusion-reverse denoising process. The resulting evidence lower bound (ELBO) objective enables tractable training and encourages p_θ to reverse the diffusion Song et al. [2021]. After training, ancestral sampling generates new x_0 by iterative sampling from p_θ starting from pure noise x_T.

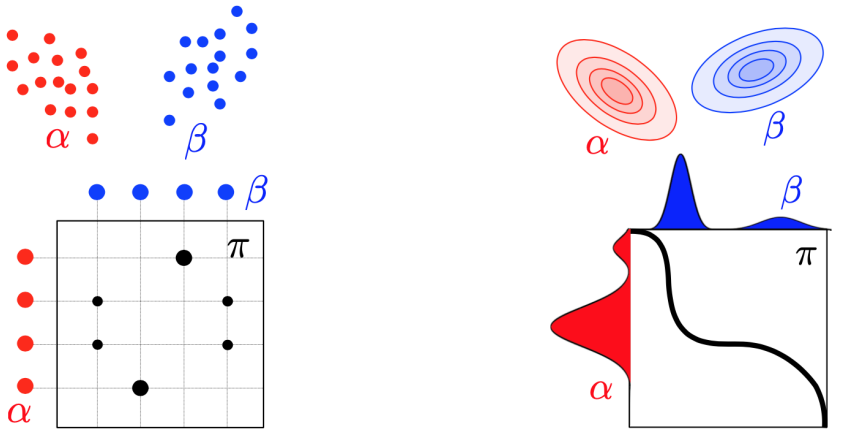
Key advantages of diffusion models include the ability to leverage highly flexible neural networks for p_θ to model complex, high-dimensional densities Nichol et al. [2022]. This helps capture intricate distributional details and generate remarkably realistic samples. However, a current limitation is the computational cost of repeated denoising during sampling. Overall, diffusion models are becoming highly competitive for generative modeling, and ongoing work to improve sampling efficiency may further enhance their capabilities.

1.2 Optimal Transport

The Optimal Transport (OT) problem finds its origin in the problem of resources allocation [Monge, 1781, Kantorovich, 2006]. In the seminal work of Monge [1781], the author considers piles of sand and how to move sand from a set of different locations, the *sources*, to a set of destinations, the *targets* given a unit moving cost. The OT is defined as the transport with the smallest possible cost. More formally, OT is a mathematical framework providing geometric tools to compare probability distributions. It defines a natural geometrical structure on the ground space, paving the way for key concepts when studying probability distributions. We start by introducing the OT problem for continuous random variables. We then focus on the special case of the Wasserstein distance and finish with a discussion on the potential advantage of using the OT framework for generative modelling.

1.2.1 Continuous Optimal Transport

The OT problem was first introduced in the case of discrete random variables, and more especially as a metric in for discrete probability distributions. Recent advances [Cuturi, 2013b] allowed practitioners to move away from linear program solvers and



(a) Discrete measures. (b) Continuous measures.

Figure 1.1: Illustration of the Optimal Transport problem for two measures α and β with the resulting transport plan given by π. (a): discrete OT. (b): continuous OT.

made the OT problem more appealing in a number of applications such as in image [Ortega et al., 2017, Wang et al., 2013, 2011, Thorpe et al., 2017] and text [Rolet et al., 2016, Huang et al., 2016, Kolouri et al., 2017] processing. Indeed, images and texts can be represented as histograms on the pixels grid and words vocabulary respectively, thus lending themselves well to the formalism of the OT problem (see AppendixA.0.1 for more details on the discrete OT problem).

In this case, we consider continuous probability distributions. We define the set of couplings $\mathcal{P}(P,Q)$, between $P \in \mathcal{P}(\mathcal{X})$ and $Q \in \mathcal{P}(\mathcal{Y})$, as:

$$\mathcal{P}(P,Q) \stackrel{\text{def}}{=} \left\{ \Gamma \in \mathcal{P}(\mathcal{X} \times \mathcal{Y}); \int_{\mathcal{Y}} \gamma(x,y)\, dy = p(x) \quad \text{and} \quad \int_{\mathcal{X}} \gamma(x,y)\, dx = q(y) \right\} \tag{1.15}$$

Similarly to the discrete case, the continuous OT problem seeks to minimize the total cost of transporting a unit volume from $P(x)$ to $Q(y)$ given the unit cost $c(x,y)$. Indeed, for a given coupling Γ between P and Q, the infinitesimal cost is given by $\delta(x,y) = c(x,y)\gamma(x,y)\, dxdy$. This result in the discrete OT formulation given by:

$$\begin{aligned} \mathcal{L}_c(P,Q) &\stackrel{\text{def}}{=} \min_{\Gamma \in \mathcal{P}(P,Q)} \int_{\mathcal{X}\times\mathcal{Y}} c(x,y)\gamma(x,y)\, dxdy \\ &= \min_{\Gamma \in \mathcal{P}(P,Q)} \mathop{\mathbb{E}}_{\Gamma(X,Y)} c(X,Y) \end{aligned} \tag{1.16}$$

where the second line provides a more probabilistic interpretation of the OT (see Figure 1.1b). The optimal coupling $\Gamma^\star$ such that:

$$\begin{aligned}\Gamma^\star &= \min_{\Gamma \in \mathcal{P}(P,Q)} \int_{\mathcal{X}\times\mathcal{Y}} c(x,y)\gamma(x,y)\, dxdy \\ &= \arg\min_{\Gamma \in \mathcal{P}(P,Q)} \mathop{\mathbb{E}}_{\Gamma(X,Y)} c(X,Y)\end{aligned} \tag{1.17}$$

is also referred to as the optimal transport plan between P and Q for the cost function c.

Equation (1.16) is an infinite-dimensional linear programming problem and under mild conditions[1], it can be shown that it always has a solution. For a more thorough review of the properties and algorithmic methods of the OT we refer the readers to Peyré and Cuturi [2019].

1.2.2 Wasserstein distance

One important property of the OT is that it defines a metric on the probability distributions space as long as the *ground cost* function, c in Equation (1.16), that satisfies certain conditions.

A function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ is a distance on $\mathcal{X}$ if it is symmetric, null *i.i.f.* $x = y$ and satisfies the triangle inequality:

$$\begin{aligned}&(i) : \forall (x,y) \in \mathcal{X}^2,\ d(x,y) = d(y,x) \\ &(ii) : \forall (x,y) \in \mathcal{X}^2,\ d(x,y) > 0 \quad \text{and} \quad d(x,y) = 0 \Leftrightarrow x = y \\ &(iii) : \forall (x,y,z) \in \mathcal{X}^3,\ d(x,y) \leq d(x,z) + d(z,y)\end{aligned} \tag{1.18}$$

We now assume $\mathcal{Y} = \mathcal{X}$ and consider a ground cost function c of the form $c(x,y) = d(x,y)^l$ where d is a distance on $\mathcal{X} \times \mathcal{X}$ and $l \in \mathbb{N}^\star$. Then, for $(P,Q) \in \mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{X})$, the l-Wasserstein distance, $\mathcal{W}_{c,l}$, between P and Q is defined as:

$$\mathcal{W}_{c,l}\big(P,Q\big) \stackrel{\text{def}}{=} \mathcal{L}_c\big(P,Q\big)^{\frac{1}{l}} \tag{1.19}$$

[1] $\mathcal{X}$ and $\mathcal{Y}$ compact spaces and c continuous.

or using Equation (1.16):

$$\mathcal{W}_{c,l}\big(P,Q\big)^l = \min_{\Gamma\in\mathcal{P}(P,Q)} \int_{\mathcal{X}\times\mathcal{X}} d(x,y)^l \gamma(x,y)\, dxdy \tag{1.20}$$

$\mathcal{W}_{c,l}$ is a distance function on $\mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{X})$. Especially, and contrary to the KL divergence used in the ML approach, it satisfies properties (i) and (iii) of Definition 1.18:

$$\begin{aligned} &(i) : \forall (P_1, P_2) \in \mathcal{P}(\mathcal{X})^2,\ \mathcal{W}_{c,l}\big(P_1, P_2\big) = \mathcal{W}_{c,l}\big(P_2, P_1\big) \\ &(iii) : \forall (P_1, P_2, P_3) \in \mathcal{P}(\mathcal{X})^3,\ \mathcal{W}_{c,l}\big(P_1, P_3\big) \leq \mathcal{W}_{c,l}\big(P_1, P_2\big) + \mathcal{W}_{c,l}\big(P_2, P_3\big) \end{aligned} \tag{1.21}$$

The Wasserstein distance naturally *lifts* the distance δ on the probability space Ω to the space of distributions on that *ground* space:

$$\big(\mathcal{X}, \delta\big) \to \big(\mathcal{P}(\mathcal{X}), \mathcal{W}_{c,l}\big) \tag{1.22}$$

Wasserstein distance for generative models As we saw in the previous sections, generative modelling involves finding the distribution, or set of parameters in the case of parametric models, that minimize a loss function $\mathcal{L}$. This optimization problem could be solved with gradient descent methods if $\mathcal{L}$ was continuous *w.r.t.* the model parameters θ. Now, the continuity of $\mathcal{L}$ is equivalent to having a continuous mapping $\theta \mapsto P_\theta$. Indeed, given a distance δ on the space of distributions and assuming that $\theta_\infty = \theta^\star$ and $P_{\theta^\star} = P_{\text{data}}$, the mapping $\theta \mapsto P_\theta$ is by definition continuous in $\theta^\star$ if:

$$\theta_m \underset{m\to\infty}{\to} \theta^\star \ \Leftrightarrow\ \delta(P_{\theta_m}, P_{\text{data}}) \underset{m\to\infty}{\to} 0 \tag{1.23}$$

Thus, the choice of distance on the distributions space, or loss function in the case of generative modelling, will characterize the continuity of the models P_θ *w.r.t.* the parameters.

Now, it can be shown that (see Appendix A.0.2):

$$\mathbf{KL}\big(P_m \parallel P_\infty\big) \underset{m\to\infty}{\to} 0 \;\Rightarrow\; \delta_{TV}(P_m, P_\infty) \underset{m\to\infty}{\to} 0 \;\Rightarrow\; \mathcal{W}_{c,l}\big(P_m, P_\infty\big) \underset{m\to\infty}{\to} 0 \quad (1.24)$$

Consequently, it should be easier to find the global minima of Equation (1.16) than to find the ones of Equation (1.4). Example 1.2.1 illustrates these differences in the simple case of learning parallel lines living in a 2-dimensional space. Especially, we can see that, in this specific case, both the TV distance and the KL divergence saturate, making gradient descent over the distribution parameter θ impossible. On the opposite, the Wasserstein distance defines a smooth function of the model parameter, which can be easily optimized.

Example 1.2.1 *Lets consider $Z \sim \mathcal{U}[0,1]$ a uniform random variable in $[0,1]$ and define P_0 the distribution of $X = (0, Z)$ and P_θ the distribution of $Y = (\theta, Z)$, with $(P_0, P_\theta) \in \mathcal{P}(\mathbb{R}^2) \times \mathcal{P}(\mathbb{R}^2)$. Then, it can easily be shown that:*

- $d_{TV}\big(P_\theta, P_0\big) = \begin{cases} 1 & \text{if} \quad \theta \neq 0 \\ 0 & \text{else.} \end{cases}$
- $\boldsymbol{KL}\big(P_\theta \parallel P_0\big) = \begin{cases} +\infty & \text{if} \quad \theta \neq 0 \\ 0 & \text{else.} \end{cases}$
- $\mathcal{W}_{\|\cdot\|_{L_m},l}\big(P_\theta, P_0\big) = |\theta|, \quad \forall (m,l) \in \mathbb{N}^2$

Thus, $\theta \mapsto d_{TV}\big(P_\theta, P_0\big)$ is a piece-wise constant function with zero gradient everywhere whilst $\theta \mapsto \boldsymbol{KL}\big(P_\theta \parallel P_0\big)$ is simply ill-defined on $\mathbb{R}^\star$. On the contrary, $\theta \mapsto \mathcal{W}_{\|\cdot\|_{L_m},l}\big(P_\theta, P_0\big)$ is continuous and differentiable and gradient descent methods can easily be used to find the optimal distribution parameter.

While Example 1.2.1 only considers very simple 2-dimensional distributions, similar observations can be made for more general multivariate distributions when their supports intercept in a set of measure null. Particularly, this is often the case with latent variable models where one assumes that the data live in a low dimensional manifold (see Section 1.1.1). Note that other divergences than the ones

of Example 1.2.1 are able to tackle distributions with non-overlapping supports. For example, the Spread Divergence considers noisy versions of the data and model distributions [Zhang et al.] while the Maximum Mean Discrepancy (MMD) leverages the properties of reproducing kernel Hilbert spaces [Gretton et al., 2005].

1.3 Wasserstein Autoencoder

1.3.1 Formulation

Motivated by the observation on the topology induced by the Wasserstein distance made in Section 1.2.1 , Bousquet et al. [2017] and Tolstikhin et al. [2018] introduced the Wasserstein Autoencoder (WAE). WAE optimizes the Wasserstein distance between the unknown data distribution and the model distribution by making two approximations of the Wasserstein distance: i) they refactorize the set of couplings in Equation (1.20) and ii) they relax the hard constraint on the marginal. By reparametrizing the set of the couplings, the authors upper-bound the Wasserstein distance by constraining the optimization space to distributions of the form described in Equation (1.3). The relaxation of the marginal constraint turns the non-convex optimization problem defined in Equation 1.20 into an easier unconstrained convex objective.

Refactorization of the joint distribution More formally, assume $P_X(X) \in \mathcal{P}(\mathcal{X})$, $P_Z(Z) \in \mathcal{P}(\mathcal{Z})$, $P_Y(Y|Z) \in \mathcal{P}(\mathcal{X})$. With abuse of notation, we also refer to $P_Y(Y)$ as the distribution on $\mathcal{X}$ such that Z is first sampled from P_Z while Y is then sampled from $P_Y(Y|Z)$ using the previously sampled z: $p_Y(y) = \int_{\mathcal{Z}} p_Y(y|z)p_Z(z)\, dz$. We consider the set of joint distributions on X, Y and Z, $\mathcal{P}_{X,Y,Z}$, such that $\forall P \in \mathcal{P}_{X,Y,Z}$:

$$p(y,z) \overset{\text{def}}{=} \int_{\mathcal{X}} p(x,y,z)\, dx = p_Y(y|z)p_Z(z) \tag{1.25}$$

$$p(x) \overset{\text{def}}{=} \int_{\mathcal{Y}\times\mathcal{Z}} p(x,y,z)\, dydz = p_X(x) \tag{1.26}$$

$$(X \perp Y) \mid Z \tag{1.27}$$

From Equation (1.25), it is immediate that $p(z) \overset{\text{def}}{=} \int_{\mathcal{X}\times\mathcal{Y}} p(x,y,z)\,dxdy = p_Z(z)$ and $p(y|z) \overset{\text{def}}{=} \frac{p(y,z)}{p(z)} = p_Y(y|z)$. And so, $P \in \mathcal{P}_{X,Y,Z}$ *i.f.f.* there exists $P(Z|X) \in \mathcal{P}(\mathcal{Z})$ such that:

$$P(X,Y,Z) = P_X(X)P(Z|X)P_Y(Y|Z) \tag{1.28}$$

and

$$p_Z^{agg}(z) \overset{\text{def}}{=} \int_{\mathcal{X}} p_X(x)p(z|x)\,dx = p_Z(z)$$

We denote by $\mathcal{P}_{X,Y}$ the set of marginals on (X,Y) induced by distributions in $\mathcal{P}_{X,Y,Z}$. By definition, we have $\mathcal{P}_{X,Y} \subset \mathcal{P}(P_X, P_Y)$. Thus we can upper-bound the Wasserstein distance between P_X and P_Y as follow:

$$\mathcal{W}_{c,l}\big(P_X,P_Y\big)^l \leq \inf_{P\in\mathcal{P}_{X,Y}} \mathop{\mathbb{E}}_{P(X,Y)} c\big(X,Y\big) \overset{\text{def}}{=} \mathcal{W}^{\dagger}_{c,l}\big(P_X,P_Y\big)^l \tag{1.29}$$

Using Equation (1.28), we have:

$$\mathcal{W}^{\dagger}_{c,l}\big(P_X,P_Y\big)^l = \inf_{\substack{P(Z|X)\in\mathcal{P}(\mathcal{Z}) \\ p_Z^{agg}=p_Z}} \mathop{\mathbb{E}}_{P_X(X)} \mathop{\mathbb{E}}_{P(Z|X)} \mathop{\mathbb{E}}_{P_Y(Y|Z)} c\big(X,Y\big) \tag{1.30}$$

where $p_Z^{agg}(z)$ is defined in Equation (1.28). Mirroring the ML approach (see Equation (1.6)), the encoding distribution $P(Z|X)$ is often referred to as the *approximate variational* posterior and denoted $Q(Z|X)$, whilst P_Z^{agg} as the *aggregated* posterior and often denoted $Q_Z(Z) = \int_{\mathcal{X}} p_X(x)Q(Z|X=x)\,dx$. In the reminder of the thesis, we will adopt these notations.

As shown in Tolstikhin et al. [2018], the inequality in Equation (1.29) becomes an equality in the case of deterministic decoder $P_\theta(Y|Z) = \delta_{G_\theta(Z)}(Y)$ where $G : \mathcal{Z} \to \mathcal{X}$ is a deterministic mapping (see Bousquet et al. [2017] for a detailed proof).

Relaxing the marginal constraint The marginal constraint in Equation (1.28) makes the objective defined in Equation (1.30) a non-convex optimization problem. Tolstikhin et al. [2018] relaxed Equation (1.30) with a convex penalty resulting in

the unconstrained objective:

$$\mathcal{W}_{c,l}^{\lambda}\big(P_X, P_Y\big)^l \stackrel{\text{def}}{=} \inf_{Q(Z|X)\in\mathcal{P}_Y} \mathop{\mathbb{E}}_{P_X(X)} \mathop{\mathbb{E}}_{Q(Z|X)} \mathop{\mathbb{E}}_{P_Y(Y|Z)} c\big(X, Y\big) + \lambda \cdot \mathcal{D}\big(Q_Z, P_Z\big) \tag{1.31}$$

where $\mathcal{D} : \mathcal{P}_Z \times \mathcal{P}_Z \to \mathbb{R}_+$ is a convex function *w.r.t.* Q such that $\mathcal{D}\big(Q, P\big) = 0 \Leftrightarrow P = Q$. Under mild conditions [Borwein and Lewis, 2000], the soft constraint of Equation (1.31) is equivalent to adding a hard constraint of the type $\mathcal{D}\big(Q_Z, P_Z\big) < \zeta_\lambda$ where $\zeta_\lambda > 0$ is a decreasing function of λ such that $\zeta_\lambda \to 0$ when $\lambda \to \infty$. Thus, when λ increases, the solutions of Equation (1.31) will reach the feasible region of Equation (1.30) where $Q_Z = P_Z$.

Parametric WAE In the case of generative modelling, X represents the true observations, Y the model samples and Z the latent variables: $P_Y(Y) = \int_{\mathcal{Z}} P_\theta(Y|Z = x)p_Z(z)\, dz$. As with VAEs, the encoding distribution $Q(Z|X)$ is parameterized by ϕ: $Q(Z|X) = Q_\phi(Z|X)$ and Tolstikhin et al. [2018], Bousquet et al. [2017] reformulate Equation (1.31) as a min-min optimization problem over θ and ϕ:

$$\widetilde{\mathcal{W}}_c^{\lambda}\big(\theta, \phi\big) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{P_X(X)} \mathop{\mathbb{E}}_{Q_\phi(Z|X)} \mathop{\mathbb{E}}_{P_\theta(Y|Z)} c\big(X, Y\big) + \lambda \cdot \mathcal{D}\big(Q_{Z,\phi}, P_Z\big) \tag{1.32}$$

Note that for simplicity, Tolstikhin et al. [2018], Bousquet et al. [2017] directly optimize the l-power of $\mathcal{W}_{c,l}^{\lambda}$ instead of the l-root of Equation (1.31). Assuming that both the cost function c and the latent divergence $\mathcal{D}$ are differentiable *w.r.t.* the model and variational parameters θ and ϕ, we can approximate $\widetilde{\mathcal{W}}_c^{\lambda}$ using Monte-Carlos sampling and minimize the resulting estimator with standard *s.g.d.* methods.

Latent divergence function Any divergence function on $\mathcal{Z} \times \mathcal{Z}$ could in theory be chosen in Equation (1.31). However, by construction, the integral over the data distribution in the aggregated posterior Q_Z is intractable. Nonetheless, we can easily obtain samples from the aggregated posterior using ancestral sampling: first, sample $x \sim p_x$, then sample $z \sim q(z|x)$. Thus, only divergence functions that can be approximated with sampling methods can be used. Secondly, the use of gradient-based methods requires the latent divergence to be tractable and differentiable *w.r.t.*

the variational parameters ϕ. The Maximum Mean Discrepancy (MMD) [Gretton et al., 2005] and the Jensen-Shanon divergence are potential candidates proposed in Tolstikhin et al. [2018], Bousquet et al. [2017].

1.3.2 Objective surgery

While VAEs and WAEs differ fundamentally in that they minimize different distances to approximate the data distribution, they share many similarities and parallelisms in their formulation given respectively in Equation (1.7) and (1.32). To make this analogy more explicit, let us reformulate the ELBO objective in Equation (1.7) as a minimization problem:

$$\begin{aligned}\mathcal{L}_{ELBO}\big(\theta,\phi\big) &\stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{P_X(X)}\big[-\mathcal{L}_{ELBO}\big(X,\theta,\phi\big)\big] \\ &= \mathop{\mathbb{E}}_{P_X(X)}\Big[\mathop{\mathbb{E}}_{Q_\phi(Z|X)} -\log p_\theta(X|Z) + \mathbf{KL}\big(Q_\phi(Z|X)\parallel P(Z)\big)\Big] \\ &= \mathop{\mathbb{E}}_{P_X(X)Q_\phi(Z|X)} -\log p_\theta(X|Z) + \mathop{\mathbb{E}}_{P_X(X)}\mathbf{KL}\big(Q_\phi(Z|X)\parallel P(Z)\big) \quad (1.33)\end{aligned}$$

Comparing Equation (1.33) to Equation (1.32), we can see striking resemblances in how the objectives can be decomposed into the sum of a reconstruction cost and a weighted latent regularizer.

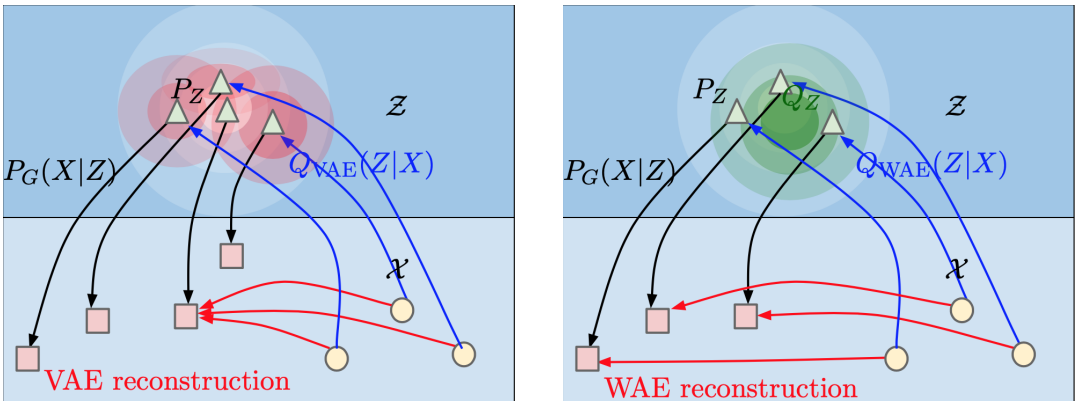
Reconstruction cost In both VAEs and WAEs, a reconstruction cost is used to measure the dissimilarity between the observations and the model reconstructions. This term penalizes models that fail at reconstructing accurately the observations. The main difference between VAEs and WAEs lies in the cost function of the reconstruction term. In VAEs, the cost function is the entropy of the generative model conditioned on the variational distribution: $R_{VAE}(x) = -\mathbb{E}_{q_\phi(z|x)}\log p_\theta(x|z)$. In WAEs, it is the ground cost c between the true data and the generated reconstructions averaged over the conditional reconstructions: $R_{WAE}(x) = \mathbb{E}_{q(z|x)}\mathbb{E}_{p_Y(y|z)}c(x,y)$.

By construction, the reconstruction term limits the choice of decoding distributions in VAEs. Only distributions with tractable (and differentiable) log density are possible due to the presence of the log function in Equation (1.33). This implies that only

decoders with implicit density p_θ such that $p_\theta(x|z) > 0$ *a.e.* can be used. As such, one can not use deterministic decoders to map the latent space to the observation space. This is a direct consequence of the strong topology induced by the KL divergence since using deterministic decoders to map low-dimensional priors to the observation space effectively results in low-dimensional latent manifold. Thus the intersection of the latent manifold with the data manifold will be contained in a set of measure null. A standard solution is to add a Gaussian noise to the output of the (deterministic) decoder, resulting in a Gaussian decoder as in Example 1.1.1. However, simply adding noise to the generative model might not be desirable, as it does not guarantee a consistent estimator [Zhang et al.] and can impact negatively the quality of the samples [Wu et al., 2016]. An alternative method [Zhang et al.] consists in adding the same noise to both the model and the data distributions. Under mild conditions, the authors show that the noisy versions of the distributions, the *spread* distributions, can be used to define a valid divergence, called *Spread Divergence*, between the two original distributions. In WAEs, the construction of the reconstruction term allows for any decoder distributions with a differentiable parametrization, and especially one can simply use deterministic decoders to map the latent variables to the observation space.

Latent regularizer In order to learn a smooth latent manifold from the observations, a latent regularizer term is present in both VAEs and WAEs. This regularizer shapes the way information is captured and encoded. Whilst the regularizers in VAEs and WAEs have the same goal of encouraging the approximate posterior to match the prior, they differ in the manner to do so. VAEs regularizes the approximate posterior on a point-wise level whereas in WAEs, only aggregated posterior is pushed toward the prior. Put differently, only the continuous mixture of approximate posteriors is penalized when drifting away from the prior. Indeed, in VAEs, for every single observation, the KL divergence penalizes encoders that differ from the prior as opposed to penalizing the aggregated posterior:

$$\mathop{\mathbb{E}}_{P_X(X)} \mathbf{KL}\big(Q_\phi(Z|X) \parallel P(Z)\big) \neq \mathbf{KL}\big(\mathop{\mathbb{E}}_{P_X(X)} Q_\phi(Z|X) \parallel P(Z)\big) \tag{1.34}$$



(a) VAE reconstruction. (b) WAE reconstruction

Figure 1.2: Schematic view of the reconstruction mechanism in VAE (a) and WAE (b). Each observation represented by a circle is mapped to a latent code in $\mathcal{Z}$ such that the variational distribution matches the prior on a point level for VAE or on average for WAE. These latent codes are then reconstructed with the generative model $P(X|Z)$.

This means that given an observation x, the conditional encoding distribution $Q(Z|x)$ will be encouraged to match the prior P_Z regardless of x. In other words, for different inputs $x \neq x'$, the resulting encoding distributions will share regions of high probability (see Figure 1.2a). Thus, it will be unlikely that latent codes sampled from the different encoding distributions $z \sim Q(Z|x)$ and $z' \sim Q(Z|x')$ stay far apart, even for highly different input observations x and x'. This, in turns, makes the task of reconstructing the observations harder for the decoder $p_Y(y|Z = z)$ where $z \sim q(z|x)$. The resulting overlap of regions with high probability under the different encoding distributions usually results in a non-smooth latent manifold. Indeed, to be able to distinguish between two similar latent codes $z \sim Q(Z|x)$ and $z' \sim Q(Z|x')$ whit $x \neq x'$, the decoder will have to be very sensitive to small variations in the latent variables. On the other hand, WAEs never encourage the encoder to match the prior for every single observation. Only the aggregated posterior averaged over the whole dataset, $Q_Z(Z)$, is pushed toward the prior. Thus, latent codes for different input observation will not be pushed toward one another space as long as the resulting mixture, when averaging over these points, is close enough (in a $\mathcal{D}$ sense) to the prior distribution as depicted in Figure 1.2b.

Another way to interpret the difference between the latent regularizers in WAEs

and VAEs is from a latent bottleneck point of view. Assuming that the latent divergence function in Equation (1.31) is chosen to be the KL divergence, then:

$$\begin{aligned}
\mathcal{D}\big(Q_Z P_Z\big) &\overset{\text{def}}{=} \mathbf{KL}\big(Q_Z \parallel P_Z\big) \\
&= -\int_{\mathcal{Z}}\int_{\mathcal{X}} p_X(x) q(z|x) \log \frac{p(z)}{\int_{\mathcal{X}} p_X(x) q(z|x)} \\
&= -\int_{\mathcal{Z}}\int_{\mathcal{X}} p_X(x) q(z|x) \log \frac{q(z|x)p(z)}{q(z|x)\int_{\mathcal{X}} p_X(x) q(z|x)} \\
&= \mathop{\mathbb{E}}_{P_X(X)} \mathbf{KL}\big(Q(Z|X) \parallel P(Z)\big) - \mathop{\mathbb{E}}_{P_X(X)} \mathbf{KL}\big(Q(Z|X) \parallel \mathop{\mathbb{E}}_{P_X(X)} Q(Z|X)\big) \\
&\leq \mathop{\mathbb{E}}_{P_X(X)} \mathbf{KL}\big(Q(Z|X) \parallel P(Z)\big) \qquad (1.35)
\end{aligned}$$

Thus, for a given latent information bottleneck capacity, for all observation x, the approximate posterior $Q(Z|X)$ will be more strongly penalized toward the prior $P(Z)$ in VAEs than in WAEs.

While offering more flexibility, the latent regularizer of WAEs comes with its own challenges. The main challenge lies in the choice of latent regularizer. Indeed, as mentioned earlier, the aggregated posterior Q_Z is intractable by assumption, with only a finite set of samples available. Thus, it will only be possible to compute samples-based estimates of the latent regularizer $\mathcal{D}\big(Q_{Z,\phi}, P_Z\big)$ in Equation (1.32). This limits the choice of divergence functions to the ones with estimators that show desirable properties. Specifically, the use of gradient descent methods to learn the variational parameters requires the latent estimator to be continuous and differentiable *w.r.t.* the variational parameters with preferably low-variance gradient. Moreover, the choice of latent divergence remains an open question as it will directly impact the performances of the model. Thus, not only it is necessary to find a divergence that fills the previous requirements, but it is also important to find one that shows good performances for the task at hand. Indeed, as we saw in Chapter 3 and Chapter 4, finding a latent divergence that suits the problem at hand impact dramatically the performances of the models. In VAEs on the other hand, the KL divergence is imposed to be the latent regularizer, which can be easily computed in many common cases (see Example 1.1.1). Additionally, the objective defined in Equation (1.32) introduces an extra hyper parameter in the form of the regularization

weight λ. While the theoretical guarantees of the Wasserstein distance described in Section 1.2.1 hold when $\lambda \to \infty$, the value of the penalization weight has to be chosen in practice. Even though hyper parameters tuning is not uncommon in machine learning, it raises the question of the role of λ in the convergence of the WAE and how close to the true Wasserstein distance the WAE is at convergence.

Chapter Summary In this chapter we lay out key background concepts needed to understand the remainder of the dissertation, especially, first focusing on generative modeling and the main methods for generative modelling such as Maximum Likelihood (ML) and Optimal Transport (OT).

First, we introduced generative models, which aim to replicate the process which generated the dataset at hand. Generative modeling involves approximating the true data distribution with a model distribution by minimizing a divergence or loss function. A popular framework to train generative models is the ML framework where the model distribution is trained to maximize the loglikelihood of the data under the model distribution. Amongst the likelihood-based methods, variational methods like Variational Autoencoders (VAEs) leverage the capacity of expressive generative models parametrized by neural networks by maximizing a tractable lower-bound of the intractable loglikelihood. We also provided a short overview of other models and methods in the field of generative modeling.

We then covered the OT framework, which provides tools to compare distributions geometrically. The Optimal Transport (OT) problem seeks the most efficient way to transport mass between distributions. A special case of OT is the Wasserstein distance, which lifts a ground metric on the observation space to a true distance between distributions. Unlike the Kullback Leibler divergence, the Wasserstein distance provides with a smoother training objective making easier to train generative models with standard gradient descent schemes.

Motivated by this, we then presented the Wasserstein Autoencoder (WAE). WAE minimizes an upper-bound on the Wasserstein distance between the data and model distributions. Like VAEs, WAEs use an encoder-decoder structure, first mapping the observation to a latent manifold before reconstructing them back to the observation space. However, WAEs differ in the reconstruction cost function and the latent regularizer. WAEs allow deterministic decoders, avoiding restrictive distributional assumptions on the decoder distribution. The latent regularizer penalizes deviations of the aggregated posterior from the prior as opposed to VAEs where the posterior is regularized on a data point level, enabling a more expressive latent representation.

Chapter 2

Improving Gaussian mixture latent variable model convergence by using Optimal Transport methods

The work presented in this chapter was published in [Gaujac et al., 2021a].

In the previous chapter, we introduced the foundations for latent generative models and Optimal Transport based methods for training such models. We will now present how they can be leveraged to improve the modelling performances of simple latent variable models compared to their likelihood counterparts.

Generative models with both discrete and continuous latent variables are highly motivated by the structure of many real-world datasets. They present, however, subtleties in training often manifesting in the discrete latent variables not being leveraged. In this chapter, we show why such models are hard to train using traditional loglikelihood maximization, and that they are amenable to training using the Optimal Transport (OT) with Wasserstein Autoencoders (WAEs). We find our discrete latent variables to be fully leveraged by the model when trained, without any modification to the objective function or significant fine tuning. Our model generates comparable samples to other approaches whilst using relatively simple

neural networks, since the discrete latent variable carries much of the descriptive burden. Furthermore, the discrete latent provides significant control over generation.

2.1 Introduction

Unsupervised learning using generative latent variable models provides a powerful approach to learning underlying low-dimensional structure from large, unlabeled datasets. Perhaps the two most common techniques for training such models are Variational Autoencoders (VAEs) [Kingma and Welling, 2014, Rezende et al., 2014] and Generative Adversarial Networks (GANs) [Goodfellow et al., 2014]. Both have advantages and disadvantages. VAEs provide a meaningful lower-bound of the loglikelihood that is stable under training and introduce an encoding distribution mapping from the data to the latent space. However, they generate blurry samples due to their objective being unable to handle deterministic decoders and the necessity of tractable models requiring simple priors [Hoffman and Johnson, 2016]. On the other hand, GANs naturally enable deterministic generative models with sharply defined samples, but their training procedure is less stable [Arjovsky and Bottou, 2017].

A relatively new approach to training generative models has emerged based on minimizing the OT distance [Villani, 2008] between the generative model distribution and that of the data. The OT approach provides a general framework for training generative models, which promises some of the best of both GANs and VAEs. Though interesting first results have been given in Tolstikhin et al. [2018], Bousquet et al. [2017], Arjovsky et al. [2017], Rubenstein et al. [2018b], the OT approach to generative modelling is still nascent.

Our contributions are twofold: we seek to improve generative modelling capabilities with discrete and continuous latent variables, but importantly, we seek also to establish that training generative models with OT can be significantly more effective than the traditional VAE approach.

Discrete latent variable models are critical to the endeavor of unsupervised learning because of the ubiquity of discreteness in the natural world, and hence in the datasets

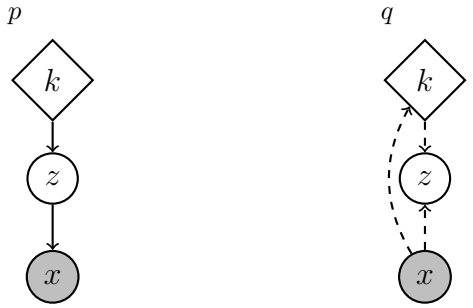
that describe it. However, they are harder to train than their continuous counterparts. This has been tackled in a number of ways (e.g., directly mitigating high-variance discrete samples [Eslami et al., 2016, Lawson et al., 2018], parametrizing discrete distributions using continuous ones [Jang et al., 2017, Maddison et al., 2017, Van den Oord et al., 2017], deliberate model design leveraging conjugacy [Johnson et al., 2016]).

However, even in the simple case where the number of mixtures is small enough that Monte Carlo sampling from the discrete latent is avoidable, training can still be problematic. For example, in Dilokthanakul et al. [2016] a Gaussian mixture latent variable model (GM-LVM) was studied, and the authors were unable to train their model on MNIST using variational inference without substantially modifying the VAE objective. What appears to happen is that the model quickly learns to "hack" the VAE objective function by collapsing the discrete variational distribution. This problem only occurs in the unsupervised setting, as Kingma et al. [2014] are able to learn the discrete latent in the semi-supervised version of the same problem once they have labeled samples for the discrete latent to latch onto. This is discussed in more detail in Section 2.2.1.

The OT approach to training generative models (in particular the Wasserstein distance, discussed in Section 2.2.2) induces a weaker topology on the space of distributions, enabling easier convergence of distributions than in the case of VAEs [Arjovsky et al., 2017]. Thus, one might conjecture that the OT approach would enable easier training of GM-LVMs than the VAE approach. We provide evidence that this is indeed the case, showing that GM-LVMs can be trained in the unsupervised setting on MNIST, and motivating further the value of the OT approach to generative modelling.

2.2 Gaussian mixture Wasserstein Autoencoders

We consider a hierarchical generative model p_G with two layers of latent variables, the highest one being discrete as shown in Figure 2.1a. Explicitly, if we denote the discrete latent k with density p_D, and the continuous latent z with density p_C, the



(a) Generative model (b) Inference distribution

Figure 2.1: GM-LVM graphical model. Round nodes represent continuous random variables, diamond-shaped nodes discrete variables and grey-shaded nodes the observations. (a): Generative model. (b): Inference distribution.

generative model is given by:

$$p_G(x) = \sum_{k=1}^{K} \int_{\mathcal{Z}} dz \; p_G(x|z) \, p_C(z|k) \, p_D(k) \tag{2.1}$$

In this work, we consider a GM-LVM with categorical distribution $p_D = \mathrm{Cat}(K)$ and continuous distribution $p_C(z|k) = \mathcal{N}\big(z; \mu_k^0, \Sigma_k^0\big)$. We refer to this GM-LVM as a GM-VAE when it is trained as a VAE [Kingma and Welling, 2014, Rezende et al., 2014] or GM-WAE when trained as a Wasserstein Autoencoder (WAE) [Tolstikhin et al., 2018, Bousquet et al., 2017] (discussed in Section 2.2.2).

2.2.1 The difficulty of training GM-VAEs

Training GM-LVMs in the traditional VAE framework (GM-VAEs) involves maximizing the Evidence Lower-Bound (ELBO) as defined in Equation (1.6) and recalled here:

$$\mathrm{ELBO}(x, P_G, Q) \stackrel{\text{def}}{=} \mathbb{E}_{q(z|x)}\big[\log p_G(x|z)\big] - \mathbf{KL}\big(q(z|x) \parallel p_G(z)\big) \tag{2.2}$$

where as before, $\mathbf{KL}\big(P_1 \parallel P_2\big)$ denotes the KL divergence between P_1 and P_2.

Such hierarchical models with discrete latent variables are empirically hard to train in the VAE framework [Dilokthanakul et al., 2016]. This is likely due to the

fact that the discrete variational distribution learns on a completely different scale from the generative distribution. Consequently, the discrete latent distribution tends to instantly learn some unbalanced structure where its classes are meaningless in order to accommodate the untrained generative distribution. The generative model then learns around that structure, galvanizing the meaningless discrete distribution early in training.

More explicitly, if we factorize the variational distribution as $q(z, k|x) = q_C(z|k, x)\, q_D(k|x)$ to mirror the prior in Equation (2.1) (see Figure 2.1b), the ELBO can be written as:

$$\begin{aligned} \text{ELBO} = \mathbb{E}_{q_D}\Big[\mathbb{E}_{q_C}\big[\log p_G(x|z)\big] - \mathbf{KL}\big(q_C(z|k,x) \parallel p_C(z|k)\big)\Big] \\ - \mathbf{KL}\big(q_D(k|x) \parallel p_D(k)\big) \end{aligned} \tag{2.3}$$

Both the first and the second term in Equation (2.3) depend on $q_D(k|x)$. However, the second term is much smaller than the first; it is bounded by $\log K$ for uniform p_D over K classes, whereas the first term is unbounded from above[1]. As a consequence, $q_D(k|x)$ will immediately shut off the k values (i.e., $q_D(k|x) = 0\ \forall x$) with large reconstruction losses, $\mathbb{E}_{q_C(z|k,x)} \log p_G(x|z)$. This is shown in the top row of Figure 2.2 where within the first 10 training steps the reconstruction loss has substantially decreased (Figure 2.2a) by simply shutting off 9 values of k in $q_D(k|x)$ (Figure 2.2b), resulting in a drastic increase of the discrete KL term (Figure 2.2a). However, this increase in the discrete KL term is negligible since the term is multiple orders of magnitude smaller than the reconstruction term in the ELBO. All of this takes place in the first few training iterations; well before the generative model has learned to use its continuous latent variables (see Figure 2.2c).

Subsequently, on a slower timescale, the generative model starts to learn to reconstruct from its continuous latent variables, causing $q_C(z|k, x)$ to shift away from its prior toward a more-useful distribution to the generative model. We see this in Figure 2.2d: the continuous KL curve grows concurrently with the downturn of the reconstruction loss term. Figure 2.2f shows that after this transition (taking a few thousands training steps), the reconstructions from the model start to look more like

[1]In the experiments, we initialized the modes of q_C to match those of the priors making the continuous KL term initially small as well.

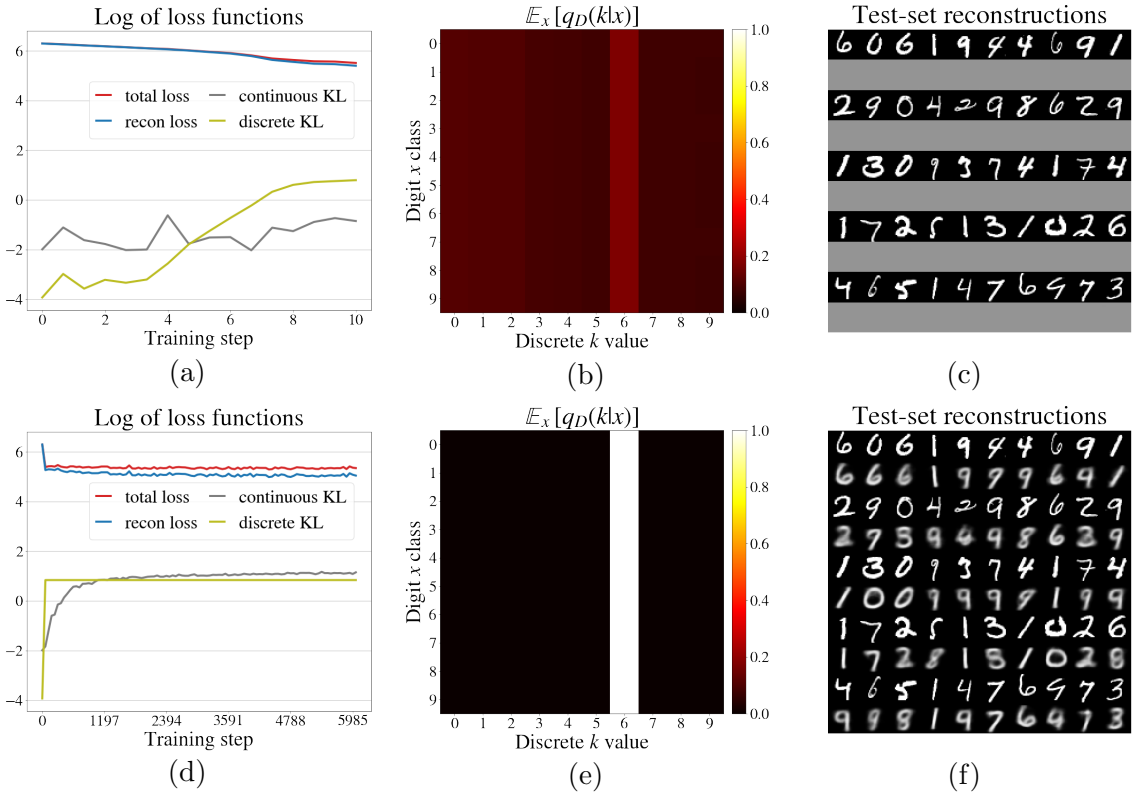


Figure 2.2: Top row shows a snapshot of the GM-VAE after 10 training steps. Loss curves are shown in (a), the discrete variational distribution in (b) with rows ℓ representing $\mathbb{E}_{\{x|\text{label}(x)=\ell\}}q_D(k|x)$, and reconstructions are shown in (c). Bottom row shows the same snapshot after 6000 training steps.

MNIST digits.

Whilst the generative model learns to use the continuous latents, the discrete distribution $q_D(k|x)$ never revives the k values that it shut off. This is because the generative model would not know how to use the latent codes $z \sim q_C(z|k,x)$ for those ks, implying a significant penalty in the reconstruction term of the ELBO. This is evidenced in Figure 2.2d by the discrete KL staying flat, and in Figure 2.2e where the columns corresponding to the shut off k values never repopulate.

We have discussed the difficulty of leveraging the structure of the latent variables in GM-VAEs using our specific implementation designed to mirror the GM-WAE of Section 2.2.2. Many other variants of this implementation performed similarly. Though the root cause of this difficulty has not been ascertained in generality, we expect it to be in part due to the per-data-point nature of the ELBO objective, in particular, the impact of the KL divergence term on learning the variational

distribution. This point will be elaborated upon with more empirical justification in Section 2.3.

2.2.2 Optimal Transport facilitates training of GM-LVMs

The difficulty associated with training GM-VAEs may be interpreted as a problem of restricted convergence of a sequence of distributions, where the sequence is indexed by the training steps. If that were so, an objective function that induces a weaker topology might help GM-LVMs to converge to a distribution that non-trivially uses its discrete latent variable. Hence, we are motivated to approach the training of such models with the OT framework, and in particular the Wasserstein distance as our objective, as it induces a weaker topology than that of maximum likelihood (Theorem 2 of Villani [2008]).

Following Tolstikhin et al. [2018], Bousquet et al. [2017], we would like to minimize the 2-Wasserstein distance between the underlying data distribution (from which we have samples) and our GM-LVM:

$$W_2^\dagger\big(P_{\text{data}}, P_G\big)^2 = \inf_{Q \in \mathcal{P}_{\mathcal{Z}\times\mathcal{K}}} \mathbb{E}_{p_{\text{data}}(x)} \mathbb{E}_{q(z,k|x)} \mathbb{E}_{p_G(y|z)} \, \|x - y\|_2^2 \tag{2.4}$$

where $\mathcal{P}_{\mathcal{Z}\times\mathcal{K}}$ is the set of all joint distributions over z and k such that:

$$\mathbb{E}_{p_{\text{data}}(x)} q(z, k|x) = p_C(z|k) p_D(k) \tag{2.5}$$

We further constraint the variational joint distribution $q(z, k|x)$ in Equation (2.5) to factorize as $q(z, k|x) = q_C(z|k, x)\, q_D(k|x)$ making the resulting infimum an upper-bound of $W_2^\dagger$. As for the VAE case, our parametrization of the variational distribution $q(z, k|x)$ is deliberate and aims to mirror the structure of the prior (see Figure 2.1b). It differs from other methods, for example Makhzani et al. [2016] who assume conditional independence between $z|x$ and $k|x$.

Since the constrained infimum is intractable, a relaxed version of $W_2^\dagger$ is introduced

as follows:

$$\widetilde{W}_2^{\dagger}\big(P_{\text{data}}, P_G\big)^2 = \inf_{Q \in \mathcal{P}_{\mathcal{Z}\times\mathcal{K}}} \mathbb{E}_{p_{\text{data}}(x)}\, \mathbb{E}_{q(z,k|x)}\, \mathbb{E}_{p_G(y|z)}\ \|x-y\|_2^2 + \lambda\, \mathcal{D}\big(\mathbb{E}_{p_{\text{data}}(x)} q(z,k|x), p_C(z|k) p_D(k)\big) \tag{2.6}$$

which is equivalent to the original distance when $\lambda \to \infty$.

As in Tolstikhin et al. [2018], Bousquet et al. [2017], we use the Maximum Mean Discrepancy (MMD) with a mixture of Inverse Multiquadric (IMQ) kernels with various bandwidth C^i. The MMD is a distance on the space of distributions and has an unbiased U-estimator [Gretton et al., 2012a]. Explicitly, if k is a reproducing positive-definite kernel and is characteristic, then the MMD associated to k is given by:

$$\text{MMD}\Big(Q \parallel P\Big) = \mathbb{E}_{q(z_1)q(z_2)}[k(z_1,z_2)] + \mathbb{E}_{p(z_1)p(z_2)}[k(z_1,z_2)] - 2\mathbb{E}_{q(z_1),p(z_2)}[k(z_1,z_2)] \tag{2.7}$$

IMQ kernels have fatter tails than the classic radial basis function kernels, proving more useful early in training when the encoder has not yet learned to match the aggregated posterior with the prior. We take a mixture of kernels k_i with bandwidths $C^i \in \{10^j, 2\times 10^j, 5\times 10^j \mid j \in \{-2,\dots,2\}\}$ to reduce the sensitivity on any one choice of kernel bandwidth (Dziugaite et al. [2015], Gretton et al. [2012b], Li et al. [2015]). The Sinkhorn distance (Cuturi [2013a], Genevay et al. [2018]) is another candidate and has been used in [Patrini et al., 2018]. However, it did not improve significantly our results but came with a high cost in computational time. Thus, we choose to use the MMD as Tolstikhin et al. [2018].

Given the discrete latent in our model, we cannot directly use Equation (2.6) with the MMD. Instead we integrate out the discrete latent variable in our approximate posterior $q(z,k|x)$: $q(z|x) = \sum_k q(z,k|x)$. The reconstruction term in Equation (2.4) is unchanged as we have for all $f(.,x)$ integrable for all x:

$$\mathbb{E}_{q(z,k|x)}[f(z,x)] = \mathbb{E}_{\sum_k q(z,k|x)}[f(z,x)] \tag{2.8}$$

Now, the constrain in Equation (2.4) becomes:

$$\mathbb{E}_{p_{\text{data}}(x)}[q(z|x)] = p(z) = \sum_k p_C(z|k)p_D(k) \tag{2.9}$$

Adding everything together, we obtain our GM-WAE objective function:

$$\begin{aligned}\widetilde{W}_2^{\dagger}\big(P_{data}, P_G\big)^2 \;=\; & \inf_{q(z|x)\in\mathcal{P}_{\mathcal{Z}}} \mathbb{E}_{p_{\text{data}}(x)}\, \mathbb{E}_{q(z|x)}\, \mathbb{E}_{p_G(y|z)}\, \|x-y\|_2^2 \\ & + \lambda\, \text{MMD}\Big(\mathbb{E}_{p_{\text{data}}(x)} q(z|x) \parallel \sum_k p_C(z|k)\, p_D(k)\Big)\end{aligned} \tag{2.10}$$

This allows us to compute the MMD between two continuous distributions, where it is defined.

As mentioned in Section 1.3.2, VAEs have the disadvantage that deterministic generative models cannot be used; this is not the case for WAEs. Thus we parametrize the generative density $p_G(x|z)$ as a deterministic distribution $x|z = g_\theta(z)$ where g_θ is a mapping from the latent to the data space specified by a deep neural network with parameters θ.

We also parametrize $q(z, k|x) = q_C(z|k, x)\, q_D(k|x)$ with neural networks. We take $q_C(z|k, x)$ for each k to be a Gaussian distribution with diagonal covariance given by the outputs of the neural network and use the reparametrization trick [Kingma and Welling, 2014, Rezende et al., 2014] to compute gradients with regard to the networks parameters. In order to avoid back propagating through discrete variables, the expectation over the distribution $q_D(k|x)$ is computed exactly as opposed to sampling first $k \sim \text{Cat}(K)$ and then obtaining the continuous latent variable $z \sim \mathcal{N}\big(z; \mu_k, \Sigma_k\big)$.

Since we know before hand that the MNIST dataset has only 10 discrete classes, it is both easily and computational feasible to compute the discrete expectation explicitly by summing over all of the 10 discrete classes. However, for more complex dataset (such as CelebA [Liu et al., 2015]) where the number of classes is significantly bigger making the explicit calculation of the discrete expectation intractable, one can compute the expectation by sampling using standard techniques that relaxed the discrete distribution [Brooks et al., 2011, Jang et al., 2017, Maddison et al., 2017].

As previously mentioned, the weakness of the topology induced by the Wasserstein distance on the space of distributions may enable the GM-WAE to overcome the training issues of VAEs discussed in Section 2.2.1. With the objective in hand, a more precise argument can be made to support this claim.

Recall from Section 2.2.1 that the problem with the GM-VAE was that the objective function demands the various distributions to be optimized at the individual data-point level. For example, the $\mathbf{KL}\big(q_D(k|x) \parallel p_D(k)\big)$ term in Equation (2.3) breaks off completely and becomes irrelevant due to its size. This causes the $q_D(k|x)$ distribution to shut off k values early, which becomes galvanized as the generative model learns.

However, in posing the problem in terms of the most efficient way to move one distribution p_G onto another p_{data}, via the latent distribution $q(z, k|x)$, the Wasserstein distance never demands the similarity of two distributions conditioned per data point. Indeed, the expectation over the data distribution in Equation (2.10) is inside both the infimum and the divergence $\mathcal{D}$. We expect that "aggregating" the posterior as such will give $q(z, k|x)$ (in particular, $q_D(k|x)$) the flexibility to learn data-point specific information whilst still matching the prior on aggregate. Indeed, it is also found in Makhzani et al. [2016] that using an adversarial game to minimize the distance between an aggregated posterior and the prior is successful at unsupervised training on MNIST with a discrete-continuous latent variable model.

2.3 Results

In this work we primarily seek to show the potential for OT techniques to enable the training of GM-LVMs. Thus, we use relatively simple neural network architectures and train on the MNIST toy dataset. Given the nature of the dataset, we use a mixture of 10 Gaussians, one for each of the 10 digits and a non-informative uniform prior over these mixtures. Namely, for each $k \in \{0, \dots, 9\}$:

$$p_D(k) = \frac{1}{10} \quad \text{and} \quad p_C(z|k) = \mathcal{N}(z; \mu_k^0, \sigma_k^0) \tag{2.11}$$

where the μ_k^0 and σ_k^0 represent the mean and covariance of each mixture and are fixed before training.

For the variational distribution, we take $q(z, k|x) = q_C(z|k, x)\, q_D(k|x)$ with

$$q_D(k|x) = \pi_k(x) \quad \text{and} \quad q_C(z|k, x) = \mathcal{N}\big(z; \mu_k(x), \mathrm{diag}(\sigma_k(x))\big) \tag{2.12}$$

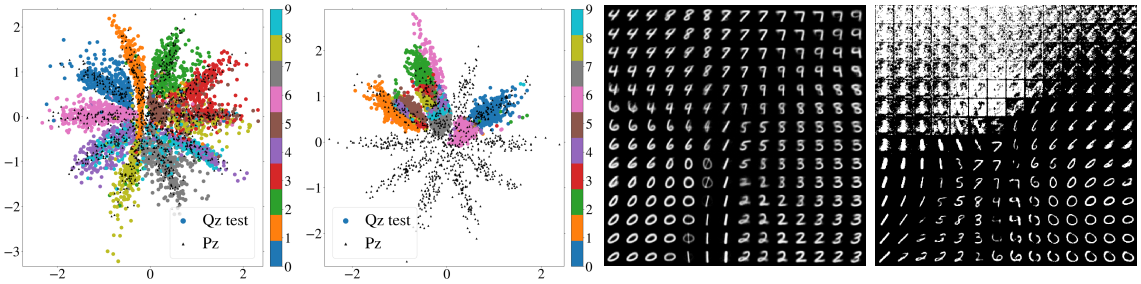
where each component is parametrized by a neural network. For $\pi_k(x)$ a 3-layer deep convolutional generative adversarial network (DCGAN) [Radford et al., 2015] is used with the largest convolution layer composed of 64 filters. The Gaussian networks $\mu_k(x)$ and $\sigma_k(x)$ are taken to be fully connected networks with two hidden dense layers, each with 32-units. Finally, for the generative model, we take $p_G(x|z)$ to be deterministic with $x|z = g_\theta(z)$, where g_θ is a 3-layers DCGAN-style network with parameters θ. All the convolutional filters have size 5×5 except for the last layer which has size 1×1. The smallest transpose convolution of g_θ has 128 filters. We use batch normalisation [Ioffe and Szegedy, 2015] and ReLU [Glorot et al., 2011] after each hidden layer and train the networks with Adam optimizer [Kingma and Ba, 2015] with a learning rate of 0.0005. We find that $\lambda = 500$ works well, although the value of λ does not impact the performances appreciably as long as it is larger than a few hundred. The mean parameters μ_k and σ_k are pretrained to match the prior moments, which accelerate the training and improves the stability (this was also done for GM-VAE in Section 2.2.1).

2.3.1 Ablation study of the learned latent manifold

In this section, we study and compare the structure of the latent manifold learned by WAE and VAE. We chose $\dim(z) = 2$ with the prior being a mixture of 10 2-D Gaussians allowing for an easy visualisation and interpretation of the latent manifold. The prior is taken such that each mixture component $p_C(z|k)$, $k \in \{1, \cdots, 9\}$, is obtained by rotating the 2-D Gaussian distribution $p_C(z|0)$ by ω_k:

$$\begin{aligned} p_C(z|0) &= \mathcal{N}(z; \mu_0^0, \sigma_0^0) \\ p_C(z|k) &= \mathcal{N}(z;\ \mathcal{R}(\omega_k) \cdot \mu_0^0,\ \mathcal{R}(\omega_k) \cdot \sigma_0^0 \cdot \mathcal{R}(\omega_k)^\top) \end{aligned} \tag{2.13}$$

where $\mu_0^0 = [\mu_x^0, \mu_y^0]$, $\sigma_0^0 = \text{diag}(\sigma_x^0, \sigma_y^0)$, $\mathcal{R}(\omega)$ designs the rotation matrix in $\mathbb{R}^2$ by ω and $\cdot$ the matrix multiplication in $\mathbb{R}^{2\times 2}$. We take $\mu_x^0 = 1$, $\mu_y^0 = 0$, $\sigma_x^0 = 0.1$, $\sigma_y^0 = 0.5$ and $\omega_k = \frac{2\pi k}{10}$ for each $k \in \{1, \dots, 9\}$. To visualize the structure captured in the latent space, we encode 5000 observation points from the testing set and show the latent codes z sampled from $q_z(z|x) = \sum_k q_C(z|k,x)q_D(k|x)$ in Figures 2.3a and 2.3b when using WAE and VAE respectively. We color each latent point according to the corresponding digit label and add samples from the prior $p_z(z) = \sum_k p_C(z|k)p_D(k)$ for better visualization. Similarly, the inverse mapping from the latent space to the observations is given Figures 2.3c and 2.3d, corresponding to the latent space interpolations for WAE and VAE respectively. Each reconstruction corresponds to a latent code sampled uniformly on the 2-d latent grid.



(a) (b) (c) (d)

Figure 2.3: Latent manifold analysis. (a): latent manifold visualization for WAE. (b): latent manifold visualization for VAE. (c): latent space interpolation for WAE. (d): latent space interpolation for VAE. 5000 test observation points and 1000 samples from the prior are used. Encoded points are colored by their digit label.

WAE manages to closely match the aggregated posterior with the prior, leveraging the discrete structure of the prior. Indeed, the latent manifold aligns with the different components of the prior (Figures 2.3a), almost succeeding in assigning each component to a unique digit class. The smooth structure of the latent interpolation plot (Figures 2.3c) shows how WAE manages to capture all the structure of the data into the latent space. In contrast, the latent representation learned by VAE fails to match the prior, and especially, only uses a few components of the latent mixture (Figures 2.3b). The poor latent interpolation (Figures 2.3d) indicates that the data manifold does not align with the chosen latent manifold defined by the prior. This illustrates one more time the difficulty to train GM-LVMs with VAE where several

prior modes collapsed, as discussed in Section 2.2.1.

2.3.2 Generative performances

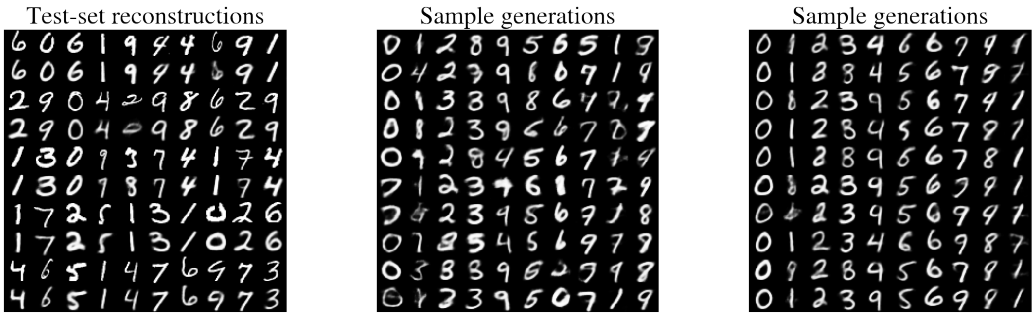
In this section, we chose μ_k^0 and σ_k^0 (Equation (2.11)) such that the μ_k^0 are mutually equidistant and $\sigma_k^0 = \sigma^0\mathbf{Id}$ with σ^0 such that $\approx 5\%$ overlap between the 10 different modes of the prior (i.e., the distance between any pair of means $\mu_{k_1}^0$ and $\mu_{k_2}^0$ is $4\sigma^0$). We fix $\dim(z) = 10$ and use the same architectures than above.

2.3.2.1 Reconstruction, samples and latent interpolation

Our implementation of GM-WAE is able to reconstruct MNIST digits from its latent variables well. In Figure 2.4a example data points from the held-out test set are shown on the odd rows, with their reconstructions on the respective rows below. The encoding of the input points is a two step process, first determining in which mode to encode the input via the discrete latent, and then drawing the continuous encoding from the corresponding mode.

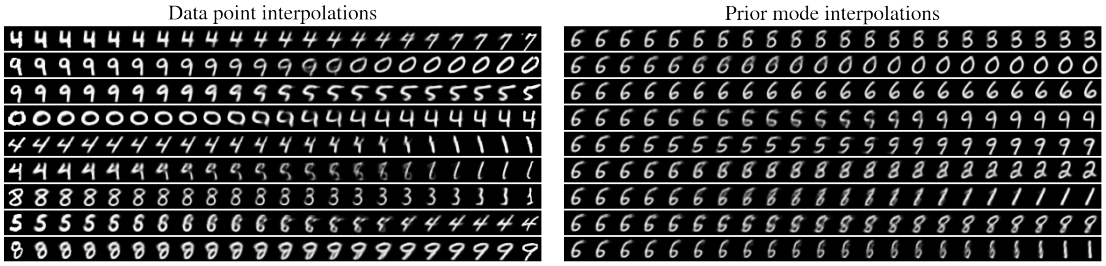
Samples from the GM-WAE are shown in Figures 2.4b and 2.4c. Since the discrete prior $p_D(k)$ is uniform, we can sample evenly across the ks in order from 0 through 9, whilst still displaying representative samples from $p(z, k) = p_C(z|k)p_D(k)$. Again, this shows how the GM-WAE learns to leverage the structure of the prior, whereas the GM-VAE results in the collapse of the several modes of the prior.

GM-WAE has a smooth manifold structure in its latent variables as shown in Figure 2.5a which plots the reconstructions of linear interpolations between pairs of data points is shown. The interpolation is done in the latent space, with uniform step size between the encodings of the two observations. This compares similarly to other WAE and VAE approaches to MNIST. In Figure 2.5b a linear interpolation is performed between the prior mode μ_6^0, and the other nine prior modes $\mu_{k\neq 6}^0$. This not only shows the smoothness of the learned latent manifold in all directions around a single mode of the prior, but also that the variatonal distribution has learned to match the modes of the prior. As excpected given the suitability of a 10-mode GM-LVM to MNIST, almost every mode of the prior now represents a different digit. This level of control built into the prior requires not only a multi-modal prior, but



(a) (b) (c)

Figure 2.4: Shown in (a) are reconstructions of held-out data from the inferred latent variables. The first, third, etc, rows are the raw data, and the rows below show the corresponding reconstructions. Digit samples $x \sim p_G(x|z)\, p_C(z|k)$ for each discrete latent variable k are shown in (b) as well as those samples closer to each mode of the prior in (c). The samples in (c) come from z values sampled from Gaussians identical to $p_C(z|k)$, except with standard deviation scaled down by 3/5.

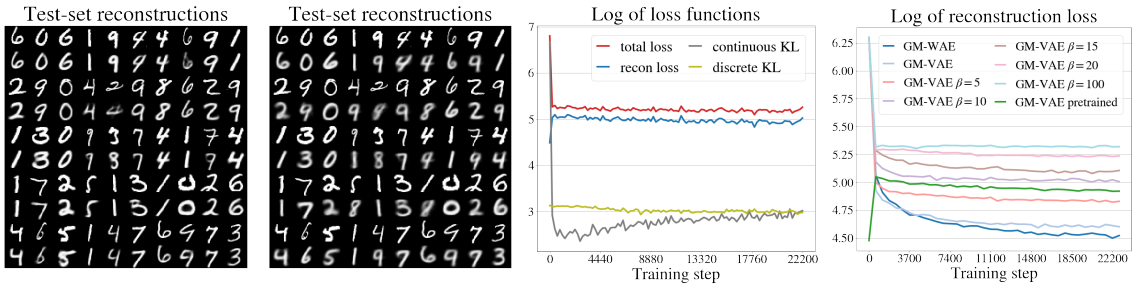


(a) (b)

Figure 2.5: Reconstructions from linear interpolations in the continuous latent space between two data points (a), and between the prior mode μ_6^0, and the other nine prior modes $\mu_{k\neq 6}^0$ (b). In (a), the true data points are shown in the first and last column next to their direct reconstructions.

also a training procedure that actually leverages the structure in both the prior and variational distribution, which seems to not be the case for VAEs (see Section 2.2.1).

The quality of samples from our GM-WAE is related to the ability of the encoder network to match the prior distribution. Figures 2.4c and 2.5b demonstrate that the latent manifold learned is similar to the prior. Near the modes of the prior the samples are credible handwritten digits, with the encoder network able to capture the structure within each mode of the data manifold (variation within each column) and clearly separate the different modes (variation between rows).



(a) (b) (c) (d)

Figure 2.6: (a) Reconstructions for an untrained VAE initialized with same parameters as our trained WAE. (b) Those same reconstructions after a few dozen thousands training steps according to the VAE objective. (c) Learning curves from an untrained VAE initialized with same parameters as our trained WAE. (d) Reconstruction loss for different VAE variations.

2.3.2.2 Comparison with VAE

We have argued that the VAE objective itself was responsible for the collapse of certain k values in the discrete variational distribution, and that the per-data-point nature of the KL played a significant role. To test this hypothesis, and to compare directly our trained WAE with the equivalent VAE discussed in Section 2.2.1, we initialize the VAE with the final parameters of the trained WAE, and train it according to the VAE objective. At initialization, the VAE with trained WAE parameters produces high quality samples and reconstructions (Figure 2.6a). However, after a few hundred iterations, the reconstructions deteriorate significantly (Figure 2.6b), and are not improved with further training.

The learning curves over the period of training between Figure 2.6a and 2.6b are shown in Figure 2.6c, where the cause of the performance deterioration is clear: the continuous KL term in the VAE objective is multiple orders of magnitude smaller than the reconstruction term, causing optimization to sacrifice reconstruction in order to reduce this KL term. Of course, the approximate posterior aggregated over the data will not be far from the prior as that distance is minimized in the WAE objective. However, this is not enough to ensure that the continuous KL term is small for every data point individually. It is thus the per-data-point nature of the KL in the VAE objective that destroys the reconstructions. Indeed, in order to minimize the per-data-point KL term in the GM-VAE objective, $q_C(z|k,x)$ is forced toward

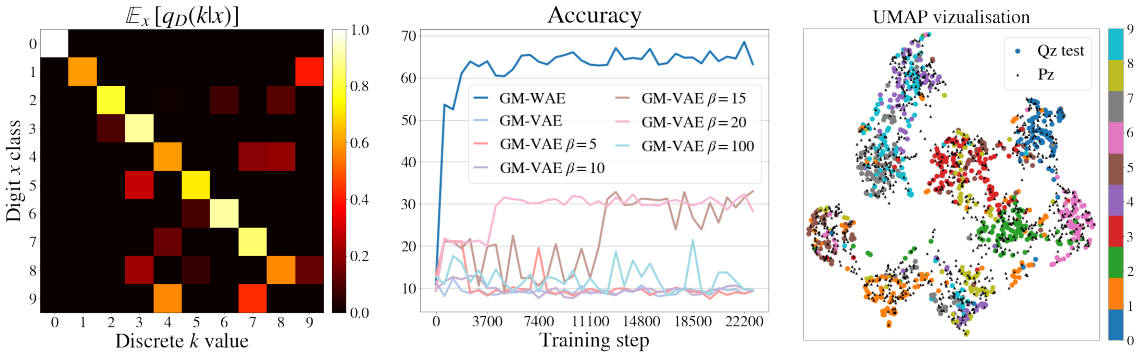
the mean μ_k^0 for every x, causing it to lose much of its x dependence. This can be seen in Figure 2.6b where the reconstructions are less customized and blurrier.

To compare the performance of GM-WAE against GM-VAE more quantitatively, we directly compare the reconstruction loss from the VAE objective (the first term on the right hand side of Equation (2.3)). Strictly speaking, this quantity is ill-defined for the GM-WAE, as the generative model is chosen to be deterministic. Instead we simply use the values returned by the GM-WAE generative model as if they were the Bernoulli mean parameters of the GM-VAE [Kingma et al., 2014]. These reconstruction loss curves are shown Figure 2.6d. Also shown are the reconstruction losses for the GM-VAE with various rescaling factors β in front of the KL terms of Equation (2.3). This rescaled KL term is inspired by both Higgins et al. [2016], which studies the impact of weighting the KL term in VAEs, and by the WAE objective itself where λ plays the role of a regularization coefficient. Whilst, the GM-WAE is not trained to minimized this reconstruction loss[2], it actually achieves the smallest reconstruction loss. This shows that GM-WAE performs better at reconstructing MNIST digits than its VAE counterpart, as measured by the VAEs own reconstruction objective.

We also show in Figure 2.6d the reconstruction curve of a GM-VAE initialized with trained GM-WAE parameters. This echoes the previous discussion concerning the deterioration of the reconstructions in GM-VAEs due to the per-data-point KL term. In Figures 2.6c and 2.6d, the GM-VAE initialized with trained GM-WAE parameters uses a rescaling factor $\beta = 10$ for visualization purposes. The same phenomenological behavior is observed with no scaling factor, just less visually pronounced.

Overall, our results for GM-WAE are qualitatively competitive with other approaches [Tolstikhin et al., 2018, Bousquet et al., 2017], despite a relatively low-complexity implementation. Furthermore, GM-WAE offers more control over generation and inference due to its latent variable structure, which cannot be achieved with the GM-VAE objective.

[2]Recall that we used the squared Euclidean distance as our ground cost function in Equation(2.10).



(a) (b) (c)

Figure 2.7: Visualization of the variational distributions. (a) shows $\mathbb{E}_{\{x|\text{label}(x)=\ell\}} q_D(k|x)$ in row ℓ. (b) shows the accuracy as a function of the training steps for our method and the same VAE variations than Figure 2.6d. (c) shows $z|x \sim \sum_k q_C(z|k,x)q_D(k|x)$ using the UMAP visualization method [McInnes and Healy, 2018]. 1000 encoded test-set digits and 1000 samples from the prior are used. Encoded points are colored by their digit label.

2.3.2.3 Latent variable fidelity

We have shown that the GM-WAE is able to both reconstruct data and generate new samples meaningfully from the prior distribution. We now turn to studying the variational distributions directly, including with how much fidelity a given class of digits is paired with a given discrete latent.

Consider first the discrete distribution $q_D(k|x)$ shown in Figure 2.7a. Here, for each digit label, $\mathbb{E}_{\{x|\text{label}(x)=\ell\}} q_D(k|x)$ is shown in row ℓ. From the staircase structure, it is clear that this distribution learns to approximately assign each discrete latent value k to a different class of digit. However, it does not do so perfectly. This is expected as the GM-WAE seeks only to reconstruct the data from its encoding, not to encode it in any particular way. This does not mean GM-WAE is failing to use its discrete latent effectively. Indeed, when comparing Figure 2.4c and Figure 2.7a, a meaningful source of overlap between different values of k and a single digit class can be seen. For example, in Figure 2.7a the digit 5 is assigned partially to $k = 3$ and $k = 5$. In Figure 2.4c, 5s drawn with a big-round lower loop are similar to digit 3 and 5s with a small loop and long upper bar are assigned to another cluster corresponding to digit 5. A similar discussion holds for 8s and 9s.

To assess the digit-class fidelity of the discrete encoder more quantitatively, we

calculate the accuracy of the digit-class assignment according to $q_D(k|x)$. To assign a digit-class label to each k value, we follow a similar protocol to that of Makhzani et al. [2016]: we assign the digit-class label to the k value that maximizes the average discrete latent distribution for that class $\mathbb{E}_{x_m, y_m=i} q(k|x_m)$ where $i = 0, \cdots, 9$ is the digit-class label, and for all m, y_m is the label of the corresponding data point x_m. In order to avoid multiple class assignments to the same cluster, we assign the label corresponding to the highest maximum. Figure 2.7b shows the resulting accuracy throughout training. Our GM-WAE achieves an accuracy on the held-out test set just shy of 70%. The accuracy corresponding to the different GM-VAE variations of Figure 2.6 is also shown in Figure 2.7b. The best performing GM-VAE with a scaling factor of $\beta = 20$ achieves approximately 30%. This shows again the difficulty of the GM-VAE to capture meaningful structure in the data. For reference, basic K-means clustering [MacQueen, 1967] achieves 50-60% accuracy, and Makhzani et al. [2016] achieve 90% (using 16 discrete classes, and substantially different model and training procedure).

Another way to study the latent variable structure of GM-WAE is to consider a low-dimensional visualization of the continuous latent z. In Figure 2.7c such a visualization is shown using UMAP [McInnes and Healy, 2018]. Distinct clusters can indeed be seen in the prior and in the samples from $q_C(z|k, x)$. Though the clusters of $z \sim q_C(z|k, x)$ do not fully align with those from the prior $z \sim p_D(z|k)$, they maintain significant overlap. Samples from $q_C(z|k, x)$ in Figure 2.7c are colored according to the true digit labels and show how GM-WAE learns to assign digits to the different clusters. In particular, the 7 / 9 cluster is clearly overlapping, as seen in Figures 2.7a, 2.4b and 2.4c.

We have see that the GM-WAE model is highly suited to the problem under study. It reconstructs data and provides meaningful samples, it effectively uses both the discrete and continuous variational distributions, all whilst maintaining close proximity between the variational distribution and the prior.

2.4 Conclusions

We studied an unsupervised generative model with a Gaussian mixture latent variable structure, well suited to data containing discrete classes of objects with continuous variation within each class. We showed that such a simple and critical class of models fails to train using the VAE framework, in the sense that it immediately learns to discard its discrete-latent structure. We further exposed the root cause of this phenomenon with empirical results. We then put to the test the abstract mathematical claim that the Wasserstein distance induces a weaker topology on the space of distributions by attempting to train the same Gaussian mixture model in the WAE framework. We found that the Wasserstein objective is successful at training this model by fully leveraging its discrete-continuous latent structure. We provided promising results on MNIST and demonstrated the additional control available with highly structured model with both discrete and continuous latent variables.

Chapter 3

Learning deep latent Hierarchies by Stacking Wasserstein Autoencoders

The work presented in this chapter is under review and can be found here [Gaujac et al., 2022].

We will now show how OT-based methods, and especially WAEs, can be used to train deep latent hierarchical model in a simple principle way. Probabilistic models with hierarchical latent variable structures provide state-of-the-art results amongst non-autoregressive, unsupervised density-based models. However, the most common approaches based on VAEs often fail to leverage deep latent hierarchies; successful approaches require complex inference and optimisation schemes. In this chapter we propose a novel approach to training models with fully Markovian deep latent hierarchies based on OT, without the need for highly bespoke inference networks and training schemes. We show that our method enables generative models to fully leverage their deep latent hierarchy, avoiding the *collapse* of the latent variables as encountered with VAEs. This provides qualitatively more interpretable latent representations whilst keeping low computational cost and memory requirements for applications where only inference is of interest.

3.1 Introduction

Probabilistic latent variable models are widely used models for extracting structure from large, unlabelled datasets. VAEs [Kingma and Welling, 2014, Rezende et al., 2014] have proven to be effective for training generative models parametrized by powerful neural networks, mapping the data into a low-dimensional latent manifold. Whilst allowing the training of expressive models, VAEs often fail when deeper hierarchies with several latent layers are used.

In fact, many of the most successful generative models use only a single latent layer. Auto-regressive latent models [Gulrajani et al., 2016, Chen et al., 2017, Van den Oord et al., 2016c], for example, achieve near state-of-the-art negative likelihood scores and show impressive samples quality. However, auto-regressive VAEs suffer from poor scalability to high-dimensional data. Flow-based models [Rezende and Mohamed, 2015, Kingma et al., 2016, Dinh et al., 2016a, Kingma and Dhariwal, 2018] are another class of generative models with one latent layer. They provide competitive sample quality and are able to scale to higher-dimensional data but achieve lower likelihood score.

Deep latent variable models are highly expressive models that aim to capture the structure of data in a hierarchical manner. Their deep latent structures offer more explainable latent representations whilst improving the generative performances, achieving state-of-the-art likelihood scores [Maaløe et al., 2019, Vahdat and Kautz, 2020]. However, they remain hard to train. Many explanations have been proposed, from the use of dissimilarity measures directly in the pixel space [Larsen et al., 2016] resulting in poor sample quality and the lack of efficient representation in the latent space [Zhao et al., 2017], to simply the poor expressiveness of the models used [Zha et al., 2017, Maaløe et al., 2019].

Solutions for training deep latent variable models range from introducing auxiliary variables to increase the expressiveness of the posterior distribution [Maaløe et al., 2016] to replacing the simple spherical Gaussian prior with two layers mixture distributions [Tomczak and Welling, 2018, Klushyn et al., 2019]. State-of-the-art methods design complex generative models and inference networks introducing skip

connections in the generative model and inference networks [Bachman, 2016, Kingma et al., 2016], bottom-down inference networks sharing parameters with the generative model [Sønderby et al., 2016], bidirectional inference networks [Maaløe et al., 2019] as well as special architectures for the building blocks of the deep networks [Vahdat and Kautz, 2020]. Additionally, these approaches often rely on complex training schemes such as KL *warm-up* [Sønderby et al., 2016, Vahdat and Kautz, 2020], *free bits* strategies [Kingma et al., 2016, Maaløe et al., 2019] and spectral regularizations of the networks weights [Yoshida and Miyato, 2017, Vahdat and Kautz, 2020] as well as specific parametrizations of the inference distributions [Vahdat and Kautz, 2020].

Optimal Transport [Villani, 2008] is a mathematical framework to compute distances between distributions. Recently, it has been successfully used to train generative models [Arjovsky et al., 2017, Tolstikhin et al., 2018, Bousquet et al., 2017, Genevay et al., 2018]. Tolstikhin et al. [2018], Bousquet et al. [2017] introduced WAEs, where an encoder maps the data into a latent space, learning a low-dimensional representation of the data. WAEs provide a non-likelihood based framework with appealing topological properties [Arjovsky et al., 2017, Bousquet et al., 2017], allowing in theory for easier training convergence between distributions. Gaujac et al. [2021a] trained a two-latent-layer generative model using WAE, showing promising results in the capacity of WAE to leverage a latent hierarchy relative to the equivalent VAE.

Following these early successes, we propose to train deep hierarchical latent models using the WAE framework, without the need for complex dependency paths in both the generative model and inference network. Similarly to earlier works [Sønderby et al., 2016, Maaløe et al., 2019, Vahdat and Kautz, 2020], we believe that a deep latent hierarchy, when trained properly, increases the expressiveness of the generative models. In order to leverage the deep latent hierarchy, we derive a novel objective function by stacking WAEs, introducing an inference network at each level and encoding the information up to the deepest layer in a simple bottom-up way. For convenience, we refer to our method as StWAE.

Our contributions are fourthfold:

- We introduce StWAE, a novel objective function based on OT, designed specifically for generative models with deep latent hierarchies.

- We show that StWAE performs significantly better when training hierarchical latent variable models than the original WAE which fails at propagating the information to the deeper latent layers.

- We compare our method with VAEs, quantitatively showing that StWAE achieves better generative performances whilst learning better latent representations.

- We qualitatively show that our method can be scaled to deeper latent hierarchies when working with more realistic datasets.

3.2 Stacked WAE

StWAEs are probabilistic latent variable models with a deep hierarchy of latents. They can be minimalistically simple in their inference and generative models, but are trained using OT in a novel way. We start by defining the probabilistic models that are of interest in this work, then introduce OT, and finally, we discuss how to train probabilistic models with deep latent hierarchies using OT, the method that we refer to as StWAE.

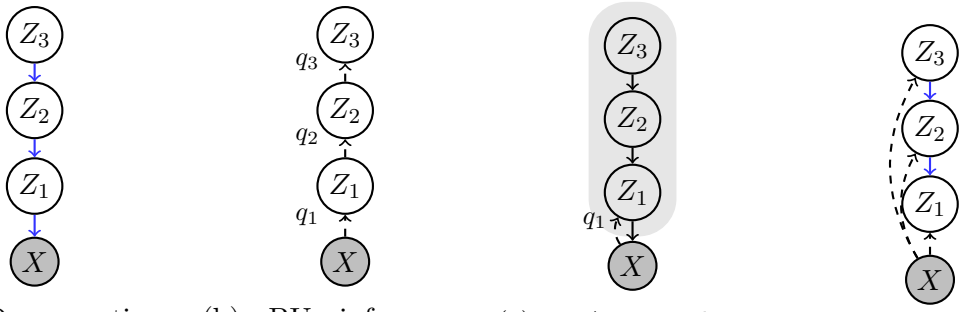
3.2.1 Generative models with deep latent hierarchies

We will consider deep generative models with Markovian hierarchies in their latent variables. Namely, where each latent variable depends exclusively on the previous one. Denoting by P_Θ the parametric model with N latent layers, where $\Theta = \theta_{1:N}$, we have:

$$p_\Theta(x, z_{1:N}) = p_{\theta_1}(x|z_1)\left[\prod_{n=2}^{N} p_{\theta_n}(z_{n-1}|z_n)\right] p_{\theta_{N+1}}(z_N) \tag{3.1}$$

where the data is $X \in \mathcal{X}$ and the latent variables are $Z_n \in \mathcal{Z}_n$ and we chose $p_{\theta_{N+1}} = \mathcal{N}(0_{\mathcal{Z}_N}, \mathcal{I}_{\mathcal{Z}_N})$. The corresponding graphical model for $N = 3$ is given Figure 3.1a.

We will be using variational inference through the WAE framework of Tolstikhin



(a) TD generative model. (b) BU inference model. (c) WAE implicit prior. (d) Ladder VAE.

Figure 3.1: (a) *Top-down* Generative model (blue lines represent generative model parameters). (b) StWAE *Bottom-up* inference model. (c) Standard WAE inference model; the inference model only map to the shallowest latent layer with implicit prior $p(z_{1:3}) = \int p(z_1|z_2)p(z_2|z_3)p(z_3)$. (d) Sønderby et al. [2016] inference model with skips connections and parameter sharing with the generative model.

et al. [2018], Bousquet et al. [2017], introducing variational distributions, $q_\Phi(z_{1:N}|x)$, where $\Phi = \phi_{1:N}$, to approximate the intractable posterior. It will be shown in Section 3.2.3 that without loss of generality, $q_\Phi(z_{1:N}|x)$ can have a Markovian latent hierarchy when following the StWAE approach. That is, without loss of generality,

$$q_\Phi(z_{1:N}|x) = q_{\phi_1}(z_1|x) \prod_{n=2}^{N} q_{\phi_n}(z_n|z_{n-1}) \tag{3.2}$$

where each $q_{\phi_n}(z_n|z_{n-1})$ is introduced iteratively by stacking WAEs at each latent layer. The corresponding graphical model for $N = 3$ is given Figure 3.1b.

We focus on this simple Markovian latent variable structure for the generative model as a proof point for StWAE. This simple modelling setup is famously difficult to train, as extensively discussed for the VAE framework (see for example Burda et al. [2015], Zhao et al. [2017], Sønderby et al. [2016]). The difficulty in training such models comes from the Markovian structure of both the inference and generative model; in particular, the difficulty of learning structure in the deeper latents. This is because, to generate samples $x \sim p_\Theta(x)$, only the joint $p_\Theta(x, z_1)$ is needed as all the deeper latent variables can be integrated out:

$$p_\Theta(x) = \int_{\mathcal{Z}_{1:N}} p_\Theta(x|z_{1:N})p_\Theta(z_{1:N})$$

$$= \int_{\mathcal{Z}_{1:N}} p_{\theta_1}(x|z_1)p_{\theta_{2:N}}(z_{1:N})$$
$$= \int_{\mathcal{Z}_1} p_{\theta_1}(x|z_1)p_{\theta_{2:N}}(z_1) \tag{3.3}$$

Whilst learning a smooth structure in each latent layer is not a strict necessity to learn a good generative model, it is however sought after if the latent is to be interpreted or used in downstream tasks. We find empirically (see Section 3.3.1) that a better generative model is also achieved when the latent hierarchy is well learnt all the way up.

Sønderby et al. [2016] sought to overcome the difficulty of learning a deep latent hierarchy by using deterministic bottom-up inference, followed by top-down inference that shared parameters with the generative model. With additional optimisation tricks (e.g. KL warm-up), their deeper latent distribution would still go unused for sufficiently deep latent hierarchies (as discussed in Maaløe et al. [2019]). In order to get deeper hierarchies of latents, Maaløe et al. [2019] introduced additional deterministic connections in the generative model as well as bidirectional inference network to facilitate the deep flow of information needed to ensure the usage of the deeper latent layers. Vahdat and Kautz [2020] achieved state-of-the-art generative performances with bespoke networks architectures and encoder parametrizations. They use additional complex optimization methods to stabilise the training of the deep models.

Whilst achieving impressive results, we believe these approaches may not be the best fit for downstream tasks where one is only interested in the inferred latent representation. Indeed, the complex encoder network with skip connections and top-down dependencies makes the information flow less interpretable as opposed to the first-order Markovian structure of StWAE. Secondly, the bottom-up and top-down dependencies come with high memory requirements at inferring time. Indeed, an extra top-down path is needed for each latent layer, requiring both additional physical memory, to store the top-down weights, and computational resources for the inference itself. This, in our view, motivates the need for methods able to train Markovian generative models with simple bottom-up inference models.

We choose the OT framework for training deep latent models due to its topological properties (see Arjovsky et al. [2017]). Still, the standard WAE encounters the same difficulties as VAEs when learning deep latent hierarchies. We thus modify the WAE objective, effectively stacking WAEs, to improve the learning of both the generative model and the inference distribution throughout the latent hierarchy.

3.2.2 Wasserstein Autoencoders

As we saw in Section 1.3, the Kantorovich formulation of the OT between the true-but-unknown data distribution P_{data} and the model distribution P_Θ, for a given cost function c, is defined by:

$$\mathrm{OT}_c(P_{data}, P_\Theta) = \inf_{\Gamma \in \mathcal{P}(P_{data}, P_\Theta)} \int_{\mathcal{X}\times\mathcal{X}} c(x, \tilde{x})\, d\Gamma(x, \tilde{x}) \tag{3.4}$$

where $\mathcal{P}(P_{data}, P_\Theta)$ is the space of all couplings of P_{data} and P_Θ; namely, the space of joint distributions Γ on $\mathcal{X} \times \mathcal{X}$ whose densities γ have marginals P_{data} and p_Θ.

In the WAE framework, the space of couplings is constrained to joint distributions Γ, with densities γ, of the form:

$$\gamma(x, \tilde{x}) = \int_{\mathcal{Z}_{1:N}} p_\Theta(\tilde{x}|z_{1:N})\, q_\Phi(z_{1:N}|x)\, p_{data}(x) \tag{3.5}$$

where $q_\Phi(z_{1:N}|x)$, for $x \in \mathcal{X}$, plays the same role as the variational distribution in variational inference.

Marginalising over $\tilde{x}$ in Equation (3.5) automatically gives $p_{data}(x)$, thus satisfying the first marginal constraint. However the second marginal constraint is not guaranteed. Due to the Markovian structure of the generative model, a sufficient condition for satisfying the second marginal constraint is:

$$\int_{\mathcal{X}} q_{\phi_1}(z_1|x)\, p_{data}(x)\, dx = p_{\theta_{2:N}}(z_1) \tag{3.6}$$

where $p_{\theta_{2:N}}(z_1)$ is the prior over Z_1:

$$p_{\theta_{2:N}}(z_1) \overset{\text{def}}{=} \int_{\mathcal{Z}_{2:N}} \left[\prod_{n=2}^{N} p_{\theta_n}(z_{n-1}|z_n) \right] p_{\theta_{N+1}}(z_N) \tag{3.7}$$

Introducing a Lagrange multiplier as in Equation 1.31, the hard constraint in Equation (3.4) can be relaxed as follow:

$$\begin{aligned} \widehat{W}_c(P_{data}, P_\Theta) = \inf_{Q_{\phi_1}(Z_1|x)} \Bigg[& \int_{\mathcal{X}\times\mathcal{X}} c(x, \tilde{x})\, \gamma(x, \tilde{x}) \\ & + \lambda_1\, \mathcal{D}_1\Big(Q_1^{\text{agg}}(Z_1),\, P_{\theta_{2:N}}(Z_1)\Big) \Bigg] \end{aligned} \tag{3.8}$$

where $\mathcal{D}_1$ is any divergence function, λ_1 a relaxation parameter, γ is defined in Equation (3.5), and $Q_1^{\text{agg}}(Z_1)$ is the aggregated posterior:

$$Q_1^{\text{agg}}(Z_1) \overset{\text{def}}{=} \int_{\mathcal{X}} Q_{\phi_1}(Z_1|x)\, p_{data}(x) \tag{3.9}$$

Given the Markovian structure of the model (Equation 3.3), all the latent variables but z_1 can be integrated out in Equation 3.5 making the joint coupling γ independent of $z_{>1}$:

$$\gamma(x, \tilde{x}) = \int_{\mathcal{Z}_1} p_{\theta_1}(\tilde{x}|z_1)\, q_{\phi_1}(z_1|x)\, p_{data}(x) \tag{3.10}$$

This results in the infimum in Equation (3.8) to be taken only over $q_{\phi_1}(z_1|x)$ instead of the full joint $q_{\phi_1}(z_{1:N}|x)$.

Whilst Equation (3.8) is in principle tractable (e.g. for Gaussian $q_{\phi_1}(z_1|x)$ and sample-based divergence function such as MMD), it only depends on the first latent Z_1. Thus it will only learn a good approximation for the joint $p_\Theta(x, z_1) = p_{\theta_1}(x|z_1)p_(z_1)$, rather than a full hierarchy with a smooth and meaningful manifold at each latent layer. We show empirically in Section 3.3.1 that Equation (3.8) is indeed insufficient for learning how to use the full hierarchy of latents, justifying the need for a new latent regularizer that can capture information across the full hierarchy.

3.2.3 Stacking WAEs for deep latent variable modelling

In the limit $\lambda_1 \to \infty$, Equation (3.8) does not depend on the choice of divergence $\mathcal{D}_1$. However, given the set of approximations used, a divergence that takes into account the smoothness of the full stack of latents will likely help the optimisation. We now explain how, by using the Wasserstein distance itself for our latent regularizer $\mathcal{D}_1$, we can derive an objective that naturally pairs up inference and generation at every level in the deep latent hierarchy. After all, the divergence in Equation (3.8) is between the intractable aggregate distribution $Q_1^{\text{agg}}(Z_1)$ from which we can only access samples, and an analytically-known distribution $P_{\theta_{2:N}}(Z_1)$. This is analogous to where we started with the Wasserstein distance between the unknown data distribution P_{data} and our fully defined model P_Θ.

Specifically, we choose $\mathcal{D}_1$ in Equation (3.8) to be the relaxed Wasserstein distance $\widehat{W}_{c_1}$, which following the same arguments as before, requires the introduction of a new variational distribution $Q_{\phi_2}(Z_2|Z_1)$:

$$\mathcal{D}_1\Big(Q_1^{\text{agg}}(Z_1),\, P_{\theta_{2:N}}(Z_1)\Big) = \inf_{Q_{\phi_2}(Z_2|z_1)} \Bigg[\int_{\mathcal{Z}_1 \times \mathcal{Z}_1} c_1(z_1, \tilde{z}_1)\, \gamma_1(z_1, \tilde{z}_1) + \lambda_2\, \mathcal{D}_2\Big(Q_2^{\text{agg}}(Z_2),\, P_{\theta_{3:N}}(Z_2)\Big) \Bigg] \tag{3.11}$$

where the prior $P_{\theta_{3:N}}$ and aggregated posterior Q_2^{agg} are defined similarly to Equation (3.7) and (3.10) respectively, swapping p_{data} with q_1^{agg} in the later. The joint is now given by

$$\gamma_1(z_1, \tilde{z}_1) \stackrel{\text{def}}{=} \int_{\mathcal{Z}_2} p_{\theta_2}(\tilde{z}_1|z_2)\, q_{\phi_2}(z_2|z_1)\, q_1^{\text{agg}}(z_1) \tag{3.12}$$

As before, without loss of generality, $Q_{\phi_2}(Z_2|Z_1)$ does not need to define a distribution over the $z_{>2}$.

The divergence $\mathcal{D}_1$ that arose in Equation (3.8) between two distributions over Z_1 is thus mapped onto the latent at the next level in the latent hierarchy, Z_2, via Equation (3.11). This process can be repeated by choosing $\mathcal{D}_2 = \widehat{W}_{c_2}$ in Equation (3.11), requiring the introduction of another variational distribution $Q_{\phi_3}(Z_3|Z_2)$ that maps

Z_2 to Z_3. Repeating this process until we get to the final layer of the hierarchical latent variable model gives the StWAE objective:

$$\mathcal{L} = \inf_{Q_\Phi(Z_{1:N}|x)} \Bigg[\Bigg[\prod_{i=1}^{N} \lambda_i \Bigg] \mathcal{D}_N \Big(Q_N^{\text{agg}}(Z_N),\, P_{\theta_{N+1}}(Z_N) \Big) + \sum_{n=0}^{N-1} \Bigg[\prod_{i=1}^{n} \lambda_i \int_{\mathcal{Z}_n \times \mathcal{Z}_n} c_n(z_n, \tilde{z}_n)\, \gamma_n(z_n, \tilde{z}_n) \Bigg] \Bigg] \tag{3.13}$$

where $(\mathcal{Z}_0, Z_0, z_0) = (\mathcal{X}, X, x)$ and by convention $\prod_{i=1}^{0} \lambda_i \overset{\text{def}}{=} 1$. The joints γ_n are defined in the same way than in Equation (3.12). The q_{ϕ_n}'s are the inference distributions introduced each time a WAE is"stacke", with their joint distribution resulting in the Markovian inference model given in Equation (3.2). The aggregated posterior distributions Q_n^{agg} are then defined as in (3.9) with q_{n-1}^{agg} in place of P_{data} and $Q_0^{\text{agg}} \overset{\text{def}}{=} P_{data}$.

Note that the StWAE objective function in Equation (3.13) still requires the specification of a divergence function at the highest latent layer $\mathcal{D}_N$, which we simply take to be the MMD as originally proposed by Tolstikhin et al. [2018]. Other choices can be made, as in Patrini et al. [2018], who choose a Wasserstein distance computed using the Sinkhorn algorithm [Cuturi, 2013a]. Whilst Patrini et al. [2018] provide a theoretical justification for the minimisation of a Wasserstein distance in the prior space, we found that it did not result in significant improvement and comes at an extra efficiency cost. Similarly, one could choose different cost functions c_n at each layer; for simplicity we take all cost functions to be the squared Euclidean distance in their respective spaces.

3.3 Experiments

We now demonstrate how the StWAE approach of Section 3.2 can be used to train deep latent hierarchies without customising the generative or inference models (e.g. no skip connections and no parameter sharing). We start by comparing the generative performances of our method with WAEs and VAEs and then analyse the latent representations learned by the different models. Finally, we qualitatively show that

StWAE can tackle relatively deep latent hierarchies to model more complex and realistic datasets.

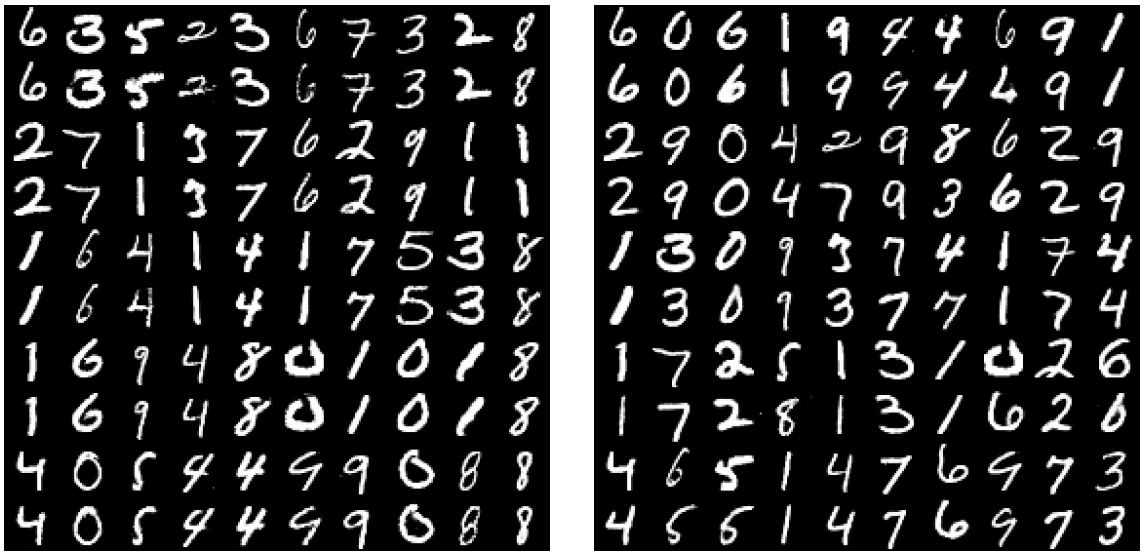
3.3.1 MNIST

Experimental setup We trained a deep hierarchical latent variable model with $N = 5$ latent layers on raw (non binarized) MNIST [LeCun and Cortes, 2010]. The latent layers have dimensions: $d_{\mathcal{Z}_1} = 32$, $d_{\mathcal{Z}_2} = 16$, $d_{\mathcal{Z}_3} = 8$, $d_{\mathcal{Z}_4} = 4$ and $d_{\mathcal{Z}_5} = 2$. We chose Gaussian distributions for both the generative and inference models, except for the bottom layer of the generative model, chosen to be deterministic, as in Tolstikhin et al. [2018], Bousquet et al. [2017]. The mean and covariance matrices are parametrized by fully connected neural networks similar to that of Sønderby et al. [2016]. The generative and inference models are parametrized as follow:

$$\begin{aligned} q_{\phi_n}(z_n|z_{n-1}) &= \mathcal{N}\big(z_n; \mu_n^q(z_{n-1}), \Sigma_n^q(z_{n-1})\big), \quad n = 1, \dots, 5 \\ p_{\theta_n}(z_{n-1}|z_n) &= \mathcal{N}\big(z_{n-1}; \mu_n^p(z_n), \Sigma_n^p(z_n)\big), \quad n = 2, \dots, 5 \\ p_{\theta_1}(x|z_1) &= \delta\big(x - f_{\theta_1}(z_1)\big) \end{aligned} \tag{3.14}$$

For both the encoder and decoder, the mean and diagonal covariance functions μ_n, Σ_n are fully-connected networks with 2 hidden layers (consider f_{θ_1} as μ_1^p). For $n = 1, 2, 3, 4, 5$, the number of units in the hidden layer is $2048, 1024, 512, 256, 128$ respectively.

For the regularization parameters, we used $\prod_{i=1}^{n} \lambda_i = \lambda_{\text{rec}}^n / d_{z_n}$ for $n = 1, \dots, 4$ (for each reconstruction term in the objective), and $\prod_{i=1}^{5} \lambda_i = \lambda_{\text{match}}$ (for the final divergence term). We then performed a grid search over the 25 pairs $(\lambda_{\text{rec}}, \lambda_{\text{match}}) \in \{0.005, 0.01, 0.05, 0.1, 0.5\} \otimes \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and found the best results (smallest loss defined in Equation (3.13)) are obtained for $\lambda_{\text{rec}} = 0.1$ and $\lambda_{\text{match}} = 10^{-4}$. For the models with KL warm-up scheme, we annealed the regularization parameter from $10e-4$ to 1 at the first 30% of training. Finally, we chose the cost function c_n to be the squared Euclidean distance: $c_n(z_n, \tilde{z}_n) = \|z_n - \tilde{z}_n\|_2^2$. The expectations in Equation (3.13) are computed analytically whenever possible and with Monte Carlo sampling otherwise. We used batch normalisation [Ioffe and Szegedy, 2015] after



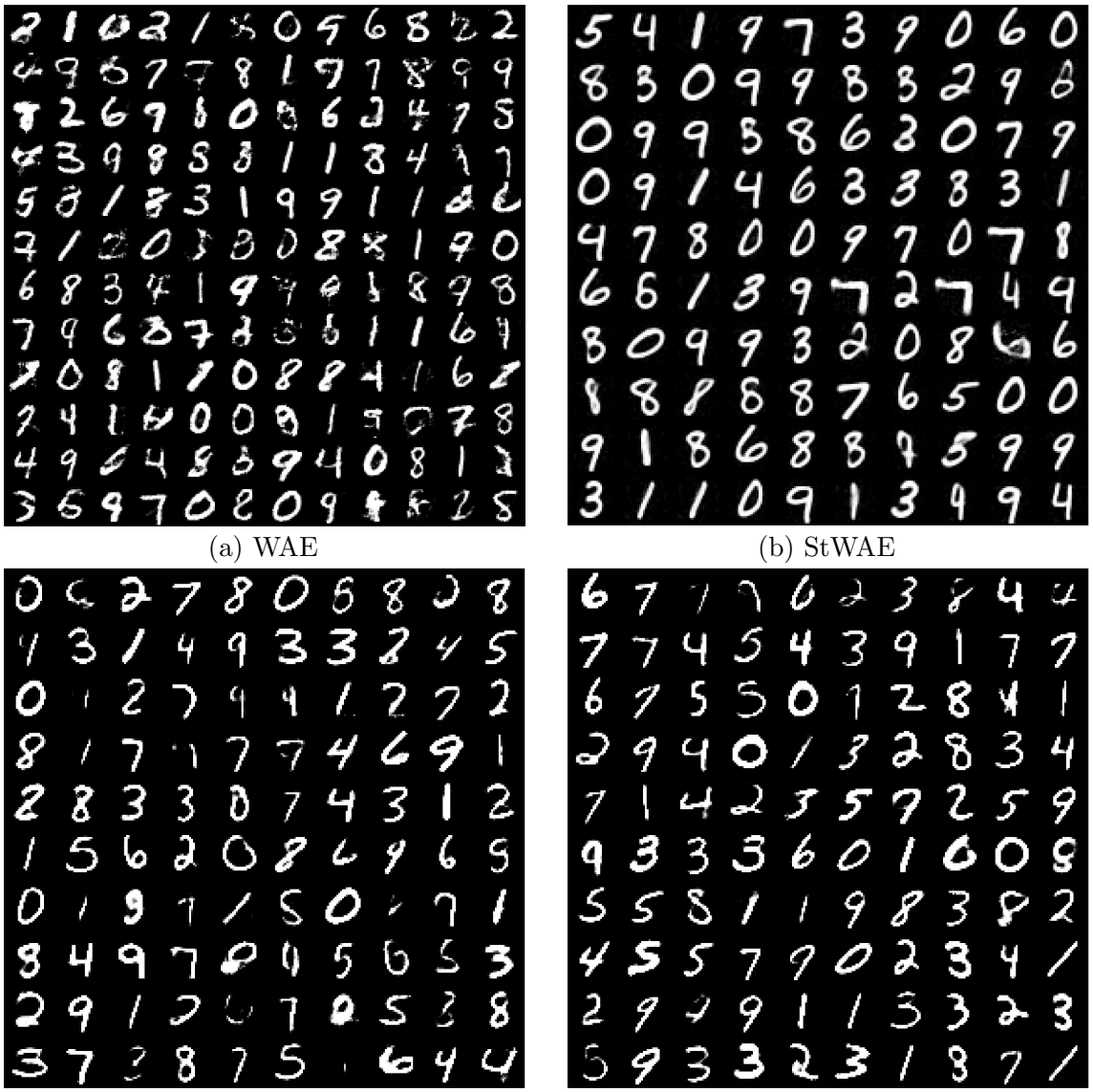
(a) (b)

Figure 3.2: Reconstruction within pairs of rows. Data is above with the corresponding reconstructions below. (a) WAE. (b) StWAE.

each hidden fully-connected layer, followed by a ReLU activation [Glorot et al., 2011] and we trained the models over $1,000$ epochs using the Adam optimiser [Kingma and Ba, 2015] with default parameters and batch size of 128.

Ablation study: StWAE versus WAE In this section, we compare StWAE and the original WAE framework for training generative models with deep hierarchical latents. In particular, we train a WAE using the objective defined in Equation (3.8) and an inference distribution given in Equation (3.2); the corresponding graphical model is shown in Figure 3.1c. This experiment can be related to the work of Tomczak and Welling [2018] and Klushyn et al. [2019] with the model trained with WAE instead of VAE. Indeed, we can rephrase it as training a 1-layer WAE whose prior over the latent variable is defined and parametrized as in Equation (3.7), and is learned in the same time than the encoder and decoder networks. However, we do not use any specific optimisation scheme nor do we constrain the structure of the prior beside the modelling choices made in Section 3.2.1, as opposed to what is done in Tomczak and Welling [2018] and Klushyn et al. [2019].

The samples generated with the standard WAE when training deep hierarchical latent variable models (Fig. 3.3a) are poor in comparison with those of the StWAE



(a) WAE (b) StWAE

(c) VAE (d) VAE+WU

Figure 3.3: Models samples. (a) Vanilla WAE. (b) StWAE. (c) VAE. (d) VAE with warm-up

Table 3.1: MSE scores.

	VAE	**VAE+WU**	**StWAE**
MSE	102.62 ± 0.27	105.06 ± 0.12	$\mathbf{50.45 \pm 1.04}$

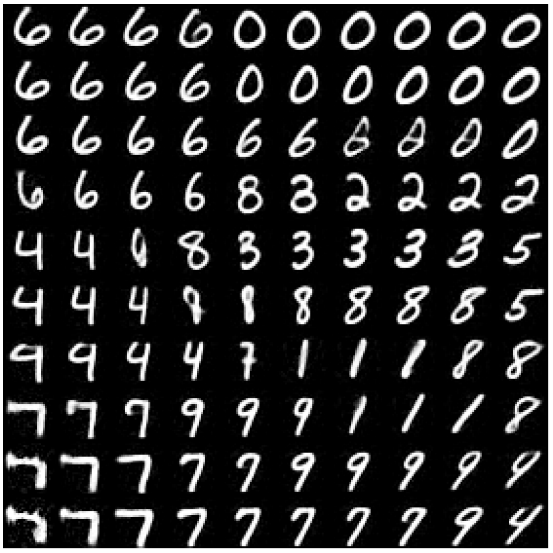
(Fig. 3.3b). However, the relatively accurate reconstructions in Figure 3.2a indicate that the model only needs the first latent layer to capture most of the data structure. This behaviour is similar to that of the Markov HVAE described in Zhao et al. [2017]. They show that, for Markov HVAEs to learn interpretable latents, one needs additional constraints beyond simply maximising the model likelihood, or in our case, the WAE objective of Tolstikhin et al. [2018] with the MMD divergence.

The lack of smooth interpolations in Figure 3.4a confirms that almost no structure has been captured in the deepest latent layer in contrast to StWAE (Fig. 3.4b). This is likely due to the fact that the standard WAE, with its objective given in Equation (3.8), is independent of the deeper latent inference distributions, thus weakening the smoothness constraint in these deeper layers.

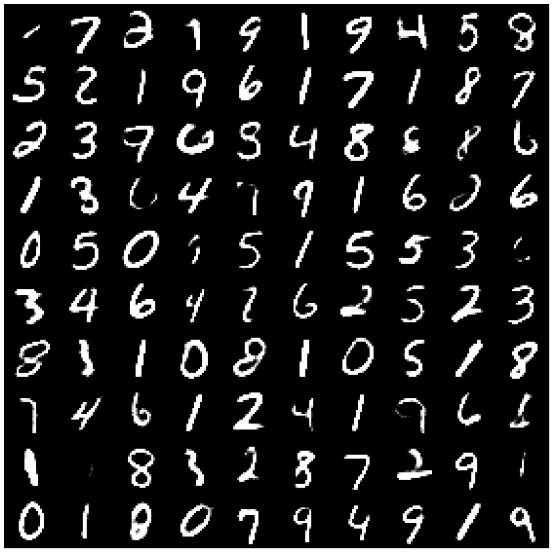
Learning a deep latent hierarchy We now compare our method with VAEs [Rezende et al., 2014, Kingma and Welling, 2014], with and without KL warm-up [Bowman et al., 2015]. Figure 3.4 shows the models samples when interpolating in the deepest latent layer $\mathcal{Z}_5$ whilst the layer-wise reconstructions are shown in Figure 3.5. Additional samples are given in Figure 3.3. Images in the interpolations are obtained by reconstructing codes in $\mathcal{Z}_5$ taken evenly in a grid varying between ± 2 standard deviations from the origin. StWAE generates more realistic samples and learns a smoother manifold than VAEs. Note that quantitative comparisons with VAE-based methods using likelihood-based scores is not possible as the likelihood of StWAE is not defined due to the deterministic decoder in the bottom layer. Whilst sample-based metrics offer an alternative for assessing the generative performances, we felt that both the visual inspection of the samples (Fig. 3.3) and the Mean Square Errors (MSE) scores (Table 3.1) were uni-vocal in the case of the MNIST dataset.

An advantage of deep latent hierarchies is their capacity to capture information

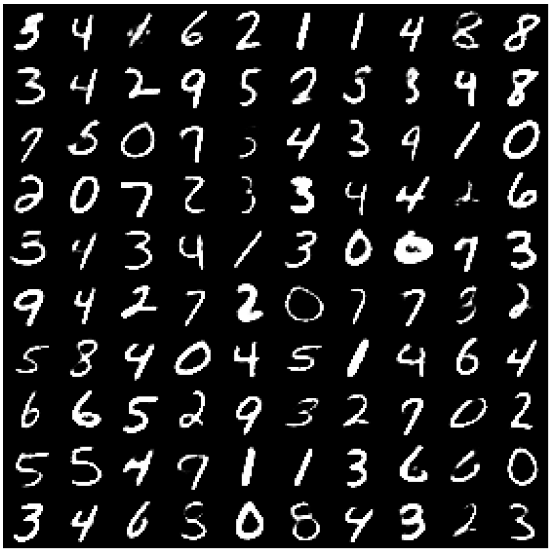
(a) WAE interpolations.



(b) StWAE interpolations.



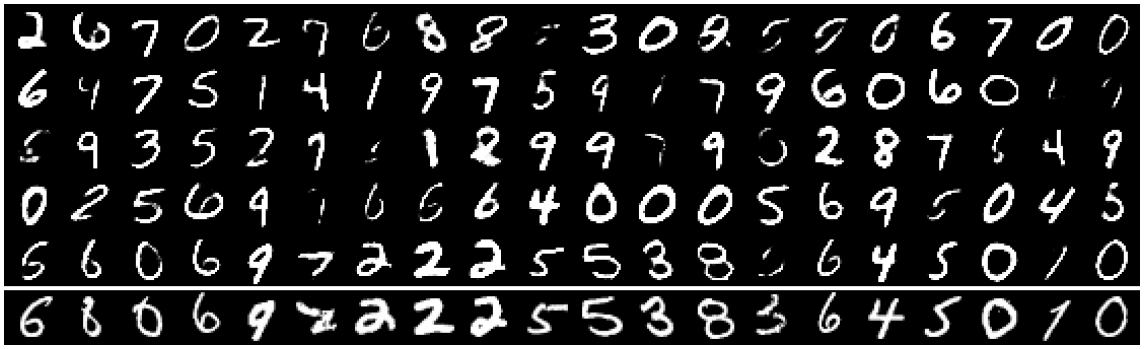
(c) VAE interpolations.



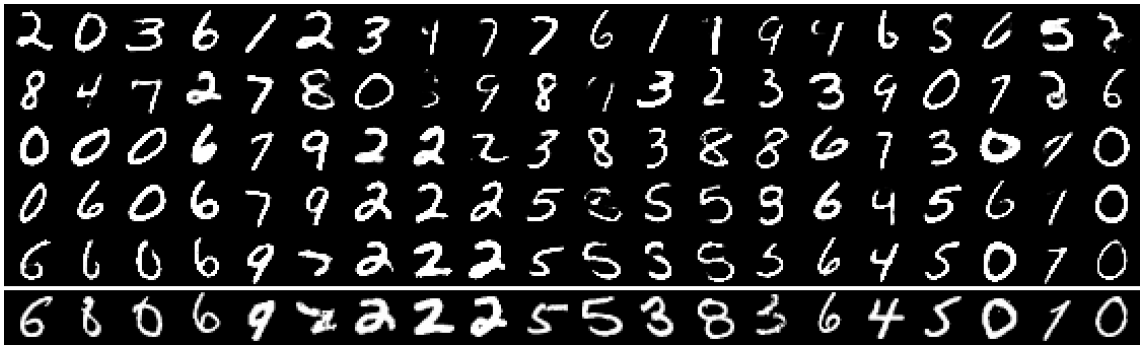
(d) VAE+WU interpolations.

Figure 3.4: 5-layers generative models. (a) Interpolations in the deepest latent layer for WAE. (b) Same than Figure 3.4a for StWAE. (c) Same than Figure 3.4a for VAE. (d) Same than Figure 3.4a for VAE with warm-up.

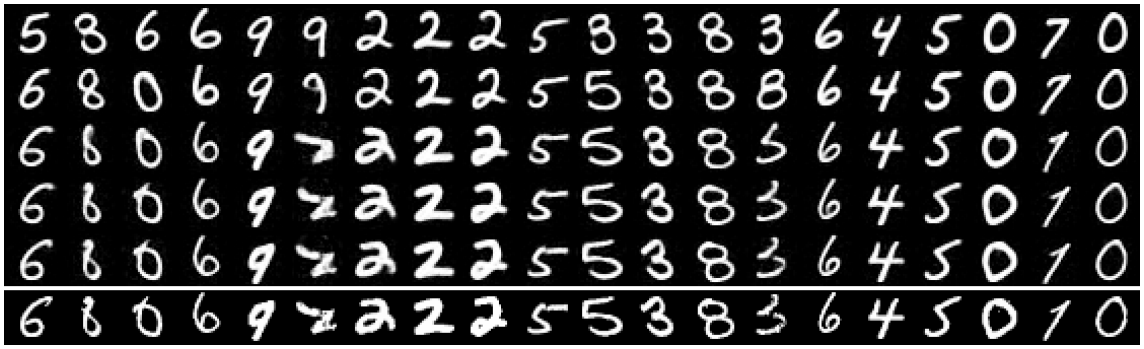
at different levels, augmenting a single layer latent space. In Figure 3.5, input images, shown in the bottom-row, are encoded and reconstructed for each latent layer. More specifically, the inputs are encoded up to the latent layer i and reconstructed from the encoded z_i using the generative model $p(x|z_1)p(z_1|z_2)\dots p(z_{i-1}|z_n)$. Row n in Figure 3.5 shows the reconstructions obtained from encoding up to layer n. In Figure 3.5c, we see that each additional encoding layer moves slightly farther away from the original input image, as it moves towards fitting the encoded points into the 2-dimensional unit normal prior. The dimensionality of the deeper latent layers is a modelling choice which determines how much information is preserved; this can be seen through the loss of information from deeper reconstructions in Figure 3.5c. Indeed, in each layer, the encoder is asked to map the incoming encodings into a lower-dimensional latent space, filtering the amount of information to keep and pass up to the deeper layer. Thus, if there is a mismatch in the dimensions between the true underlying generative process of the data and the chosen model, the encoder will have to project the encodings into lower-dimensional spaces, losing information along the way. More specifically, the deeper layers encode for more global and abstract information whilst local information is captured in the shallower layers. An inspection of the layer-wise reconstructions shows that the deepest layer (top row) encodes for the digit id, with relative uniformity in the reconstruction for the same digit id. The second layer in the hierarchy (2nd row from top) encodes the shape (see for example the different loop sizes for the 2s in the 6th, 7th and 8th columns). The third layer (3rd row from top) encodes the straw width (different width for the 2s in the 6th, 7th and 8th columns from left). The fourth and fifth layers (4th and 5th rows from top) refine iteratively the reconstructions, from coarser to finer details (for example the size and shape of the loops of the 3, 12th column from left). Understanding the way the information is encoded and passed up to the deeper layers as well as what information is encoded in each layer is an open question and could be the subject of future works.



(a) VAE

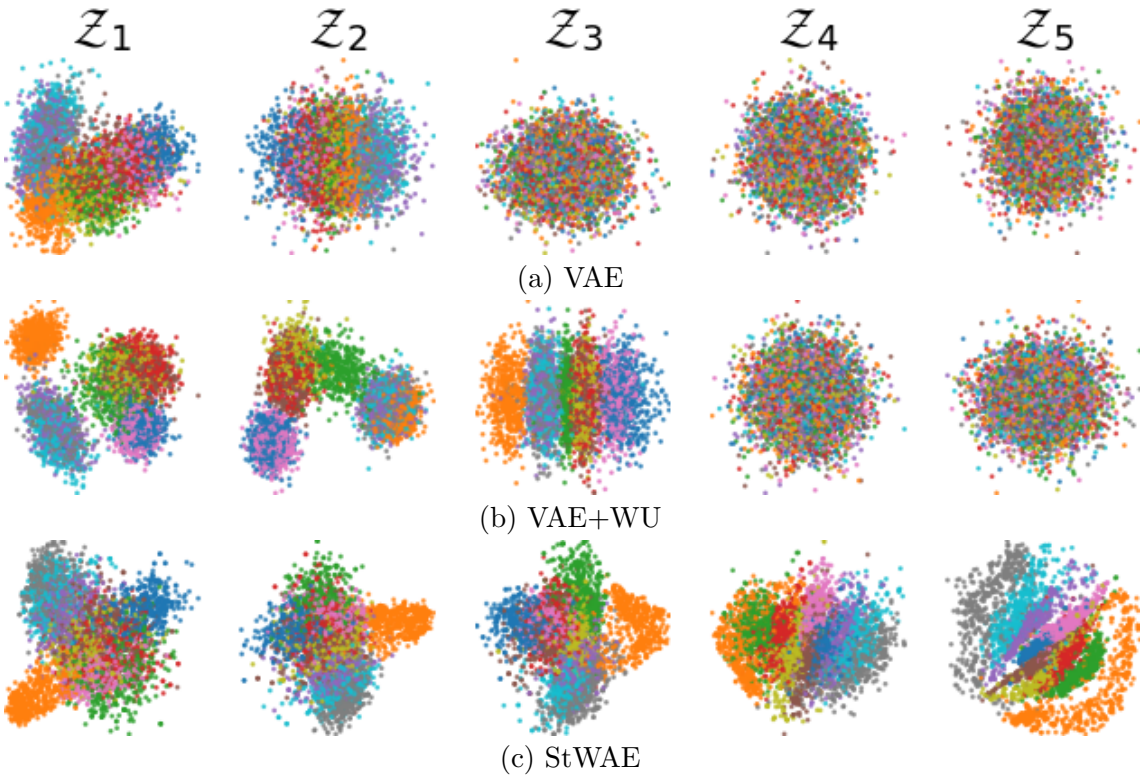


(b) VAE+WU



(c) StWAE

Figure 3.5: Layer-wise reconstructions. In each plot, the bottom-row is data and the i^{th} row from the bottom is generated using the latent codes z_i which are from the i^{th} encoding layer. (a): VAE. (b): VAE + Warm-Up. (c): StWAE.



(a) VAE

(b) VAE+WU

(c) StWAE

Figure 3.6: Visualisation of latent spaces $\mathcal{Z}_i$. Each colour corresponds to a digit label. $d_{\mathcal{Z}_5} = 2$ can be directly plotted; for higher dimensions we use PCA. (a) VAE. (b) VAE+Warm-Up. (c) StWAE.

Latent representations An important motivation for using deep hierarchical model is their capacity to learn powerful latent representations through all the latent hierarchy. As already mentioned, StWAE manages to use all of its latent layers, closely capturing the covariance structure of the data in the deeper latent layers (Fig. 3.4b), something that VAE methods struggle to accomplish as seen in Figures 3.4c and 3.4d (see also Zhao et al. [2017]). In Figure 3.6, we performed semantic clustering along the latent hierarchy, where the encoded input images are coloured with corresponding digit labels through the latent layers. We see through each layer that StWAE leverages the full hierarchy in its latents, with structured manifolds learnt at each stochastic layer (Fig. 3.6c). On the other hand, VAEs fail at learning any structure in the different latent spaces (Fig. 3.6a.) Vanilla VAE only captures structure up to the second latent layer, with the deeper latent representations collapsing to a standard normal prior, whilst using a KL warm-up scheme improves the latent representation by activating one additional latent layer (Fig. 3.6b). Note

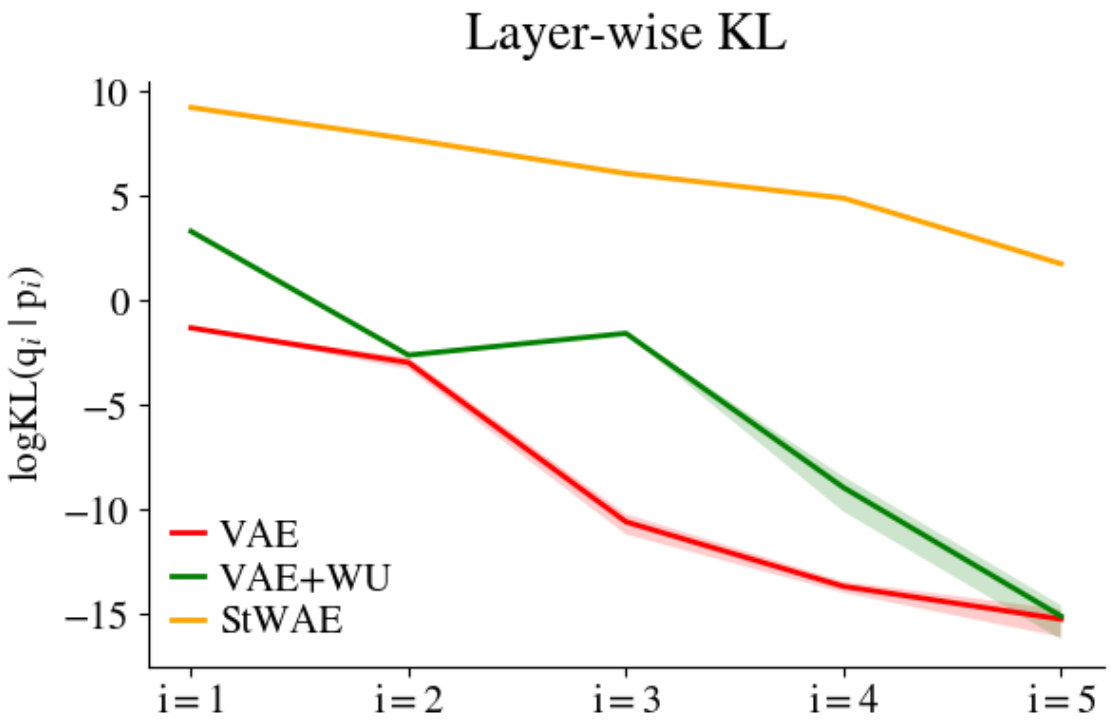


Figure 3.7: Layer-wise KL for each latent of the hierarchy averaged over 5 random runs. We re-normalise each KL term by the dimension of the corresponding latent layer. Shaded area correspond to ±1 standard deviation from the mean.

that for the shallower layers with higher dimensions (*e.g.* $d_{\mathcal{Z}_1} = 32$ or $d_{\mathcal{Z}_2} = 16$), the PCA algorithm results in a poor visualisation and other visualisation techniques could be used such as UMAP [McInnes and Healy, 2018].

More quantitatively, we can measure how much information is encoded at each layer by computing the KL between the encoded distribution $q_n(z_n|z_{n-1})$ and the prior distribution $p_{n+1}(z_n|z_{n+1})$ at each layer. Results are shown in Figure 3.7, where each KL terms is re-normalised by the dimension of the corresponding latent layer. For the VAEs methods, these terms quickly collapse to zero as we go deeper in the hierarchy, which corresponds to the collapse of the inference model, carrying no information about the data. On the contrary, in StWAE, the encoder network is able to carry information through all the hierarchy up to the top layer with significantly higher KL values. This comes from the fact that the VAE objective can be decomposed into a sum of terms minimising these KL terms at each layer whereas StWAE only matches the *aggregated* posterior to the prior at each layer. This collapse in the inference model also results in poor generative performances as

can been seen in Figures 3.4c and 3.4d (see Figure 3.3 for models samples) as well as poor reconstruction performances shown in Figures 3.5a and 3.5b. The reconstruction MSE for the different models are given Table 3.1, showing that the reconstruction performances of StWAE are substantially better (lower MSE is better) than the ones of VAEs.

3.3.2 Real world datasets

We now turn to more realistic datasets to qualitatively show that StWAE is also able to leverage deeper latent hierarchies. In particular, we trained a 6-layers and a 10-layers StWAE on Street View House Numbers (SVHN) [Netzer et al., 2011] and CelebA [Liu et al., 2015], respectively. It is worth stating that we are not aiming to achieve state-of-the-art performances on these datasets, but rather to qualitatively show that our method can learn a deep hierarchical latent representation of the data.

As Rubenstein et al. [2018b], we regularize the variance of the encoder networks with a log penalty term given by Equation (3.15):

$$\mathcal{L}_{\text{pen}} = \sum_{n=1}^{N} \lambda_n^{\Sigma} \sum_{i=1}^{d_{\mathcal{Z}_n}} |\log \Sigma_n^q[i]| \tag{3.15}$$

where Σ_n^q is the covariance matrix of the n^{th} inference network. This prevents the collapse of the encoder that leads to poor sample quality. Indeed, with the collapse of the encoder, the input data would be mapped to a latent space with dimension higher than its intrinsic dimension by a deterministic function, resulting in a poor coverage of the latent space [Rubenstein et al., 2018b]. We find that an exponentially decreasing penalty term λ_n^{Σ} (see following sections) works well for the datasets at hand with our experimental setup. This choice is motivated by the fact that the bigger the latent dimension (shallower latent layers in the hierarchy), the more likely it is that the latent dimension is larger than the data intrinsic dimension.

The mean and covariance of the inference networks and generative models for both SVHN and CelebA are parametrized by 3-layers ResNet [Kaiming et al., 2015]. A M-layers ResNet [Kaiming et al., 2015] is composed of $M-1$ convolutional blocks followed by a resampling convolution, and a residual connection. The outputs of

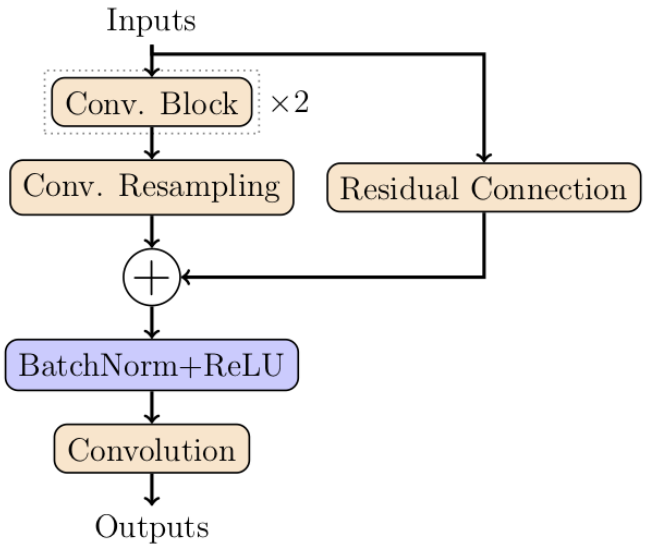


Figure 3.8: Residual network with 3 hidden convolutions.

the two tracks are added and a last operation (either fully connected or convolution layer) is applied to produce the final output. A convolutional block is composed of a convolution layer followed by batch normalisation [Ioffe and Szegedy, 2015] and a ReLU non-linearity [Glorot et al., 2011]. We also use batch normalisation and ReLU after the sum of the convolutional block and the residual connection outputs. See Figure 3.8 for an example of a 3-layers residual network with a last convolution operation. For the resampling layers, we use convolution layers with stride 2. If no resampling is performed, then the resampling convolution is a simple convolution layer with stride 1 and the skip connection performs the identity operation. The latent dimensions are then given by the dimensions and the number of features in the last convolutional operation. We chose the squared Euclidean distance as our ground cost: $c_n(z_n, \tilde{z}_n) = \|z_n - \tilde{z}_n\|_2^2$, and the expectations in Equation (3.13) are computed analytically whenever possible, otherwise using Monte Carlo sampling.

Street View House Numbers We trained a 6-layers StWAE using both the training dataset (73,257 digits) and the available additional dataset (531,131 digits). The 6-layers StWAE has Gaussian inference networks and generative models, with mean and covariance functions parametrized by 3-layers ResNet [Kaiming et al., 2015]. We trained the models over 1000 epochs using the Adam optimiser [Kingma

Table 3.2: Details of the architectures used in Section 3.3.2. Top-table: architecture of the 6-layers StWAE trained on SVHN. Bottom-table: architecture of the 10-layers StWAE trained on CelebA.

Layer i	Filters dim.	Resampling	Output dim.
Layer 1	5×5×64	down / up	16×16×2
Layer 2	3×3×64	None	16×16×1
Layer 3	3×3×96	down / up	8×8×2
Layer 4	3×3×96	None	8×8×1
Layer 5	3×3×128	down / up	4×4×2
Layer 6	3×3×128	None	4×4×1

(a) SVHN

Layer i	Filters dim.	Resampling	Output dim.
Layer 1	7×7×64	down / up	32×32×8
Layer 2	5×5×64	None	32×32×6
Layer 3	3×3×64	None	32×32×4
Layer 4	3×3×64	down / up	16×16×8
Layer 5	3×3×96	None	16×16×6
Layer 6	3×3×96	None	16×16×4
Layer 7	3×3×96	down / up	8×8×8
Layer 8	3×3×128	None	8×8×6
Layer 9	3×3×128	None	8×8×4
Layer 10	3×3×128	down / up	4×4×8

(b) CelebA

and Ba, 2015] with default parameters and batch size of 100.

For each ResNet, we used $M = 2$ convolutional blocks for each latent layers. Resampling is performed in layers $n = 1, 3, 5$ (stride 2 in the resampling convolutions). Each filters have the same size within the residual networks, and their numbers are increased (in the inference networks) or decreased (in the generative models) when resampling. More specifically, networks in layers $n = 1, 2$ have 64 convolution filters, layers $n = 3, 4$ 96 filters and layers $5, 6$ 128. We chose the number of features to be $2, 1, 2, 1, 2$ and 1 for the latent layers (last convolutional operations of the ResNets). The details are given in Table 3.2a.

For the regularization hyper parameters, we use $\prod_{i=1}^{n} \lambda_i = \lambda_{\text{rec}}^{(n-1)+1}$ for $n = 1, \dots, 5$, and $\prod_{i=1}^{6} \lambda_i = \lambda_{\text{match}}$. The effective regularization weights in Equation 3.13 scale exponentially. With this chosen regularization scheme, the layer-wise regularization term scale in $\mathcal{O}\big(\lambda_{\text{rec}}^{n}\big)$. We found that $\big(\lambda_{\text{rec}}, \lambda_{\text{match}}\big) = \big(0.5, 10\big)$ worked well

Figure 3.9: Same than Figure 3.5c for a 6-layer StWAE trained on SVHN.

with our experimental setup. As mentioned above, in order to avoid the collapse of the encoders variances, we penalised the log variance of the inference networks. We found that $\lambda_n^\Sigma = \lambda^\Sigma \cdot \mathrm{e}^{-(n-1)}$, for $n = 1, \dots, 6$, with $\lambda^\Sigma = 2.5$ worked well in our setting. More details of the network architectures can be found in Table 3.2a.

The reconstructions of the data points (along the bottom row) at each latent layer in the hierarchy are given Figure 3.9. Similarly to MNIST, we can see that the deeper latent layers may not be large enough to enable high-fidelity reconstructions. Our intention is to show that it is indeed possible to leverage the latent hierarchy, propagating the information through all the layers, rather than to model SVHN perfectly.

CelebA We trained a 10-layers StWAE. Similarly to that of SVHN, the inference and generative networks are fully-convolutional ResNet. More precisely, We used the same ResNet building blocks than previously, with $M = 2$ convolutional blocks, each filter within a ResNet block having the same size, and increasing or reducing their numbers in the last convolutional operation if resampling is performed. Layers $i = 1, 4, 7, 10$ have stride 2 with 64 filters for networks in layers $i = 1, 2, 3, 4$, 96 for networks in layers $i = 5, 6, 7$ and 128 for the remaining layers. We chose the number of features, controlling the latent dimensions to be $8, 6, 4, 2, 8, 6, 4, 8, 6, 4$ with the last latent layer having 8 feature maps. See Table 3.2b for the details.

As before, we used an exponentially decreasing regularization parameters with $\prod_{i=1}^n \lambda_i = \lambda_{\text{rec}}^{(n-1)/3+1}$ and $\prod_{i=1}^{10} \lambda_i = \lambda_{\text{match}}$ where $\left(\lambda_{\text{rec}}, \lambda_{\text{match}}\right) = \left(10^{-1}, 10^{0}\right)$. The

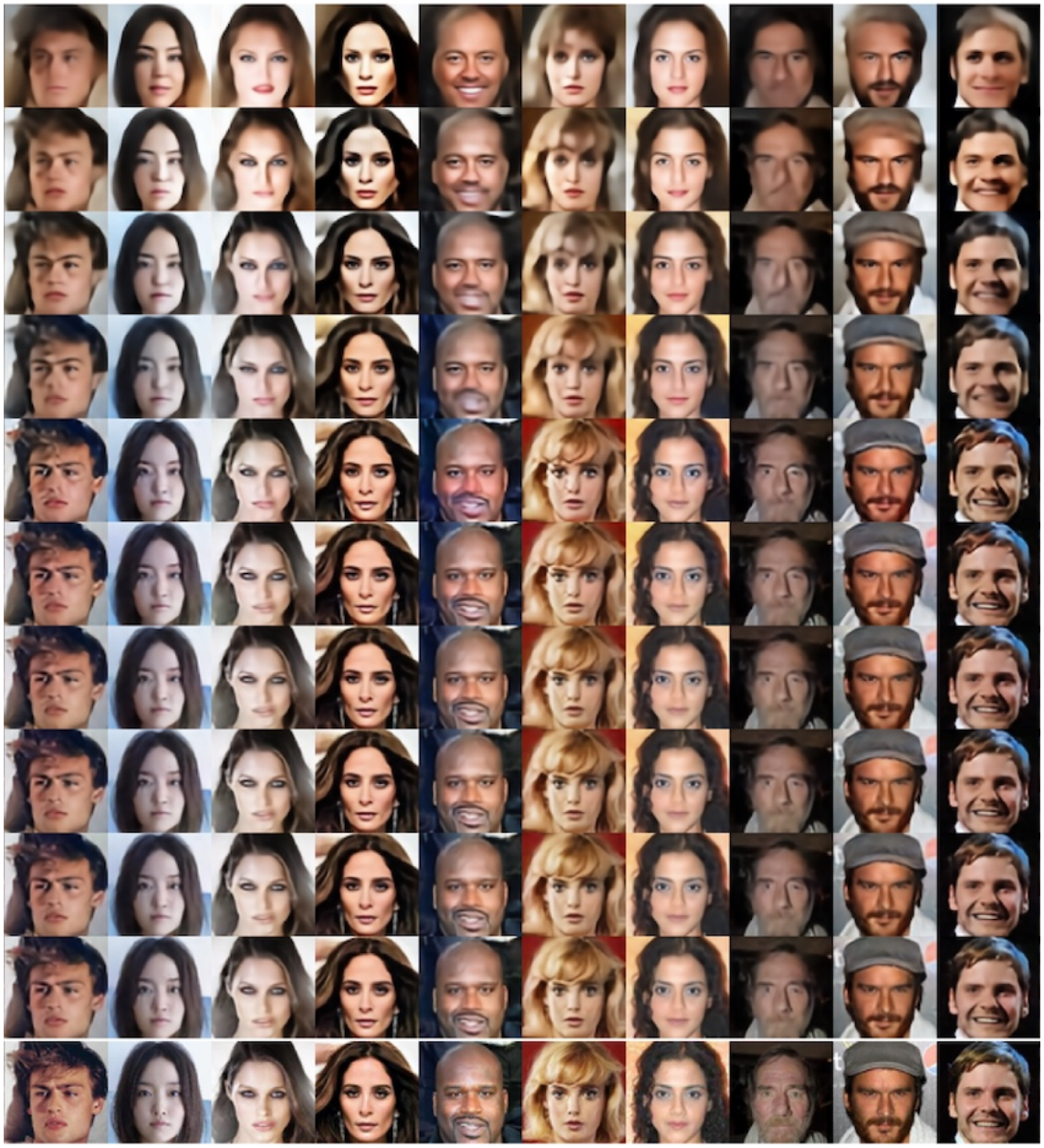


Figure 3.10: 10-layer StWAE trained on CelebA. Reconstructions for the different encoding layers, as in Figure 3.5c.

same exponential penalization scheme of the encoder variance is used with with $\lambda^{\Sigma} = 2.5$.

As with SVHN, Figure 3.10 shows that StWAE manages to use all its latent layers, up to the deepest ones. Whilst the full reconstructions (Fig. 3.10, top-row) are relatively close to the original data points, shown in the bottom-row, we can notice here again a loss of information as we go deeper in the hierarchy. In other words, the deeper the encoding, the blurrier the reconstructions. As before, this can be due to the excessive filtering of information by the encoder when going up in the hierarchy due to the miss match between the intrinsic dimension and the latent dimension.

3.4 Conclusion

In this work we introduced a novel objective function for training generative models with deep hierarchies of latent variables using Optimal Transport. Our approach recursively applies the Wasserstein distance as a regularization divergence, allowing for the stacking of WAEs for arbitrarily deep latent hierarchies. We showed that this approach enables the learning of smooth latent distributions even in deep latent hierarchies, which otherwise requires extensive model design and tweaking of the optimisation procedure to train.

Chapter 4

Learning disentangled representations with the Wasserstein Autoencoder

The work presented in this chapter was published in Gaujac et al. [2021b].

Disentangled representation learning has undoubtedly benefited from objective function surgery. However, a delicate balancing act of tuning is still required in order to trade off reconstruction fidelity versus disentanglement. Building on previous successes of penalizing the total correlation in the latent variables, we propose Total Correlation Wasserstein Autoencoder (TCWAE). Working in the WAE paradigm naturally enables the separation of the Total Correlation (TC) term [Watanabe, 1960], providing an explicit disentanglement control over the learned representation, whilst offering more flexibility in the choice of reconstruction cost. We propose two variants using different KL estimators and analyse in turn the impact of having different ground cost functions and latent regularization terms. Extensive quantitative comparisons on datasets with known generative factors shows that our methods present competitive results relative to state-of-the-art techniques. We further study the trade off between disentanglement and reconstruction on more difficult datasets with unknown generative factors, where the flexibility of the WAE paradigm leads to improved reconstructions.

4.1 Introduction

Learning representations of the data is at the heart of deep learning; the ability to interpret those representations empowers practitioners to improve the performance and robustness of their models [Bengio et al., 2013, van Steenkiste et al., 2019]. In the case where the data is underpinned by independent latent generative factors, a good representation should encode information about the data in a semantically meaningful manner with statistically independent latents encoding for each factor. Bengio et al. [2013] define a disentangled representation as having the property that a change in one dimension corresponds to a change in one factor of variation, whilst being relatively invariant to changes in other factors. Whilst many attempts to formalize this concept have been proposed [Higgins et al., 2018, Eastwood and Williams, 2018, Do and Tran, 2019], finding a principled and reproducible approach to assess disentanglement is still an open problem [Locatello et al., 2019].

Recent successful unsupervised learning methods have shown how simply modifying the ELBO objective, either re-weighting the latent regularization terms or directly regularizing the statistical dependencies in the latent variables, can be effective in learning disentangled representations. Higgins et al. [2016] and Burgess et al. [2018] control the information bottleneck capacity of VAEs [Kingma and Welling, 2014, Rezende et al., 2014] by heavily penalizing the latent regularization term. Chen et al. [2018] perform ELBO surgery to isolate the terms at the origin of disentanglement in β-VAE [Higgins et al., 2016], improving the reconstruction-disentanglement trade off. Esmaeili et al. [2018] further improve the reconstruction capacity of β-TCVAE [Chen et al., 2018] by introducing structural dependencies both between groups of variables and between variables within each group. Alternatively, directly regularizing the aggregated posterior to the prior with density-free divergences [Zhao et al., 2019] or moments matching [Kumar et al., 2018], or simply penalizing a high TC in the latent has shown good disentanglement performance [Kim and Mnih, 2018].

In fact, information theory has been a fertile ground to tackle representation learning. Achille and Soatto [2018] re-interpret VAEs from an Information Bottleneck view [Tishby et al., 1999], re-phrasing it as a trade off between sufficiency and minimality

of the representation and regularizing a pseudo TC between the aggregated posterior and the true conditional posterior. Similarly, Gao et al. [2019] use the principle of total Correlation Explanation (CorEX) [Ver Steeg and Galstyan, 2014] and maximize the mutual information between the observation and a subset of anchor latent points. Maximizing the Mutual Information (MI) between the observation and the latent has been broadly used [Van den Oord et al., 2018, Hjelm et al., 2019, Bachman et al., 2019, Tschannen et al., 2020], showing encouraging results in representation learning. However, Tschannen et al. [2020] argued that MI maximization alone cannot explain the disentanglement performance of these methods.

Building on developments in Optimal Transport (OT) [Villani, 2008], Tolstikhin et al. [2018], Bousquet et al. [2017] introduced the Wasserstein Autoencoder (WAE), an alternative to VAEs for learning generative models. WAE maps the data into a (low-dimensional) latent space whilst regularizing the averaged encoding distribution. This is in contrast with VAEs where the posterior is regularized at each data point, and allows the encoding distribution to capture meaningful information from the data whilst still matching the prior in average. Interestingly, by directly regularizing the aggregated posterior, WAE hints at more explicit control on the way the information is encoded, and thus better disentanglement. The reconstruction term of the WAE allows for any cost function on the observation space and non-deterministic decoders. This removes the need of random decoders for which the log density is tractable and derivable (with regard to the decoder parameters) as it is the case in VAEs. The combination of non-degenerated Gaussian decoders with the Kullback-Leibler (KL) divergence acting as the posterior regularizer is viewed as the cause of the samples blurriness in VAEs [Bousquet et al., 2017].

Few works have sought to use WAE for disentanglement learning. Initial experiments from Rubenstein et al. [2018a] showed encouraging results but did not fully leverage the flexibility offered by WAEs. The authors simply used the original WAE objective with cross-entropy reconstruction cost without studying the impact of using different reconstruction terms and latent regularization functions. Mirroring the KL-based TC, Xiao and Wang [2019] introduced Wasserstein Total Correlation (WTC). Resorting to the triangle inequality, they decomposed the latent regular-

ization term of the WAE into the sum of two WTC terms. However, WTC lacks an information-theoretic interpretation such as the one offered by the TC and the authors resorted to the dual formulation of the 1-Wasserstein distance to approximate the WTC, adversarially training two critics.

In this work, following the success of regularizing the TC in disentanglement, we propose to use the KL divergence as the latent regularization function in the WAE. We introduce the Total Correlation WAE (TCWAE) with an explicit dependency on the TC of the aggregated posterior. We study separately the impact of using a different ground cost function in the reconstruction term and the impact of a different composition of the latent regularization term. Performing extensive comparisons with successful methods on a number of datasets, we found that TCWAEs achieve competitive disentanglement performance whilst improving modelling performance by allowing flexibility in the choice of reconstruction cost.

4.2 Importance of Total correlation in disentanglement

4.2.1 Total correlation

The TC of a random vector $Z \in \mathcal{Z}$ under P is defined by

$$\mathbf{TC}(P(Z)) \stackrel{\text{def}}{=} \sum_{i=1}^{d_Z} H_{P_i}(Z_i) - H_P(Z) \tag{4.1}$$

where $p_i(z_i)$ is the marginal density over only the i-th component z_i, and $H_P(Z) \stackrel{\text{def}}{=} -\mathbb{E}_P \log p(Z)$ is the Shannon differential entropy, which encodes the information contained in Z under P. Since

$$\sum_{i=1}^{d_Z} H_{P_i}(Z_i) \leq H_P(Z) \tag{4.2}$$

with equality when the marginals Z_i are mutually independent. The TC can be interpreted as the loss of information when assuming mutual independence of the

Z_i; namely, it measures the mutual dependence of the marginals. Thus, in the context of disentanglement learning, we seek a low TC of the aggregated posterior, $p(z) = \int_{\mathcal{X}} p(z|x)\, p(x)\, dx$, which forces the model to encode the data into statistically independent latent codes. High MI between the data and the latent is then obtained when the posterior, $p(z|x)$, manages to capture relevant information from the data.

4.2.2 Total correlation in ELBO

We consider latent generative models $p_\theta(x) = \int_{\mathcal{Z}} p_\theta(x|z)\, p(z)\, dz$ with prior $p(z)$ and decoder network, $p_\theta(x|z)$, parametrized by θ. As we saw in Section 1.1.2, VAEs approximate the intractable posterior $p_\theta(z|x)$ by introducing an encoding distribution (the encoder), $q_\phi(z|x)$. The encoder and decoder are simultaneously learned by optimizing the variational lower-bound or ELBO (Equation (1.7)), with regards to their respective parameters θ and ϕ. For convenience we recall here the definition of the ELBO for a single observation x:

$$\mathcal{L}_{ELBO}(x, \theta, \phi) = \int_{\mathcal{Z}} q_\phi(z|x) \log p_\theta(x|z)\; dz - \mathbf{KL}\big(Q_\phi(Z|x) \parallel P(Z)\big) \tag{4.3}$$

Following Hoffman and Johnson [2016], we treat the data index, n, as a uniform random variable over $\{1, \ldots, N\}$: $p(n) = \frac{1}{N}$. We can then define respectively the posterior, *joint* posterior and *aggregated* posterior:

$$q_\phi(z|n) = q_\phi(z|x_n), \quad q_\phi(z, n) = q_\phi(z|n)\, p(n), \quad q_\phi(z) = \sum_{n=1}^{N} q_\phi(z|n)\, p(n) \tag{4.4}$$

Using this notation, we decompose the KL term in Equation (4.3) into the sum of ⓘ, an *index-code mutual information* term, and ⓘⓘ, a *marginal KL* term, with:

$$ⓘ = \mathbf{KL}\big(Q_\phi(Z, N) \parallel Q_\phi(Z)P(N)\big) \quad \text{and} \quad ⓘⓘ = \mathbf{KL}\big(Q_\phi(Z) \parallel P(Z)\big) \tag{4.5}$$

The index-code mutual information (index-code MI) represents the MI between the data and the latent under the joint distribution $q(z, n)$, and the marginal KL enforces the aggregated posterior to match the prior. The marginal KL term plays

an important role in disentanglement. Indeed, it pushes the encoder network to match the prior when *averaged*, as opposed to matching the prior for each data point. Combined with a factorized prior $p(z) = \prod_i p_i(z_i)$, as it is often the case, the aggregated posterior is forced to factorize and align with the axis of the prior. More explicitly, the marginal KL term in Equation (4.5) can be decomposed as sum of a *Total Correlation* term and a *dimension-wise KL* term:

$$\text{ii} = \mathbf{TC}\Big(Q_\phi(Z)\Big) + \sum_{i=1}^{d_\mathcal{Z}} \mathbf{KL}\big(Q_{\phi,i}(Z_i) \parallel P_i(Z_i)\big) \tag{4.6}$$

Thus maximizing the ELBO implicitly minimizes the TC of the aggregated posterior, enforcing the aggregated posterior to disentangle as Higgins et al. [2016] and Burgess et al. [2018] observed when strongly penalizing the KL term in Equation (4.3). Chen et al. [2018] leverage the KL decomposition in Equation (4.6) by refining the heavy latent penalization to the TC only. However, the index-code MI term of Equation (4.5) seems to have little to no role in disentanglement, potentially harming the reconstruction performance as we will see in Section 4.4.1.

4.3 Is WAE naturally good at disentangling?

4.3.1 WAE

Re-using the notations from the previous sections, we recall here the WAE formulation of Tolstikhin et al. [2018], Bousquet et al. [2017]:

$$W_{\mathcal{D},c}(\theta,\phi) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{P_D(X)} \mathop{\mathbb{E}}_{Q_\phi(Z|X)} \mathop{\mathbb{E}}_{P_\theta(\tilde{X}|Z)} \big[c(X,\tilde{X})\big] + \lambda\, \mathcal{D}\Big(Q_\phi(Z) \parallel P(Z)\Big) \tag{4.7}$$

where $\mathcal{D}$ is any divergence function and λ a relaxation parameter. The decoder, $p_\theta(\tilde{x}|z)$, and the encoder, $q_\phi(z|x)$, are optimized simultaneously by dropping the closed-form minimization over the encoder network, with standard gradient descent methods.

Whilst objective (4.7) resembles the ELBO with a reconstruction term and a latent regularization term, WAE explicitly penalizes the aggregate posterior as

opposed to VAE. As seen in Equation (4.5), the TC dependency of the ELBO only appears implicitly and competes with other terms and especially, the index-code MI. Following Section 4.2.2, this motivates, the use of WAE in disentanglement learning.

Another important difference lies in the functional form of the reconstruction cost in the reconstruction term. Indeed, we saw previously that WAE allows for more flexibility in the reconstruction term, and in particular, it allows for cost functions both better suited to the data at hand and for the use of deterministic decoder networks [Tolstikhin et al., 2018, Frogner et al., 2015]. This can potentially result in an improved reconstruction-disentanglement trade off as we empirically find in Sections 4.4.2 and 4.4.1.

4.3.2 TCWAE

For the simplicity of notations, we drop the explicit dependency of the distributions on their parameters θ and ϕ throughout this section.

We chose the divergence function $\mathcal{D}$ in Equation (4.7) to be the KL divergence and assume a factorized prior (*e.g.* $p(z) = \mathcal{N}(0_{d_\mathcal{Z}}, \mathcal{I}_{d_\mathcal{Z}})$), thus obtaining the same decomposition as in Equation (4.6). Similarly to Chen et al. [2018], we use different parameters, β and γ, for each term in the decomposition in Equation (4.6) of the latent regularization term from Equation (4.7), obtaining our TCWAE objective:

$$W_{TC} \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{P(X_n)Q(Z|X_n)} \Big[\mathop{\mathbb{E}}_{P(\tilde{X}_n|Z)} [c(X_n, \tilde{X}_n)] \Big] \tag{4.8}$$
$$+ \beta \mathbf{KL}\big(Q(Z) \parallel \prod_{i=1}^{d_\mathcal{Z}} Q_i(Z_i)\big) + \gamma \sum_{i=1}^{d_\mathcal{Z}} \mathbf{KL}\big(Q_i(Z_i) \parallel P_i(Z_i)\big)$$

TCWAE is identical to the WAE objective with KL divergence when $\lambda = \beta = \gamma$; and provides an upper-bound with $\min(\beta, \gamma) = \lambda$.

Equation (4.8) can be directly related to the β-TCVAE objective of Chen et al. [2018]:

$$\mathcal{L}_{\beta TC} \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{P(X_n)Q(Z|X_n)} \Big[-\log p(X_n|Z) \Big] + \alpha \mathbf{KL}\big(Q(Z, N) \parallel Q(Z)P(N)\big) \tag{4.9}$$

$$+ \beta \mathbf{KL}\big(Q(Z) \parallel \prod_{i=1}^{d_Z} Q_i(Z_i)\big) + \gamma \sum_{i=1}^{d_Z} \mathbf{KL}\big(Q_i(Z_i) \parallel P_i(Z_i)\big)$$

As mentioned above, the differences are the absence of index-code MI and a different reconstruction cost function. Setting $\alpha = 0$ in Equation (4.9) makes the two latent regularizations match but breaks the inequality provided by the ELBO (Equation (4.3)). Matching the two reconstruction terms would be possible if we could find a ground cost function c such that $\mathbb{E}_{P(\tilde{X}_n|Z)} c(x_n, \tilde{X}_n) = -\log p(x_n|Z)$ for all x_n.

4.3.3 Estimators

With the goal of being grounded by information theory and earlier works on disentanglement, using the KL as the latent divergence function, as opposed to other sample-based divergences [Tolstikhin et al., 2018, Patrini et al., 2018], presents its own challenges. Indeed, the KL terms are intractable; in particular, we will need estimators to approximate the entropy terms. We propose to use two estimators, one based on importance weight-sampling [Chen et al., 2018], the other on adversarial estimation using the density-ratio trick [Kim and Mnih, 2018].

TCWAE-MWS Chen et al. [2018] propose to estimate the intractable terms $\mathbb{E}_Q \log q$ and $\mathbb{E}_{Q_i} \log q_i$ within the KL in Equation (4.8) with Minibatch-Weighted Sampling (MWS). Considering a batch of observation $\{x_1, \dots x_{N_{\text{batch}}}\}$, they sample the latent codes $z_n \sim Q(Z|x_n)$ and compute:

$$\mathop{\mathbb{E}}_{Q(Z)} \log q(Z) \approx \frac{1}{N_{\text{batch}}} \sum_{n=1}^{N_{\text{batch}}} \log \frac{1}{N \times N_{\text{batch}}} \sum_{m=1}^{N_{\text{batch}}} q(z_m|x_n) \tag{4.10}$$

This estimator, whilst being easily computed from samples, is a biased estimator of $\mathbb{E}_q \log q(Z)$. Chen et al. [2018] also proposed an unbiased version, the Minibatch-Stratified Sampling (MSS). However, they found that it did not result in improved performance, and thus, they and we choose to use the simpler MWS estimator. We call the resulting algorithm the TCWAE-MWS. Other sampled-based estimators of the entropy or the KL divergence have been proposed [Rubenstein et al., 2019,

Esmaeili et al., 2018]. However, we choose the solution of Chen et al. [2018] for i) its simplicity and ii) the similarities between the TCWAE and β-TCVAE objectives.

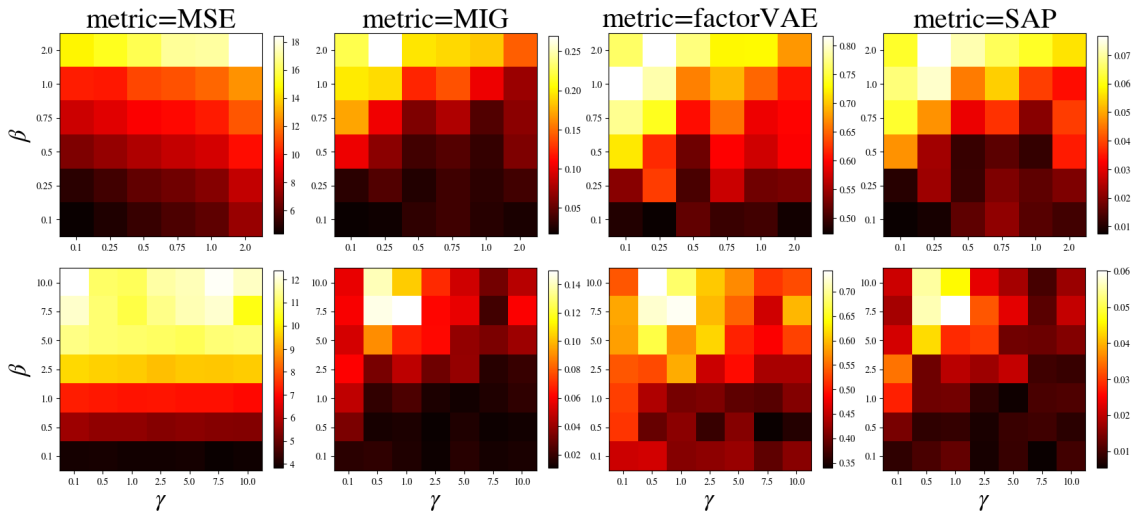
TCWAE-GAN A different approach, similar in spirit to the WAE-GAN originally proposed by Tolstikhin et al. [2018], Bousquet et al. [2017], is based on adversarial-training. Whilst Tolstikhin et al. [2018], Bousquet et al. [2017] use the adversarial training to approximate the JS divergence, Kim and Mnih [2018] use the density-ratio trick and adversarial training to estimate the intractable terms in Equation (4.8). The density-ratio trick [Nguyen et al., 2008, Sugiyama et al., 2011] estimates the KL divergence as:

$$\mathbf{KL}\big(Q(Z) \parallel \prod_{i=1}^{d_\mathcal{Z}} Q_i(Z_i)\big) \approx \mathop{\mathbb{E}}_{Q(Z)} \log \frac{D(Z)}{1 - D(Z)} \tag{4.11}$$

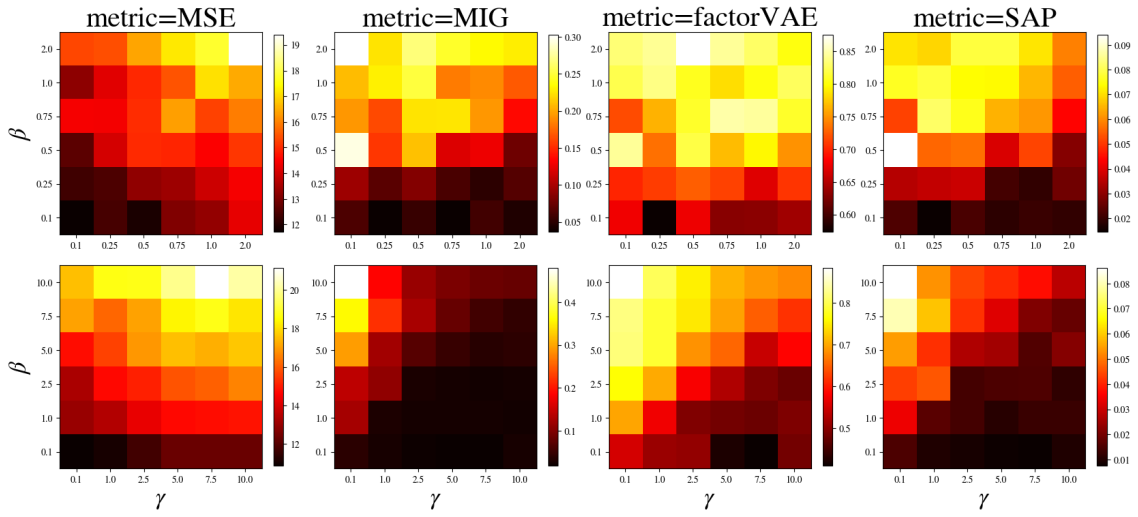
where D plays the same role as the discriminator in GANs and outputs an estimate of the probability that z is sampled from $Q(Z)$ and not from $\prod_{i=1}^{d_\mathcal{Z}} Q_i(Z_i)$. Given that we can easily sample from $q(z)$, we can use Monte-Carlo sampling to estimate the expectation in Equation (4.11). The discriminator D is adversarially trained alongside the decoder and encoder networks. We call this adversarial version the TCWAE-GAN.

4.4 Experiments

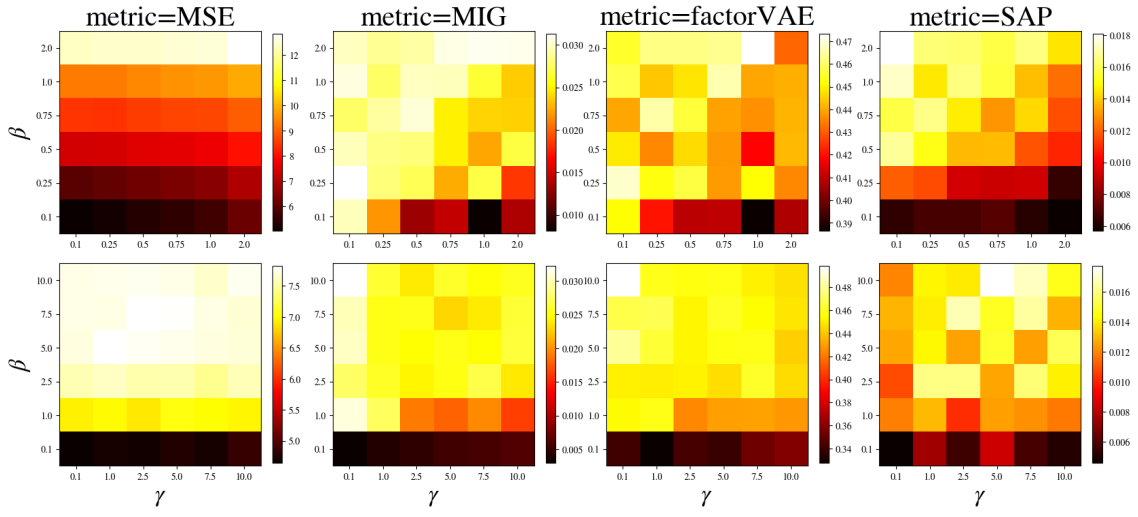
We perform a series of quantitative and qualitative experiments, isolating in turn the effect of the ground cost and latent regularization functions in TCWAE. We found that whilst the absence of index-code MI in TCWAEs does not impact the disentanglement performance, using the square Euclidean distance as our ground cost function improves the reconstruction whilst retaining the same disentanglement. Finally, we compare our methods with the benchmarks β-TCVAE and FactorVAE, both quantitatively on toy datasets and qualitatively more challenging datasets with unknown generative factors.



(a) dSprites



(b) 3D shapes



(c) smallNORB

Figure 4.1: Reconstruction and disentanglement scores heat maps. Top row corresponds to TCWAE-MWS and bottom row to TCWAE-GAN. Smaller MSE and higher disentanglement scores represent better models.

4.4.1 Quantitative analysis: disentanglement on toy datasets

In this section we train the different methods on the dSprites [Matthey et al., 2017], 3D shapes [Kim and Mnih, 2018] and smallNORB [LeCun et al., 2004] datasets whose ground-truth generative-factors are given in Table B.1. We use the Mutual Information Gap (MIG, [Chen et al., 2018]), the factorVAE metric [Kim and Mnih, 2018] and the Separated Attribute Predictability score (SAP, [Kumar et al., 2018]) to asses the disentanglement performances (see Locatello et al. [2019] for the implementation). We assess the reconstruction performances with the Mean Square Error (MSE) of the reconstructions. See Appendix B.1 for more details on the experimental setups.

γ tuning Mirroring Chen et al. [2018], Kim and Mnih [2018], we first tune γ, responsible for the dimension-wise KL regularization, subsequently focusing on the role of the TC term in the disentanglement performance. We trained the TCWAEs with six different values for each parameter, resulting in thirty-six different models. We show the heat map for the different datasets in Figure 4.1. We observe that whilst β controls the trade off between reconstruction and disentanglement, γ affects the range achievable when varying β. We choose γ with the best overall mean scores when aggregated over all the different β. See Appendix B.2 for more details.

For each method and dataset, the violin plots of the different metrics for five random runs are given Figures 4.2, 4.3 and 4.4. As argued by Locatello et al. [2019], the disentanglement scores are relatively sensible to hyper parameters tuning. The chosen values for each method and dataset are given in Table B.5 and fixed in all the following experiments.

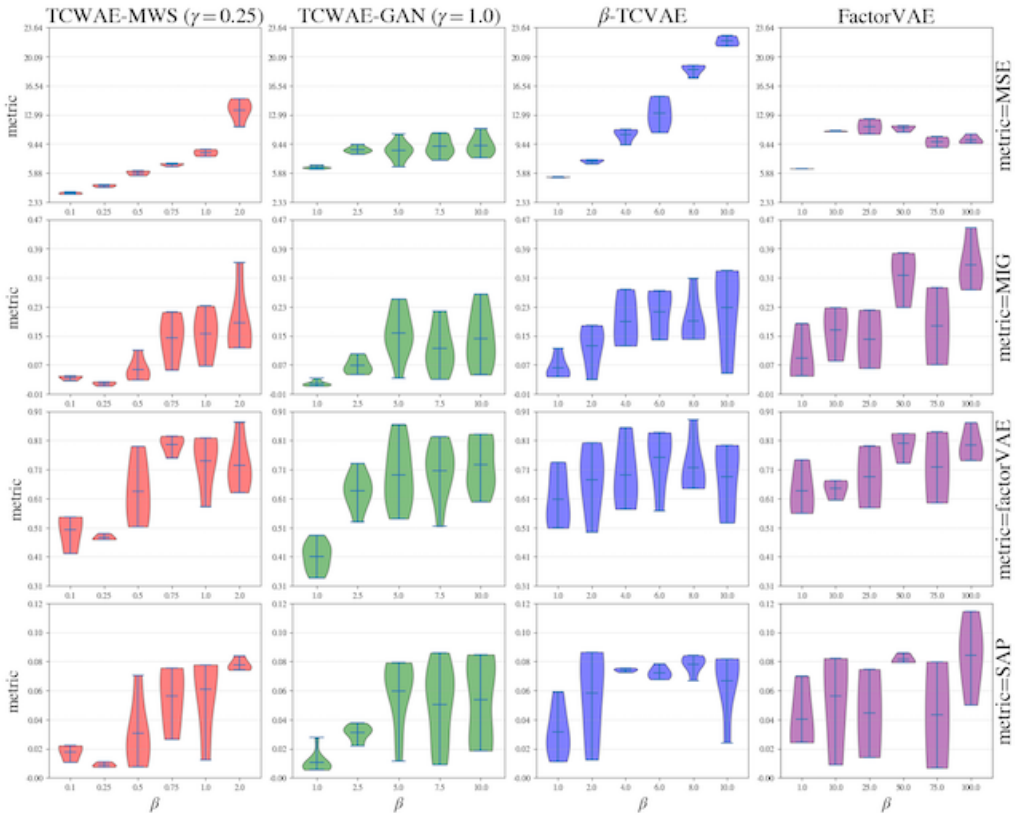


Figure 4.2: Disentanglement versus γ violin plots on dSprites

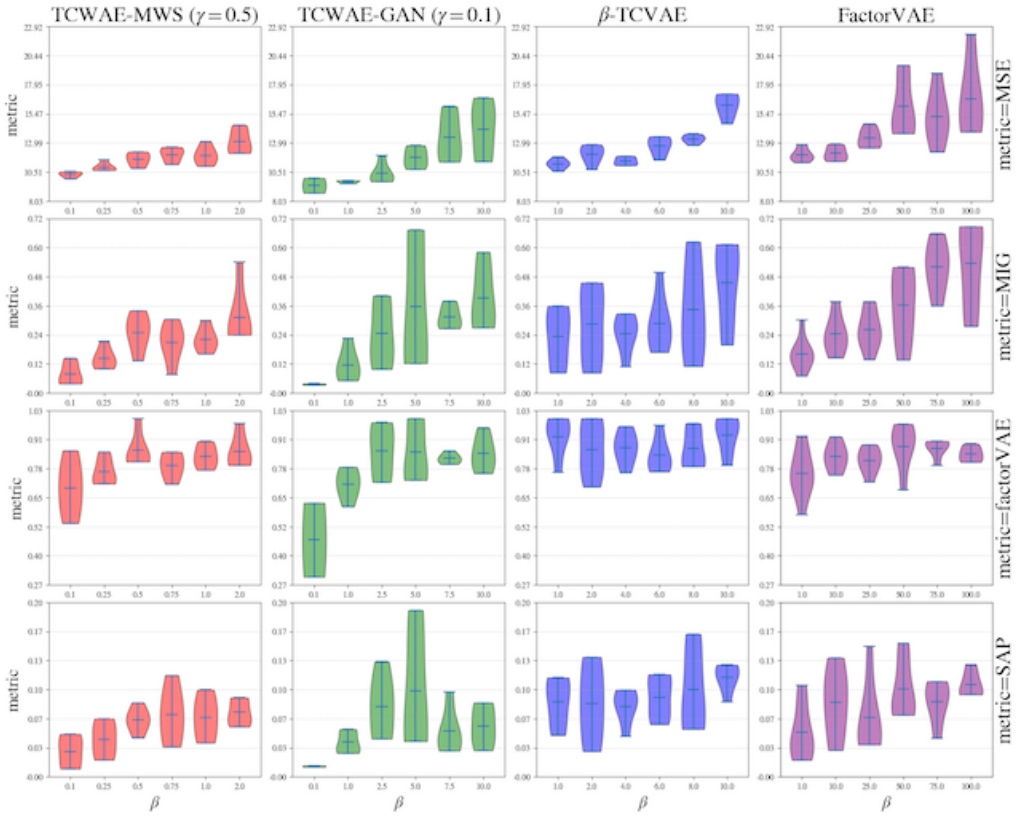


Figure 4.3: Disentanglement versus γ violin plots on 3D shapes

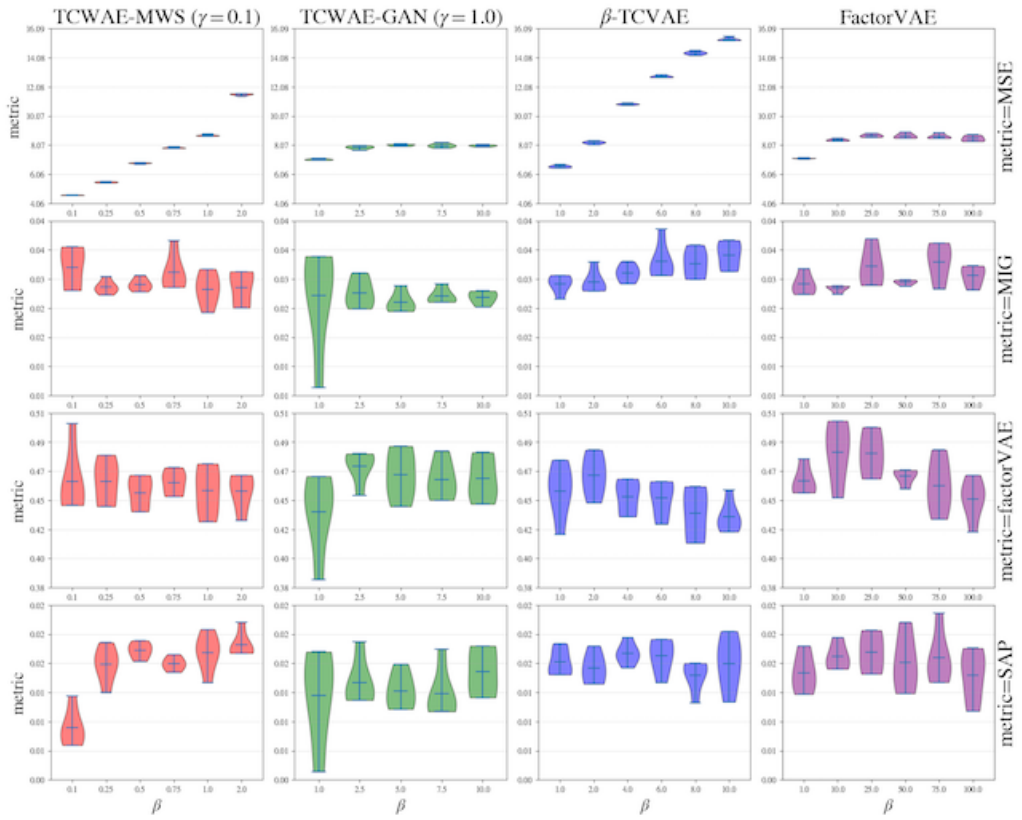


Figure 4.4: Disentanglement versus γ violin plots on smallNORB

Ablation study of the ground cost function We study the impact of using the square euclidean distance as the ground cost function as opposed to the cross-entropy cost as in the ELBO. When using the cross-entropy, the TCWAE objective is equivalent to the one defined in Equation (4.9) with $\alpha = 0$; in this case, TCWAE-MWS simply boils down to β-TCVAE with no index-code MI term.

We train two sets of TCWAE-MWS, one with the square euclidean distance and one with the cross-entropy, for different β and compare the reconstruction-disentanglement trade off by plotting the different disentanglement scores versus the MSE in Figure 4.5. We see that the square euclidean distance provides better reconstructions on all datasets as measured by the MSE (points on the left side of the plots have lower MSE). Models trained with the square euclidean distance show similar disentanglement performance for smallNORB and better ones for dSprites. In the case of 3D shapes, some disentanglement performance has been traded away for better reconstruction. We would argue that this simply comes from the range of parameters used for both β and γ. These results show that the flexibility in

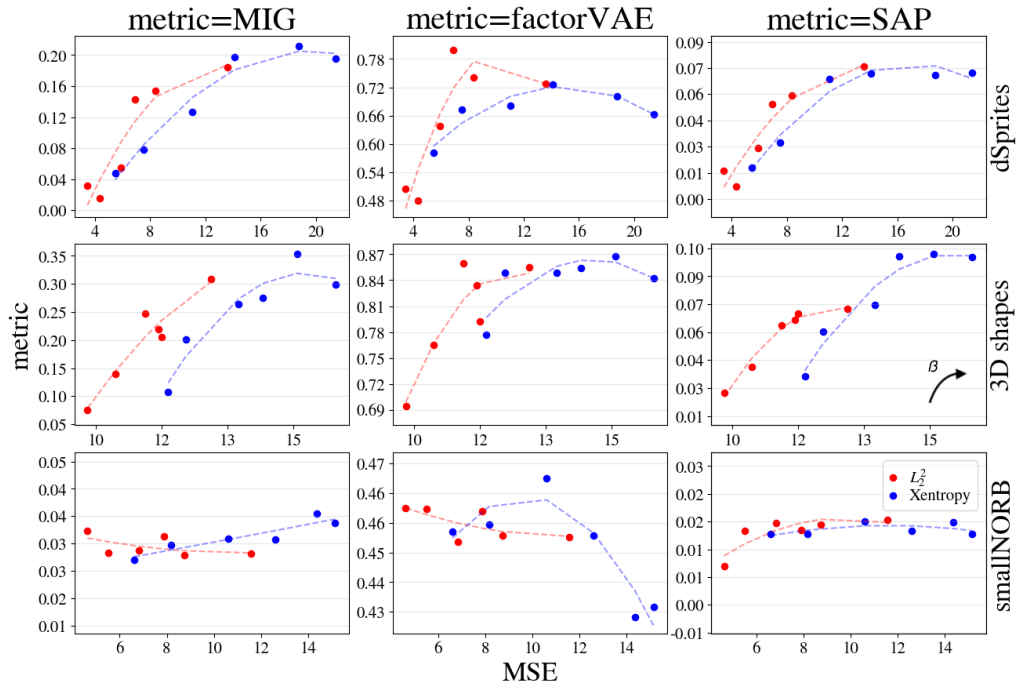


Figure 4.5: Disentanglement versus reconstruction for TCWAE-MWS with square euclidean distance (red) and cross-entropy (blue) cost. Points represent different models with different β (quadratic regression represented by the dashed lines). Low MSE and high scores (top-left corner) is better. Higher TC penalisation (bigger β) results in higher MSE and higher scores.

choice of the ground cost function of the reconstruction term of TCWAE offers better reconstruction-disentanglement trade off by improving the reconstruction whilst exhibiting competitive disentanglement.

Ablation study of the index-code MI As mentioned in Section 4.3.2, the latent regularization of TCWAE and β-TCVAE only differ by the presence or absence of index-code MI. Here, we show that it has minimal impact on the reconstruction-disentanglement trade off. We modify our objective Equation (4.8) by explicitly adding an index-code MI term. The resulting objective is now an upper-bound of the (TC)WAE objective and is equivalent to a pseudo β-TCVAE where the reconstruction cost would be replaced by the square euclidean distance. We use $\alpha = \gamma$ and we compare the reconstruction-disentangle trade off of the modified TCWAE, denoted by TCWAE MI, with the original TCWAE in Figure 4.6.

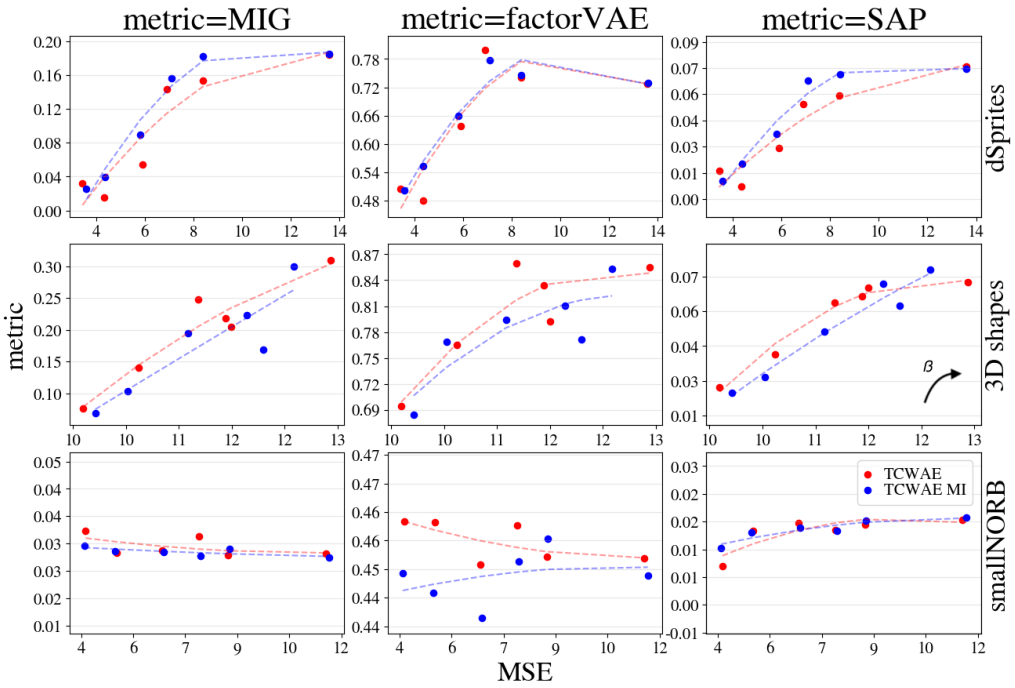


Figure 4.6: Same than Figure 4.5 for TCWAE-MWS with (blue) and without code-index MI (red).

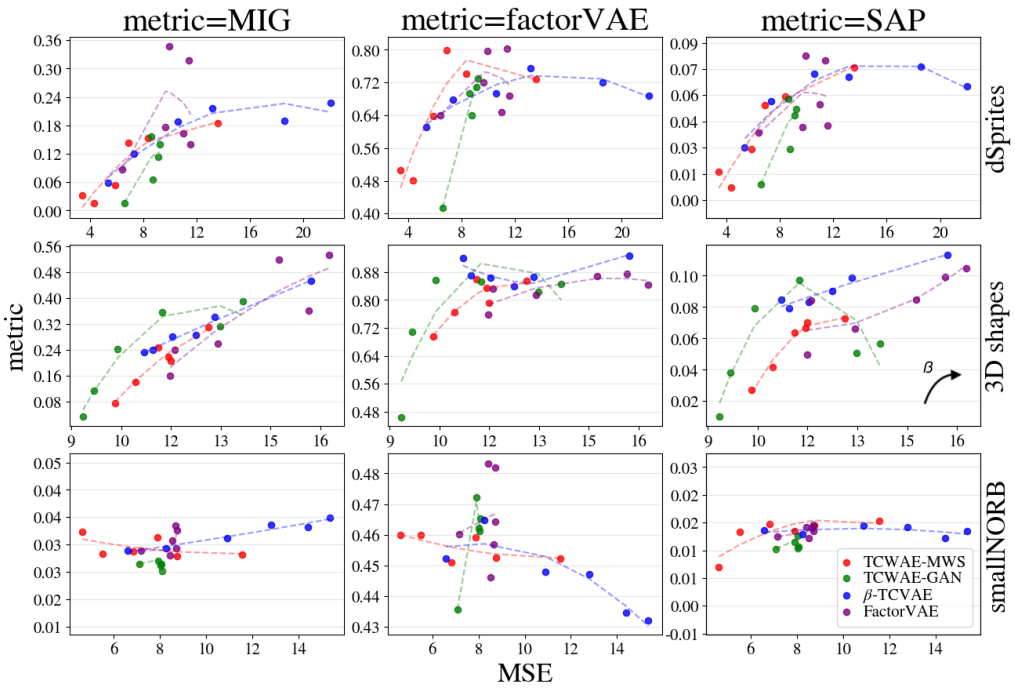


Figure 4.7: Same than Figure 4.5 for TCWAE-MWS (red), TCWAE-GAN (green), β-TCVAE (blue) and FactorVAE (purple).

All other things being equal, we observe that adding an index-code MI term has little to no impact on the reconstruction (vertical alignment of the points for the

same β). The presence of index-code MI does not result in significant performance difference across all datasets, conforms to the observation of Chen et al. [2018]. This suggests that TCWAE is a more natural objective for disentanglement learning removing the inconsequential index-code MI term, and thus the need for a third hyper parameter to tune (choice of α in Equation (4.9)).

Disentanglement performance We benchmark our methods against β-TCVAE [Chen et al., 2018] and FactorVAE [Kim and Mnih, 2018] on the three different datasets. As Chen et al. [2018] we take $\alpha = 1$ in β-TCVAE whilst for γ, we follow the same method than the one used before for TCWAE (see Appendix B.2 and Table B.5). As previously, we visualise the reconstruction-disentanglement trade off by plotting the different disentanglement scores against the MSE in Figure 4.7. As expected, the TC term controls the reconstruction-disentanglement trade off as more TC penalization leads to better disentanglement scores but higher reconstruction cost. However, similarly to their VAE counterparts, too much penalization on the TC deteriorates the disentanglement as the poor quality of the reconstructions prevents any disentanglement in the generative factors.

With the range of parameters chosen, both the TCWAE-GAN and FactorVAE reconstructions seem to be less sensitive to the TC regularization as shown by the more concentrated MSE scores. TCWAE models also exhibit smaller MSE than VAEs methods for almost all the range of selected parameters. Thus, whilst it remains difficult to assert the dominance of one method, both because the performance varies from one dataset to another and from one metric to another within each dataset[1], we argue that TCWAE improves the reconstruction performance (smaller MSE) whilst retaining competitive disentanglement performance (see top row of Figures 4.2, 4.3 and 4.4). In Table 4.1, we report for each method the best β taken to be the one achieving an overall best ranking over the disentanglement scores (see Appendix B.1 for more details). TCWAEs achieve competitive disentanglement performance when compared to the benchmark VAE-based methods.

[1]See Locatello et al. [2019] for the importance of inductive biases in disentanglement learning.

Table 4.1: Reconstruction and disentanglement scores (± one standard deviation). Best scores are in bold whilst second best are underlined.

Method	**MSE**	**MIG**	**factorVAE**	**SAP**
TCWAE MWS	13.6 ± 1.2	$0.18 \pm .09$	$\underline{0.73 \pm .08}$	$\underline{0.076 \pm .004}$
TCWAE GAN	$\mathbf{8.6 \pm 1.1}$	$0.16 \pm .06$	$0.69 \pm .10$	$0.058 \pm .022$
β-TCVAE	18.57 ± 0.6	$\underline{0.19 \pm .06}$	$0.72 \pm .09$	$\underline{0.076 \pm .006}$
FactorVAE	$\underline{9.9 \pm 0.5}$	$\mathbf{0.34 \pm .06}$	$\mathbf{0.80 \pm .05}$	$\mathbf{0.083 \pm .023}$

(a) dSprites

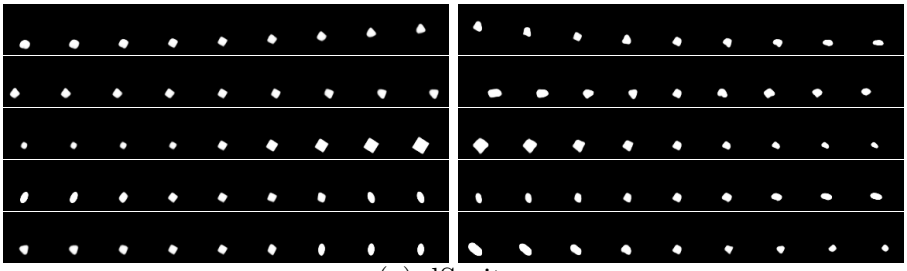
Method	**MSE**	**MIG**	**factorVAE**	**SAP**
TCWAE MWS	$\underline{13.1 \pm 0.9}$	$0.31 \pm .12$	$\underline{0.85 \pm .07}$	$0.073 \pm .014$
TCWAE GAN	$\mathbf{11.8 \pm 0.8}$	$0.36 \pm .26$	$\underline{0.85 \pm .12}$	$0.097 \pm .065$
β-TCVAE	16.2 ± 0.9	$\underline{0.45 \pm .16}$	$\mathbf{0.93 \pm .07}$	$\mathbf{0.113 \pm .015}$
FactorVAE	16.8 ± 3.0	$\mathbf{0.53 \pm .17}$	$0.84 \pm .03$	$\underline{0.105 \pm .014}$

(b) 3D shapes

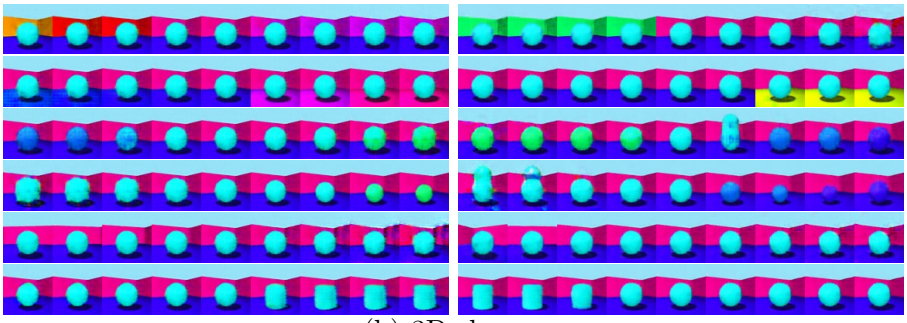
Method	**MSE**	**MIG**	**factorVAE**	**SAP**
TCWAE MWS	$\mathbf{4.6 \pm 0.0}$	$0.032 \pm .00$	$0.46 \pm .02$	$0.008 \pm .002$
TCWAE GAN	$\underline{7.9 \pm 0.2}$	$0.027 \pm .00$	$\underline{0.47 \pm .01}$	$\underline{0.014 \pm .003}$
β-TCVAE	12.80 ± 0.1	$\mathbf{0.034 \pm .00}$	$0.45 \pm .01$	$\mathbf{0.017 \pm .002}$
FactorVAE	8.7 ± 0.1	$\underline{0.033 \pm .00}$	$\mathbf{0.48 \pm .02}$	$\mathbf{0.017 \pm .002}$

(c) smallNORB

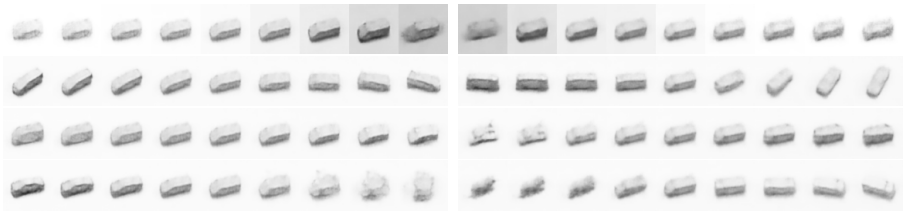
Latent traversals For each dataset, we plot the latent traversals to visually assess the learned representation in Figure 4.8. More specifically, we encode one observation and traverse the latent dimensions one at the time (rows) and reconstruct the resulting latent traversals (columns). Only the active latent dimensions are shown. TCWAEs manage to capture and disentangle most of the generative factors in every dataset. The exception being the discrete generative factor representing the shape category in dSprites and 3D shapes and the instance category in smallNORB. This probably comes from the fact that the models try to learn discrete factors with continuous Gaussian variables. Possible fixes would be to use more structured priors and posteriors with hierarchical latent variables such as in Esmaeili et al. [2018], but is out-of-scope for this work.



(a) dSprites

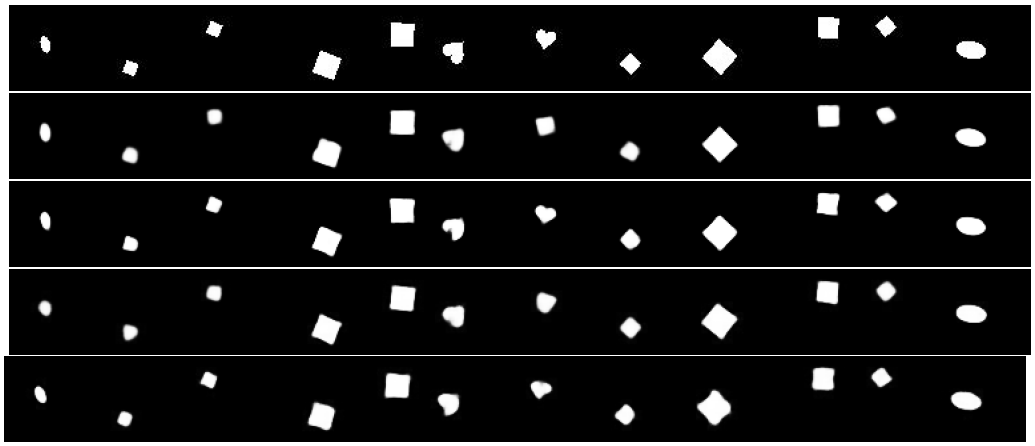


(b) 3D shapes

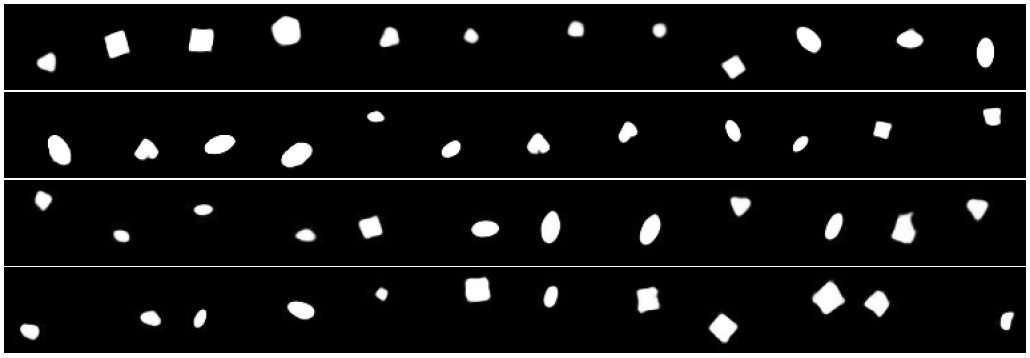


(c) smallNORB

Figure 4.8: Active latent traversals for each model. The parameters are the same as the ones used in Table 4.1. Subplots on the left column show traversals for TCWAE-MWS whilst subplots on the right show traversals TCWAE-GAN. Traversal range is $[-2, 2]$.



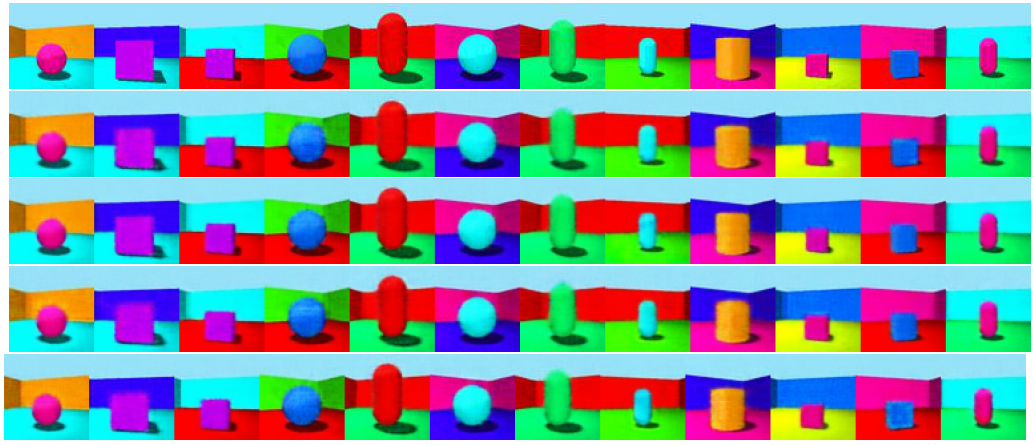
(a) Reconstructions



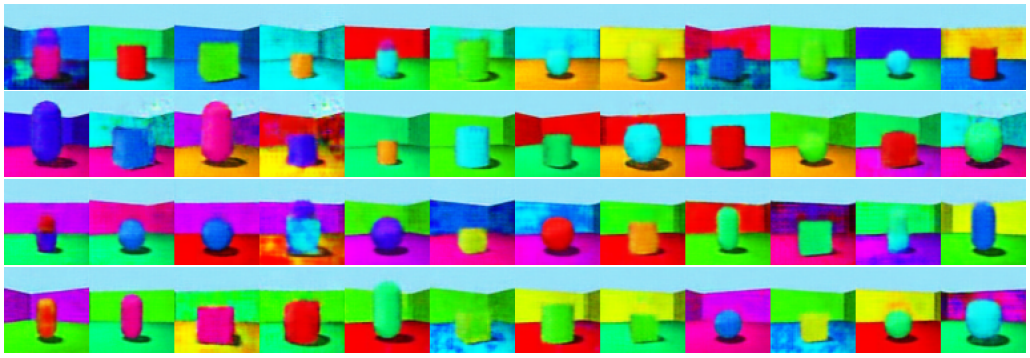
(b) Samples

Figure 4.9: Reconstructions and samples for the different methods on dSprites. Top reconstruction row corresponds to true data points whilst the second-from-top to bottom reconstruction rows correspond in order to the reconstructions of the TCWAE-MWS, TCWAE-GAN, β-TCVAE and FactorVAE. Samples rows correspond from top to bottom to TCWAE-MWS, TCWAE-GAN, β-TCVAE and FactorVAE

To qualitatively assess the generative performances, we plot the reconstructions and samples of the different methods for the different datasets in Figures 4.9, 4.10 and 4.11. Whilst all methods show similar reconstruction performance, the visual inspection of the samples reinforces the justification for a different reconstruction cost function. Indeed, TCWAEs exhibit qualitatively better generative performance with crispier and more natural looking samples.

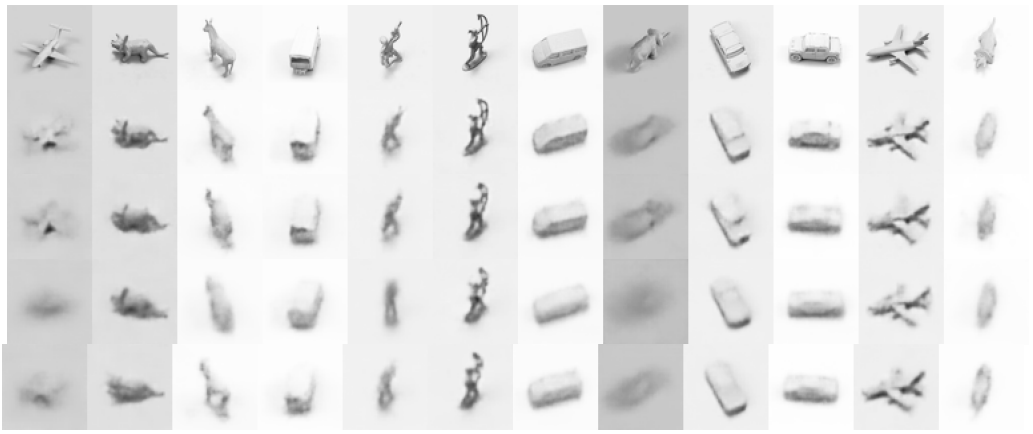


(a) Reconstructions

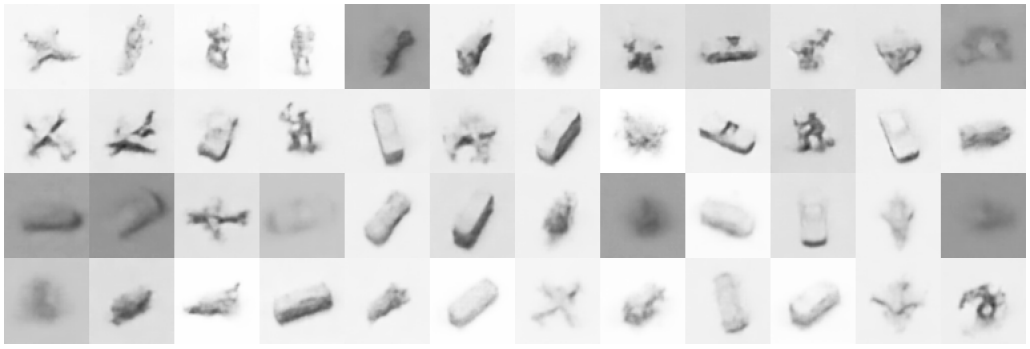


(b) Samples

Figure 4.10: Same as Figure 4.9 for 3D shapes.

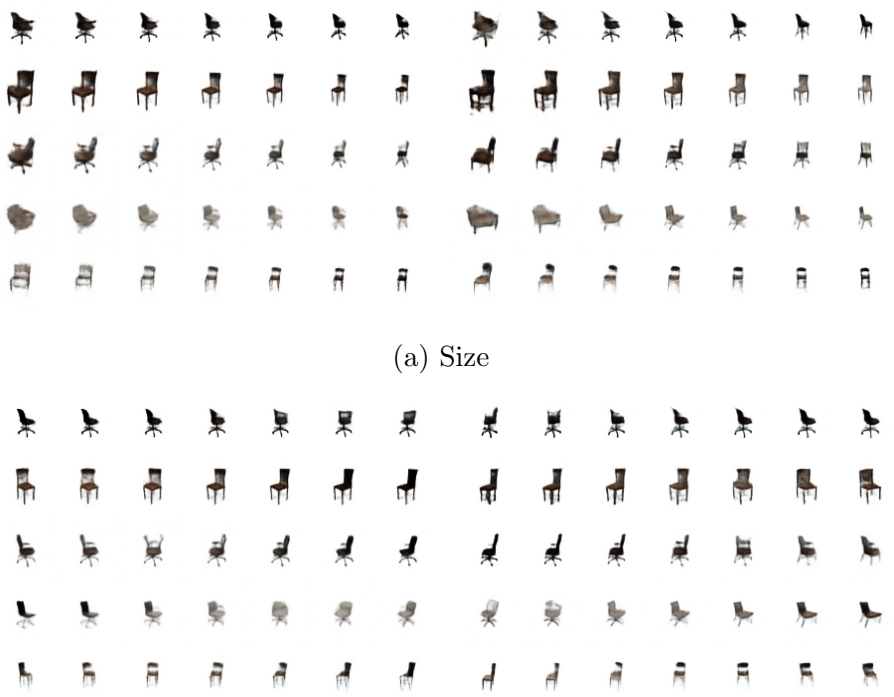


(a) Reconstructions



(b) Samples

Figure 4.11: Same as Figure 4.9 for smallNORB.



(a) Size

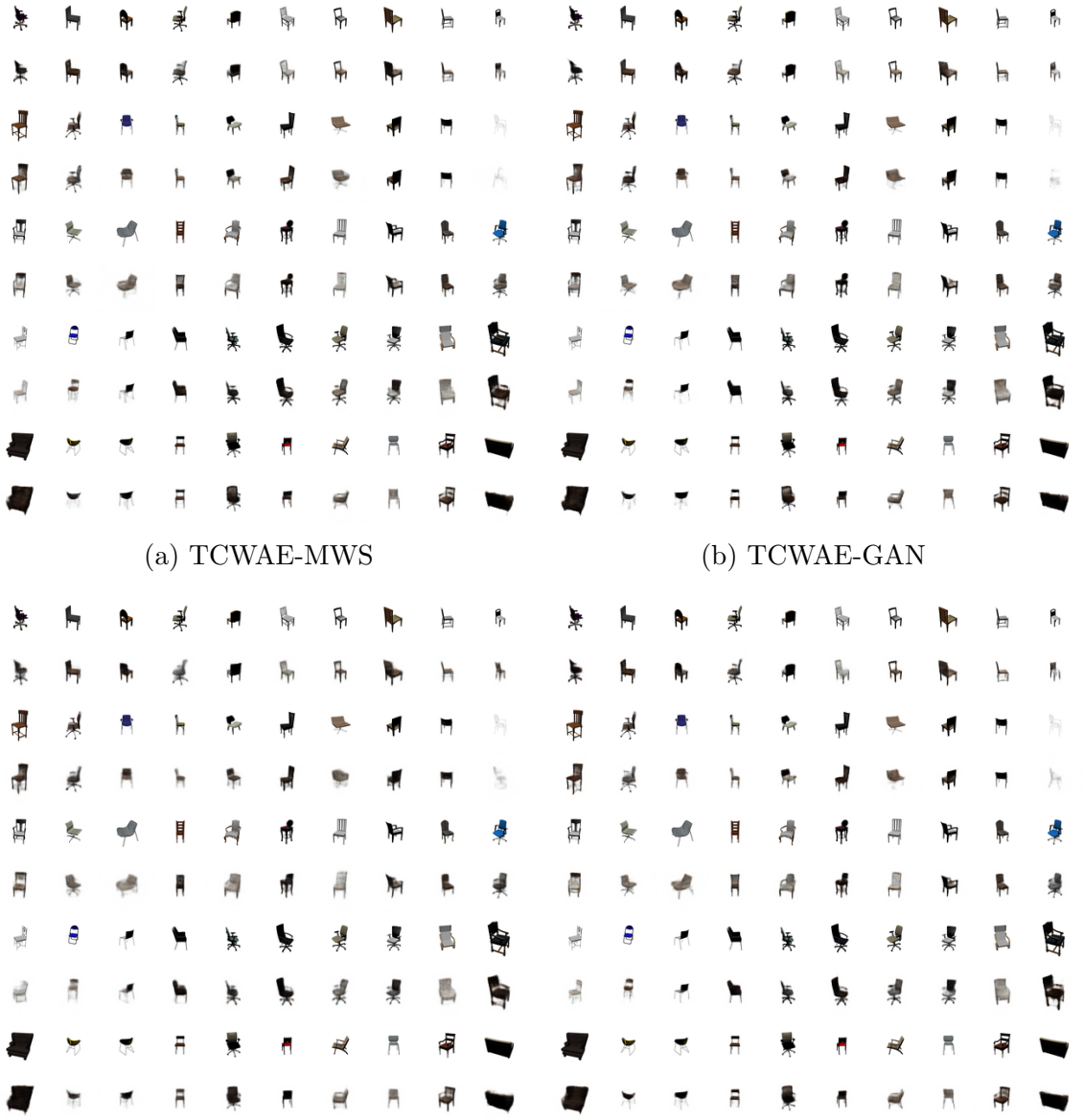
(b) Orientation

Figure 4.12: Latent traversals for TCWAE-MWS (left column) and TCWAE-GAN (right column) on 3D chairs. Within each subplot, each line corresponds to one input data point. We vary evenly the encoded latent codes in the interval $[-2, 2]$.

4.4.2 Qualitative analysis: disentanglement on real-world datasets

We train our methods on 3D chairs [Aubry et al., 2014] and CelebA [Liu et al., 2015] whose generative factors are not known and qualitatively find that TCWAEs achieve good disentanglement. Reconstructions and samples are given Figures 4.13 and 4.14 whilst Figure 4.12 shows the latent traversals of two different factors learned by the TCWAEs on 3D chairs (see Appendix B.3 for additional generative factors). Similarly for CelebA, we plot two different generative factors found by TCWAEs in Figure 4.15 (additional factors are given in Appendix B.3) and the reconstructions and model samples are given Figures B.2 and B.3 in Appendix B.3.

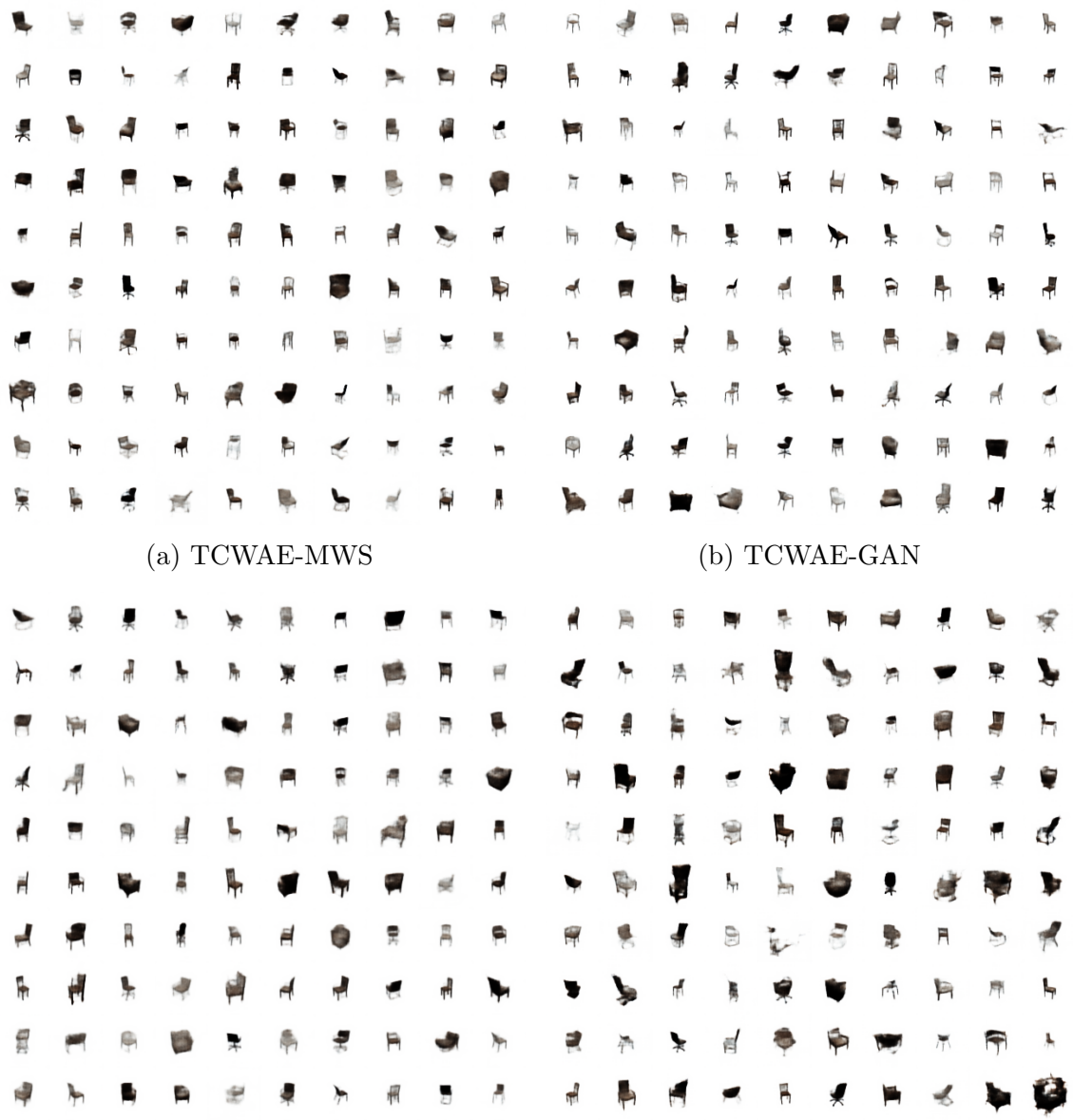
Visually, TCWAEs manage to capture different generative factors whilst retaining good samples. This confirms our intuition that the flexibility offered in the reconstruction term, mainly the possibility to chose the reconstruction cost and use deterministic decoders, improves the reconstruction-disentanglement trade off.



(a) TCWAE-MWS (b) TCWAE-GAN

(c) β-TCVAE (d) FactorVAE

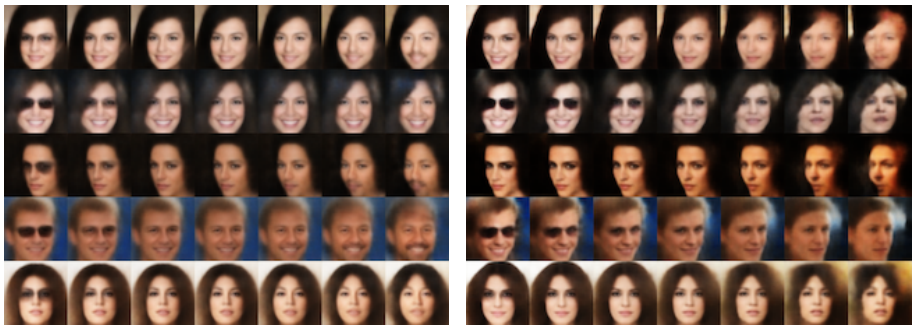
Figure 4.13: Reconstructions for 3D chairs. Within pairs of rows, the observation is above with the corresponding reconstruction below.



(a) TCWAE-MWS (b) TCWAE-GAN

(c) β-TCVAE (d) FactorVAE

Figure 4.14: Model samples for 3D chairs.



(a) Glasses/Beard

(b) Orientation

Figure 4.15: Same than Figure 4.12 for CelebA with latent traversal range $[-4, 4]$.

In order to further assess the quality of the reconstructions and samples, we compute the MSE of the reconstructions and the FID scores [Heusel et al., 2017] of the reconstructions and samples. Results are reported in Table 4.2. Whilst the fact that TCWAEs indeed presents better MSE is not a surprise (the reconstruction cost actually minimises the MSE in TCWAEs), TCWAEs also present better reconstruction FID scores. More interesting from a generative modeling point of view, TCWAEs also present better samples than their VAE-counterparts when looking at the samples FID scores. This is visually confirmed when inspecting the model samples given in Figures 4.14 and B.3. These results show the benefit of using a different ground cost function, allowing better reconstruction and generative performance whilst retaining good disentanglement performance.

Table 4.2: MSE and FID scores for 3D chairs and CelebA (± one standard deviation). Best scores are in bold whilst second best are underlined.

Method	MSE	Rec. FID	Samples FID
TCWAE-MWS	38.2 ± 0.1	$\mathbf{61.3 \pm 0.4}$	$\mathbf{72.6 \pm 0.5}$
TCWAE-GAN	$\mathbf{36.6 \pm 0.3}$	$\underline{61.6 \pm 1.4}$	$\underline{74.6 \pm 1.7}$
β-TCVAE	54.0 ± 0.5	74.9 ± 1.7	85.1 ± 0.1
FactorVAE	$\underline{36.7 \pm 0.7}$	62.8 ± 1.0	98.9 ± 6.6

(a) 3D chairs

Method	MSE	Rec. FID	Samples FID
TCWAE-MWS	$\mathbf{150.5 \pm 0.7}$	75.2 ± 0.2	$\underline{82.8 \pm 0.0}$
TCWAE-GAN	186.4 ± 18.9	$\mathbf{72.4 \pm 3.5}$	$\mathbf{73.3 \pm 2.6}$
β-TCVAE	$\underline{182.4 \pm 0.3}$	84.5 ± 0.2	91.5 ± 0.7
FactorVAE	237.6 ± 10.7	$\underline{74.0 \pm 1.4}$	83.3 ± 4.5

(b) CelebA

4.5 Conclusion

Inspired by the surgery of the KL regularization term of the ELBO objective, we developed a new disentanglement method based on the WAE objective. We chose the KL divergence between the aggregated posterior and the prior as our latent divergence function. The WAE framework naturally enables the latent regularization to depend explicitly on the TC of the aggregated posterior, which is directly associated with disentanglement. Using two different estimators for the KL terms, we showed that our method achieves competitive disentanglement on toy datasets. Moreover, the flexibility in the choice of reconstruction cost function makes our method more compelling when working with more challenging datasets.

Conclusion

Generative modelling is a popular method to learn and approximate the often unknown process by which a set of data has been generated. Generative models recently gained in popularity due to their ability to learn from large amounts of unlabelled data while remaining interpretable. Especially, the recent Variational Autoencoders (VAEs) [Kingma and Welling, 2014, Rezende et al., 2014] have shown impressive performances in images and text generation, both in term of sample quality and likelihood score. However, their training remains challenging for deep latent models when combined with powerful encoder and decoder networks. These difficulties are inherent to the ill defined likelihood approach and are reflected in the inflexibility of the Evidence Lower Bound objective. This has motivated the emergence of likelihood-free alternatives, especially Optimal Transport (OT) based approaches. Amongst these, the recent Wasserstein Autoencoder (WAE) [Tolstikhin et al., 2018, Bousquet et al., 2017] echoes the encoder-decoder structure of VAEs but presents geometric properties that promise easier convergence of the model distributions when using gradient descent methods.

While discrete latent models have high potential for many real world applications given the omnipresence of discrete data, their use has been limited by the difficulties encountered when training them. Especially, VAEs are known to fail at fully exploiting the discrete structure of the discrete latent model without signal supervision. Work-around solutions range from tailored encoder and decoder networks [Jang et al., 2017, Maddison et al., 2017, Van den Oord et al., 2017, Johnson et al., 2016] and training procedures [Eslami et al., 2016, Lawson et al., 2018] to adding back some supervision through a subset of labelled discrete data [Kingma et al., 2014]. In Chapter 2, we investigated why vanilla VAEs are not suited for the discrete setting. We found that

VAEs quickly learn to ignore the discrete structure of the model, resulting in the collapse of the latent modes. On the contrary, we showed that WAE was successful at fully leveraging the discrete-continuous latent structure of Gaussian mixture latent variable models. WAEs achieved promising generative results on MNIST whilst displaying a more interpretable latent representation for structured models with both discrete and continuous latent variables.

In Chapter 3, we introduced StWAE, a new training objective especially designed for deep latent hierarchical generative models. Deep latent hierarchies have the potential to increase the model expressiveness as well as offering a more explainable latent representation. Yet VAEs often struggle to fully leverage the full depth of the latent hierarchy. Successful VAE-based methods rely on highly tailored inference and generative network designs [Sønderby et al., 2016, Bachman, 2016, Kingma et al., 2016, Vahdat and Kautz, 2020] as well as complex training schemes [Kingma et al., 2016, Maaløe et al., 2019, Yoshida and Miyato, 2017, Vahdat and Kautz, 2020]. Our approach recursively applies the Wasserstein distance as a latent regularisation divergence, creating a stack of WAEs spanning the full latent hierarchy. We performed quantitative comparison with other VAE-based methods when training first order Markovian latent models and analysed the representation captured through the latent hierarchy. We showed that when trained with StWAE, the simple first order Markovian structure reduces the computational cost of the model whilst providing a qualitatively more interpretable latent representation.

In addition to offering a simpler training for learning deep latent models, we showed in Chapter 4 that the WAE framework also lands itself to learning interpretable representation by introducing the Total Correlation Wasserstein Autoencoder (TCWAE). Model interpretability is known to improve the performance and robustness of the model [Bengio et al., 2013, van Steenkiste et al., 2019], making the learning of interpretable latent representation important for many downstream tasks. Specifically, a latent representation with statistically independent latent variables, each encoding for a unique and semantically meaningful underlying generative factors Bengio et al. [2013], provides a better control over how the model captures information from the data. TCWAE leverages the decomposition of the KL divergence to

make the dependency on the total correlation of the aggregated posterior explicit, improving the disentanglement of the latent representation. We proposed to use two different estimators for the intractable KL terms and showed that our methods achieved competitive disentanglement performance on benchmark toy datasets. Finally, leveraging the flexibility of the reconstruction cost of the WAE, we found that TCWAE presents better generative performances when working with more challenging datasets.

The work presented here open the door to several research directions for the future:

Advances in generative modelling Arguably in this thesis, we focused on simple models, both in their factorisation into conditionally independent distributions and their parametrization with simple networks. Whilst model simplicity can be beneficial on its own as it allows for a better model interpretability, it can also limit the generative performance of the models and results. Significant advances in generative modelling have been made in recent years with more complex graphical models [Maaløe et al., 2019, Vahdat and Kautz, 2020], network architectures [Vaswani et al., 2017, Dosovitskiy et al., 2021], and training methods [Rezende and Mohamed, 2015, Sohl-Dickstein et al., 2015a, Song and Ermon, 2019, Ho et al., 2020]. How to adapt these recent advances to incorporate them in our works and apply them to our models, potentially bridging the gap with state-of-the-art approaches, is an open question.

Learning at scale In the past years, state-of-the-art models have become more powerful in part due to the availability of bigger datasets and more powerful computational resources, but also more efficient network parametrizations and training schemes. On the other hand, the work presented on this thesis have been trained mostly on small toy datasets with relatively simple networks. Can we realise the benefits of latent variable models at scale? How will we address the various challenges that training such large models is likely to raise (e.g. instability, KL collapse, etc.)?

Learning semantic representations While we performed quantitative assessments of the representation learned by our different models, we did not fully explore and evaluate their limits. Especially, we did not systematically compare and analyse our models on downstream tasks. However, how to assess and measure the quality of the learned representation is in itself a challenge. Indeed, representation are only meaningful for the specific tasks they are to be used for. Thus designing downstream tasks that can test and assess the performance of the learned representations is an important direction for future works.

Bibliography

A. Achille and S. Soatto. Information dropout: Learning optimal representations through noisy computation. In *Transactions on Pattern Analysis and Machine Intelligence*, 2018.

M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, 2017.

M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3D chairs: Exemplar part-based 2D-3D alignment using a large dataset of CAD models. In *Conference on Computer Vision and Pattern Recognition*, 2014.

P. Bachman. An architecture for deep, hierarchical generative models. In *Advances in Neural Information Processing Systems*, 2016.

P. Bachman, R. D. Hjelm, and W. Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, 2019.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. In *Transactions on Pattern Analysis and Machine Intelligence*, 2013.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization, Theory and Examples*. Springer, 2000.

O. Bousquet, S. Gelly, I. Tolstikhin, C. J. Simon-Gabriel, and B. Schölkopf. From optimal transport to generative modeling: The VEGAN cookbook. *arXiv:1705.07642*, 2017.

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Józefowicz, and S. Bengio. Generating sentences from a continuous space. *arXiv:1511.06349*, 2015.

A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. 2019.

S. Brooks, A. Gelman, G. Jones, and M. Xiao-Li. *Handbook of Markov chain Monte Carlo*. Chapman & Hall/CRC, 2011.

Y. Burda, G. R., and R. Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2015.

C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-VAE. *arXiv:804.03599*, 2018.

R. T. K. Chen, X. Li, R. Grosse, and D. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, 2018.

X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. Variational lossy autoencoder. *arXiv:1611.02731*, 2017.

K. Cho, B. Van Merri"enboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, 2013a.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, 2013b.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. In *Journal of the Royal Statistical Society*, 1977.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018.

P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, 2022.

N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *arXiv:1611.02648*, 2016.

L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation. *arXiv:1410.8516*, 2014.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2016a.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real nvp. In *International Conference on Learning Representations*, 2016b.

K. Do and T. Tran. Theory and evaluation metrics for learning disentangled representations. *arXiv:1908.09961*, 2019.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

G. K. Dziugaite, D. M. Roy, and Z. Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Conference on Uncertainty in Artificial Intelligence*, 2015.

C. Eastwood and C. K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, K. Kavukcuoglu, and G. E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems*, 2016.

B. Esmaeili, H. B. Wu, S. Jain, A. Bozkurt, N. Siddharth, B. Paige, D. H. Brooks, J. Dy, and J.-W. van de Meent. Structured disentangled representations. In *International Conference on Artificial Intelligence and Statistics*, 2018.

C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio. Learning with a Wasserstein loss. In *Advances in Neural Information Processing Systems*, 2015.

S. Gao, R. Brekelmans, G. Ver Steeg, and A. Galstyan. Auto-encoding total correlation explanation. In *International Conference on Artificial Intelligence and Statistics*, 2019.

B. Gaujac, I. Feige, and D. Barber. Gaussian mixture models with Wasserstein distance. In *Asian Conference on Machine Learning*, 2021a.

B. Gaujac, I. Feige, and D. Barber. Learning disentangled representations with the Wasserstein autoencoder. In *European Conference on Machine Learning*, 2021b.

B. Gaujac, I. Feige, and D. Barber. Learning deep-latent hierarchies by stacking Wasserstein autoencoders. *arXiv:2010.03467*, 2022.

A. Genevay, G. Peyre, and M. Cuturi. Learning generative models with Sinkhorn divergences. In *International Conference on Artificial Intelligence and Statistics*, 2018.

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2011.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2014.

A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *International Conference on Algorithmic Learning Theory*, 2005.

A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. In *Journal of Machine Learning Research*, 2012a.

A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B. K. Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *Advances in Neural Information Processing Systems*, 2012b.

I. Gulrajani, K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, and A. Courville. PixelVAE: A latent variable model for natural images. In *Advances in Neural Information Processing Systems*, 2016.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition*, 2016.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2016.

I. Higgins, D. Amos, D. Pfau, S. Racanière, L. Matthey, D. J. Rezende, and A. Lerchner. Towards a definition of disentangled representations. *arXiv:1812.02230*, 2018.

R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997.

M. D. Hoffman and M. J. Johnson. ELBO surgery: Yet another way to carve up the variational evidence lower bound. In *Advances in Neural Information Processing Systems*, 2016.

C. Huang, G.and Guo, M. Kusner, Y. Sun, Y. Sha, and K. Q. Weinberger. Supervised word mover's distance. In *Advances in Neural Information Processing Systems*, 2016.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015.

E. Jang, S. Gu, and B. Poole. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017.

M. J. Johnson, D. Duvenaud, A. B. Wiltschko, S. R. Datta, and R. P. Adams. Composing graphical models with neural networks for structured representations and fast inference. In *Advances in Neural Information Processing Systems*, 2016.

H. Kaiming, Z. Xiangyu, R. Shaoqing, and S. Jian. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2015.

L. Kantorovich. On the translocation of masses. In *Journal of Mathematical Sciences*, 2006.

H. Kim and A. Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, 2018.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, 2018.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, 2014.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.

A. Klushyn, N. Chen, R. Kurle, B. Cseke, and P. van ver Smagt. Learning hierarchical priors in VAEs. In *Advances in Neural Information Processing Systems*, 2019.

S. Kolouri, S. Park, M. Thorpe, D. Slepčev, and G. Rohde. Optimal mass transport: Signal processing and machine-learning applications. In *Signal Processing Magazine*, 2017.

L. G. Kraft. *A device for quantizing, grouping, and coding amplitude-modulated pulses*. Massachusetts Institute of Technology, 1949.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

A. Kumar, P. Sattigeri, and A. Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. In *International Conference on Learning Representations*, 2018.

A. B. L. Larsen, K. Sønderby S., H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *International Conference on Machine Learning*, 2016.

D. Lawson, G. Tucker, C.-C. Chiu, C. Raffel, K. Swersky, and N. Jaitly. Learning hard alignments with variational inference. In *International Conference on Acoustics, Speech and Signal Processing*, 2018.

Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010. `http://yann.lecun.com/exdb/mnist/`.

Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

Y. Li, K. Swersky, and R. Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, 2015.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision*, 2015.

F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, 2019.

L. Maaløe, M. Fraccaro, V. Liévin, and O. Winther. BIVA: A very deep hierarchy of latent variables for generative modeling. In *Advances in neural information processing systems*, 2019.

L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther. Auxiliary deep generative models. In *International Conference on Machine Learning*, 2016.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

A. Makhzani, J. Shlens, N. Jaitly, and I. Goodfellow. Adversarial autoencoders. In *International Conference on Learning Representations*, 2016.

L. Matthey, H. I., D. Hassabis, and A. Lerchner. dSprites: Disentanglement testing Sprites dataset. 2017. `https://github.com/deepmind/dsprites-dataset/`.

L. McInnes and J. Healy. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv:1802.03426*, 2018.

B. McMillan. Two inequalities implied by unique decipherability. In *Transactions on Information Theory*, 1956.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv:1312.5602*, 2013.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 2016.

G. Monge. *Memoire sur la theorie des deblais et des remblais.* 1781.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Advances in Neural Information Processing Systems Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

X. Nguyen, M. J. Wainwright, and I. J. Michael. Estimating divergence functionals and the likelihood ratio by penalized convex risk minimization. In *Advances in Neural Information Processing Systems*, 2008.

A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, 2022.

G. Ortega, J. Alberto, J. Rabin, B. Galerne, and T. Hurtut. Optimal patch assignment for statistically constrained texture synthesis. In *Scale Space and Variational Methods in Computer Vision*, 2017.

J. W. Paisley, D. M. Blei, and M. I. Jordan. Variational bayesian inference with stochastic search. In *International Conference on Machine Learning*, 2012.

G. Patrini, M. C., P. Forré, S. Bhargav, M. Welling, R. Van den Berg, T. Genewein, and F. Nielsen. Sinkhorn autoencoders. *arXiv:1810.01118*, 2018.

G. Peyré and M. Cuturi. Computational optimal transport: With applications to data science. In *Foundations and Trends in Machine Learning*, 2019.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2015.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016.

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. *OpenAI Technical Report*, 2018.

D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

A. Rolet, M. Cuturi, and G. Peyré. Fast dictionary learning with a smoothed Wasserstein loss. In *International Conference on Artificial Intelligence and Statistics*, 2016.

P. Rubenstein, O. Bousquet, J. Djolonga, C. Riquelme, and I. Tolstikhin. Practical and consistent estimation of f-divergences. In *Advances in Neural Information Processing Systems*, 2019.

P. K. Rubenstein, B. Schoelkopf, and I. Tolstikhin. Learning disentangled representations with Wasserstein autoencoders. In *International Conference on Learning Representations Workshop*, 2018a.

P. K. Rubenstein, B. Schölkopf, and I. Tolstikhin. On the latent space of Wasserstein auto-encoders. In *Workshop track - International Conference on Learning Representations*, 2018b.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, 2016.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. In *International conference on machine learning*, 2017.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Machine Learning*, 2015.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015a.

J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *International Conference on Machine Learning*, 2015b.

C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, 2016.

J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019.

M. Sugiyama, T. Suzuki, and T. Kanamori. Density ratio matching under the Bregman divergence: A unified framework of density ratio estimation. In *Annals of the Institute of Statistical Mathematics*, 2011.

M. Thorpe, S. Park, S. Kolouri, G. Rohde, and D. Slepčev. A transportation l^p distance for signal analysis. In *Journal of Mathematical Imaging and Vision*, 2017.

N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Annual Allerton Conference on Communication, Control and Computing*, 1999.

I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schölkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

J. M. Tomczak and W. Welling. VAE with VampPrior. In *International Conference on Artificial Intelligence and Statistics*, 2018.

M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations*, 2020.

A. Vahdat and J. Kautz. NVAE: A deep hierarchical variational autoencoder. In *Advances in Neural Information Processing Systems*, 2020.

A. Van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *ISCA Speech Synthesis Workshop*, 2016a.

A. Van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*, 2016b.

A. Van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with PixelCNN decoders. In *Advances in Neural Information Processing Systems*, 2016c.

A. Van den Oord, O. Vinyals, and K. kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.

A. Van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018.

S. van Steenkiste, F. Locatello, J. Schmidhuber, and O. Bachem. Are disentangled representations helpful for abstract visual reasoning? In *Advances in Neural Information Processing Systems*, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

G. Ver Steeg and A. Galstyan. Discovering structure in high-dimensional data through correlation explanation. In *Advances in Neural Information Processing Systems*, 2014.

C. Villani. *Optimal Transport: Old and New*. Springer, 2008.

W. Wang, J. Ozolek, D. Slepcev, A. Lee, C. Chen, and G. Rohde. An optimal transportation approach for nuclear structure-based pathology. In *Transactions on Medical Imaging*, 2011.

W. Wang, D. Slepcev, S. Basu, J. Ozolek, and G. Rohde. A linear optimal transportation framework for quantifying and visualizing variations in sets of images. In *International journal of computer vision*, 2013.

S. Watanabe. Information theoretical analysis of multivariate correlation. In *IBM Journal of Research and Development*, 1960.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

Y. Wu, Y. Burda, R. Salakhutdinov, and R. Grosse. On the quantitative analysis of decoder-based generative models. *arXiv:1611.04273*, 2016.

Y. Xiao and W. Y. Wang. Disentangled representation learning with Wasserstein total correlation. *arXiv:1912.12818*, 2019.

Y. Yoshida and T. Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv:1705.10941*, 2017.

S. Zha, J. Song, and S. Ermon. Towards deeper understanding of variational autoencoding models. *arxiv:1702.08658*, 2017.

M. Zhang, P. Hayes, T. Bird, R. Habib, and D. Barber. Spread divergence. In *International Conference on Machine Learning*.

S. Zhao, J. Song, and S. Ermon. Learning hierarchical features from deep generative models. In *International Conference on Machine Learning*, 2017.

S. Zhao, J. Song, and S. Ermon. InfoVAE: Balancing learning and inference in variational autoencoders. In *AAAI Conference on Artificial Intelligence*, 2019.

Appendix A

Foundations

A.0.1 Discrete Optimal Transport

In the discrete case, we consider discrete measure distributions on $\mathcal{X}$. For a set of point $\mathcal{X}_M = \{x_1, \cdots, x_M\} \subset \mathcal{X}$, the set of discrete measures, α, on $\mathcal{X}_M$, is defined by:

$$\mathcal{P}(\mathcal{X}_M) \stackrel{\text{def}}{=} \Big\{ \alpha = \sum_{m=1}^{M} a_m \delta_{x_m};\ a_m \in \mathbb{R}_+^\star : \sum_{m=1}^{M} a_m = 1 \Big\} \tag{A.1}$$

where δ_x is the Dirac centered at x. Often, when the support of the distributions is explicit and non ambiguous, the histogram description is used instead. A histogram a is defined on the probability simplex Σ_M:

$$\Sigma_M \stackrel{\text{def}}{=} \Big\{ a_m \in \mathbb{R}_+ : \sum_{m=1}^{M} a_m = 1 \Big\} \tag{A.2}$$

Note that in the definition of discrete measures, we consider only strictly positive histograms to avoid degeneracies where locations are associated with zero mass.

Given two sets of points $\mathcal{X}_M = \{x_1, \cdots, x_M\} \subset \mathcal{X}$ and $\mathcal{Y}_N = \{y_1, \cdots, y_N\} \subset \mathcal{Y}$, and a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$, we consider the cost matrix $C \in \mathbb{R}^{M \times N}$ given by $C_{m,n} \stackrel{\text{def}}{=} c(x_m, y_n)$ and the two probability measures $\alpha = \sum_{m=1}^{M} a_m \delta_{x_m}$ and $\beta = \sum_{n=1}^{N} b_n \delta_{y_n}$. The OT seeks for the optimal map from $\mathcal{X}_M$ to $\mathcal{Y}_N$ that *pushes* α toward β, resulting in the smallest transport cost as shown in Figure 1.1a. Formally,

we can introduce the set of discrete couplings between histograms a and b, denoted $U(a, b)$, as:

$$U(a,b) \stackrel{\text{def}}{=} \Big\{ P \in \mathbb{R}_+^{M \times N};\ P\mathbf{1}_N = a \quad \text{and} \quad P^\top \mathbf{1}_M = b \Big\} \tag{A.3}$$

Then, we can formulate the OT problem between α, β with the associated cost function c as:

$$\mathcal{L}_c\big(\alpha, \beta\big) \stackrel{\text{def}}{=} \min_{P \in U(a,b)} C \cdot P \tag{A.4}$$

A.0.2 Topology of the Wasserstein distance

A topological vector space is a vector space whose operations on its elements are continuous. Normed vector spaces are such topological vector spaces as the norm of a vector space induces a metric on this space and this metric induces itself a topological structure. This topology characterizes the convergence of sequences of functions. Especially, we can define a notion of strong and weak topology on topological vector spaces. Given two distances δ_1, δ_2 on $\mathcal{P}(\mathcal{X})$, the topology induced by δ_1 is weaker than the one induced by δ_2 if the set of convergent sequences under δ_2 is included in the set of convergent sequences under δ_1:

$$\begin{aligned} \delta_1 \text{ weaker than } \delta_2 \ \Leftrightarrow \big\{ (P_m)_m \in \mathcal{P}(\mathcal{X}),\ \delta_2(P_m, P_\infty) \underset{m\to\infty}{\to} 0 \big\} \\ \subset \big\{ (P_m)_m \in \mathcal{P}(\mathcal{X}),\ \delta_1(P_m, P_\infty) \underset{m\to\infty}{\to} 0 \big\} \end{aligned} \tag{A.5}$$

In our case, we consider the space of distribution probability on $\mathcal{X}$, $\mathcal{P}(\mathcal{X})$. As a dual space, it has a strong and weak topology. The strong topology is given by the Total Variation (TV) norm:

$$\forall P \in \mathcal{P}(\mathcal{X}), \quad \|P\|_{TV} \stackrel{\text{def}}{=} \sup_{A \in \mathcal{X}} |P(A)| \tag{A.6}$$

and its associated distance δ_{TV}:

$$\forall (P, Q) \in \mathcal{P}(\mathcal{X})^2, \quad \delta_{TV}(P, Q) \stackrel{\text{def}}{=} \|P - Q\|_{TV} \tag{A.7}$$

whilst the Wasserstein distance induces the weak topology [Villani, 2008]. More precisely, the relative *strength* of the topologies induced respectively by the KL divergence[1], the TV norm and the Wasserstein distance are characterized as follow [Arjovsky et al., 2017]:

$$\mathbf{KL}\big(P_m \parallel P_\infty\big) \underset{m\to\infty}{\to} 0 \;\Rightarrow\; \delta_{TV}(P_m, P_\infty) \underset{m\to\infty}{\to} 0 \;\Rightarrow\; \mathcal{W}_{c,l}\big(P_m, P_\infty\big) \underset{m\to\infty}{\to} 0 \quad \text{(A.8)}$$

Note that the first inequality in Equation (A.8) also holds for the *other* KL: $\mathbf{KL}\big(P_\infty \parallel P_m\big)$[2]. Moreover, the convergence under the Wasserstein distance, or the *weak* convergence, is equivalent to the convergence in distributions for random variables [Arjovsky et al., 2017].

[1]The KL divergence not being a distance, it actually does not induce a topology. However, the Jensen-Shannon satisfies the distance axioms defined in Equation (1.18) and induces the same topology than the one induced by the TV norm.

[2]Recall that due to the KL asymmetry, $\mathbf{KL}\big(P_m \parallel P_\infty\big) \neq \mathbf{KL}\big(P_\infty \parallel P_m\big)$.

Appendix B

TCWAE

B.1 Implementation details

Data sets We train and compare our methods on five different data sets, three with known ground-truth generative factors (see Table B.1): dSprites Matthey et al. [2017] with 737,280 binary, 64×64 images, 3D shapes Kim and Mnih [2018] with 480,000 RGB, 64×64 images and smallNORB LeCun et al. [2004] with 48,600 greyscale, 64×64 images; and two with unknown ground-truth generative factors: 3D chairs Aubry et al. [2014] with 86,366 RGB, 64×64 images and CelebA Liu et al. [2015] with 202,599 RGB 64×64 images.

Table B.1: Ground-truth generative-factors.

Data set	**Generative factors** (# of values)
dSprites	Shape (3), Orientation (40), Position X (32), Position Y (32)
3D shapes	Floor hue (10), Wall hue (10), Object hue (10), Scale (8), Shape (4), Orientation (15)
smallNORB	Categories (5), Lightings (6), Elevations (9), Azimuths (18)

Models and optimization setup In Section 4.4.1, we take the latent dimension $d_{\mathcal{Z}} = 10$ for all the models, while we use $d_{\mathcal{Z}} = 16$ for 3D chairs and $d_{\mathcal{Z}} = 32$ for CelebA in Section 4.4.2. We use Gaussian encoders with diagonal covariance matrix

in all the models, deterministic decoder networks for TCWAEs and Gaussian decoder with diagonal covariance matrix for the VAE-based methods. We follow Locatello et al. [2019] for the architectures in all the experiments expect for CelebA where we follow Tolstikhin et al. [2018] (details of the networks architectures are given below). For all experiments, We use a batch size of 64 with the Adam optimizer Kingma and Ba [2015] with a learning rate of 0.0001, beta1 of 0.9, beta2 of 0.999 and epsilon of 0.0008. We select the γ in Section 4.4.1 by training a first batch of TCWAEs for 300,000 iterations. Once we chose and fixed γ, we train the TCWAEs with the selected γ, β-TCVAE and FactorVAE for the different β over 600,000 iterations. Note that for the index-code MI ablation study, we choose the same γ for both the TCWAE and TCWAE MI. for all the experiments, we show the results averaged over 5 training runs. In Section 4.4.2, we train the all models over 600,000 iterations.

Metrics and scores We follow Locatello et al. [2019] for the implementation of the disentanglement metrics to the difference that we use 5000 points for the MIG and 5000 training and 1000 testing points for the FactorVAE and SAP scores. We follow Heusel et al. [2017] for the FID implementation in Section 4.4.1: we first compute the activation statistics of the features maps on the full test set for both the reconstruction, respectively samples, and the true observations. We then compute the Frechet distance between two Gaussian with the computed statistics. We use a validation run to select the parameters values and report the MSE and FID scores on a test run.

We select γ in Section 4.4.1 such that, when averaged over all models with this γ, it achieves the best overall score, s, where the score is defined as the sum of the ranking on each individual metric: $s = r_{MSE} + \sum_{metric} r_{metric}$ where r_{MSE} designed the ranking of the MSE (lower is better) and r_{metric}, for $metric$ in {MIG, FactorVAE, SAP}, is the ranking of the disentanglement performance as measured by the given metric (higher is better). β is then chosen such that it achieves the best overall disentanglement score, s_{dis}: $s_{dis} = \sum_{metric} r_{metric}$ where r_{metric} is defined above.

Networks architectures The Gaussian encoder networks, $q_\phi(z|x)$ and decoder network, $p_\theta(x|z)$, are parametrized by neural networks as follow:

$$p_\theta(x|z) = \begin{cases} \delta_{\boldsymbol{f_\theta(z)}} & \text{if WAE based method,} \\ \mathcal{N}\big(\boldsymbol{\mu_\theta}(z), \boldsymbol{\sigma^2_\theta}(z)\big) & \text{otherwise.} \end{cases}$$
$$q_\phi(z|x) = \mathcal{N}\big(\boldsymbol{\mu_\phi}(x), \boldsymbol{\sigma^2_\phi}(x)\big)$$

where $\boldsymbol{f_\theta}$, $\boldsymbol{\mu_\theta}$, $\boldsymbol{\sigma^2_\theta}$, $\boldsymbol{\mu_\phi}$ and $\boldsymbol{\sigma^2_\phi}$ are the outputs of convolutional neural networks. All the experiments use the architectures of Locatello et al. [2019] except for CelebA where we use the architecture inspired by Tolstikhin et al. [2018]. The details for the architectures are given Tables B.2.

All the discriminator networks, D, are fully connected networks and share the same architecture with the optimisation setup given Table B.3.

B.2 Quantitative experiments

Hyper parameter tuning

The ranges of parameters for each experiment are given Table B.4. For example, in Section 4.4.1, we train the TCWAEs with the square euclidean distance for $(\beta, \gamma) \in \{0.1, 0.25, 0.5, 0.75, 1, 2\} \times \{0.1, 0.25, 0.5, 0.75, 1, 2\}$.

Following the method described in Section B.1, the chosen γ are given in Table B.5.

Table B.2: Networks architectures

Encoder	**Decoder**
Input: $64 \times 64 \times$ c	Input: $d_{\mathcal{Z}}$
CONV_2 $4 \times 4 \times 32$, ReLU	FC 256, ReLU
CONV_2 $4 \times 4 \times 32$, ReLU	FC $4 \times 4 \times 64$, ReLU
CONV_2 $4 \times 4 \times 64$, ReLU	CONV_2 $4 \times 4 \times 64$, ReLU
CONV_2 $4 \times 4 \times 64$, ReLU	CONV_2 $4 \times 4 \times 32$, ReLU
FC 256, ReLU	CONV_2 $4 \times 4 \times 32$, ReLU
FC $2 \times d_{\mathcal{Z}}$	CONV_2 $4 \times 4 \times c$

(a) Locatello et al. [2019] architectures

Encoder	**Decoder**
Input: $64 \times 64 \times c$	Input: $d_{\mathcal{Z}}$
CONV_2 $4 \times 4 \times 32$, BN+ReLU	FC $8 \times 8 \times 256$, BN+ReLU
CONV_2 $4 \times 4 \times 64$, BN+ReLU	CONV_2 $4 \times 4 \times 128$, BN+ReLU
CONV_2 $4 \times 4 \times 128$, BN+ReLU	CONV_2 $4 \times 4 \times 64$, BN+ReLU
CONV_2 $4 \times 4 \times 256$, BN+ReLU	CONV_2 $4 \times 4 \times 32$, BN+RelU
FC $2 \times d_{\mathcal{Z}}$	CONV_2 $4 \times 4 \times c$

(b) CelebA networks architectures

Table B.3: Discriminator setup

Parameter	**Value**	**Architecture**
lr Sec. 4.4.1	$1e^{-4}$	Input: $d_{\mathcal{Z}}$
lr Sec. 4.4.2	$1e^{-5}$	FC 1000, ReLU
beta 1	0.5	FC 1000, ReLU
beta 2	0.9	FC 1000, ReLU
epsilon	1e-08	FC 1000, ReLU
		FC 1000, ReLU
		FC 1000, ReLU
		FC 2

Table B.4: β values for the different cost functions on each data set.

Method	**Section 4.4.1**	**Section 4.4.2**
	$\lVert\cdot\rVert_2^2$	
TCWAE-MWS	$\{0.1, 0.25, 0.5, 0.75, 1, 2\}$	$\{1, 2, 5, 10, 15, 20\}$
TCWAE-GAN	$\{0.5, 1, 2.5, 5, 7.5, 10\}$[a]	$\{1, 2, 5, 10, 20, 50\}$
	Cross-Entropy	
TCWAE-MWS	$\{1, 2, 4, 6, 8, 10\}$	n.a.
TCWAE-GAN	$\{1, 10, 25, 50, 75, 100\}$	n.a.
β-TCVAE	$\{1, 2, 4, 6, 8, 10\}$	$\{1, 2, 5, 10, 15, 20\}$
FactorVAE	$\{1, 10, 25, 50, 75, 100\}$	$\{1, 2, 5, 10, 20, 50\}$

[a] We found that $\gamma = 0.1$ resulted in modes collapsing with dSprites and smallNORB, thus opted to start with $\gamma = 0.5$ instead for these two data sets.

Table B.5: γ values for the different methods on each data set.

Method	**dSprites**	**3D shapes**	**smallNORB**
		$\|\|2\|\|_.^2$	
TCWAE MWS	0.25	0.5	0.1
TCWAE GAN	1	0.1	1
		Cross-entropy	
TCWAE MWS	1	4	1
TCWAE GAN	1	100	1
β-TCVAE	6	10	4
FactorVAE	75	50	10

B.3 Qualitative experiments

We plot additional factors found by the TCWAEs in Figures B.1 and B.4 for respectively 3Dchairs and CelebA.

Reconstructions and samples for the different models on CelebA chairs are given Figures B.2 and B.3.

(a) Legs type

(b) Backrest size

Figure B.1: Latent traversals for TCWAE-MWS (left column) and TCWAE-GAN (right column) on 3D chairs. Each line corresponds to one input data point while each subplot corresponds to one latent factor. We vary evenly the encoded latent codes in the interval $[-2, 2]$.

(a) TCWAE-MWS (b) TCWAE-GAN

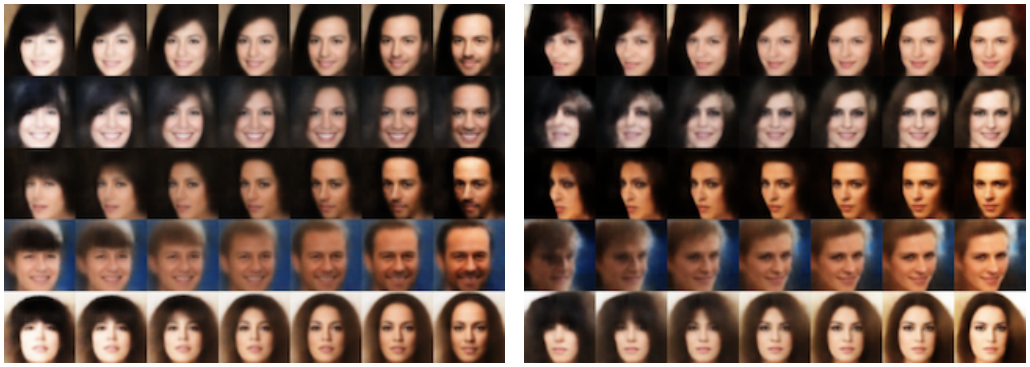
(c) β-TCVAE (d) FactorVAE

Figure B.2: Same as Figure 4.13 for CelebA.

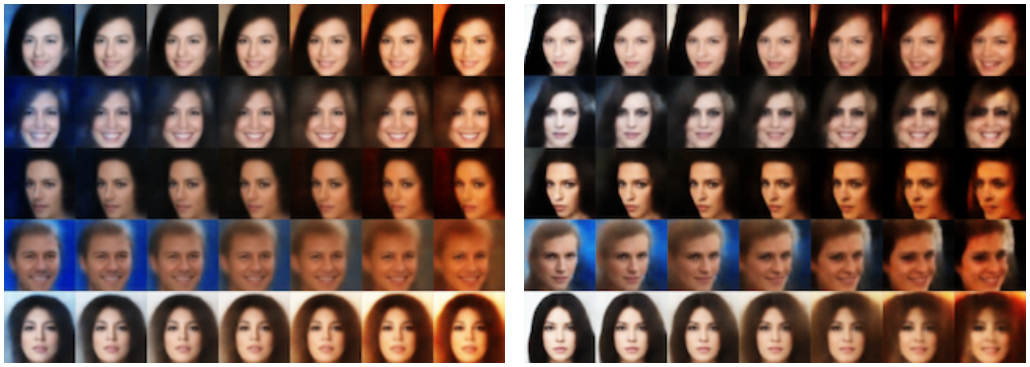
(a) TCWAE-MWS (b) TCWAE-GAN

(c) β-TCVAE (d) FactorVAE

Figure B.3: Same as Figure 4.14 for CelebA.



(a) Baldness



(b) Hue

Figure B.4: Latent traversals for TCWAE-MWS (left column) and TCWAE-GAN (right column) on CelebA. Each line corresponds to one input data point while each subplot corresponds to one latent factor. We vary evenly the encoded latent codes in the interval $[-4, 4]$.