THE POLARISED REGIME OF VARIATIONAL AUTOENCODERS

LISA BONHEME

A thesis submitted for the degree of
Doctor of Philosophy

School of Computing
University of Kent

September 14, 2023

ABSTRACT

Variational Autoencoders (VAEs) learn in a polarised regime, a sparsity-enforcing regime where a subset of latent representations (the active variables) are used for reconstruction while the others (the passive variables) are ignored by the decoder. While this regime is well-studied for "standard" VAEs given a single data example, a more general analysis of its properties and impact is lacking. This work extends the polarised regime to a larger family of VAEs (including identifiable VAEs) and to multiple data examples. After analysing the properties of this extended version, we prove that latent representations of different models learned in a polarised regime have the same number of active and passive variables when they have a high representational similarity. We further show that the polarised regime can be used to explain discrepancies between mean and sampled representations and predict a good number of latent dimensions for VAEs and deterministic Autoencoders (AEs). Overall, we demonstrate that the polarised regime is a valuable tool which can be used to assess, explain, and improve the quality of the latent representations learned by VAEs.

ACKNOWLEDGMENTS

I am very grateful to Dr Marek Grzes for being a caring and dedicated supervisor. You allowed me to experiment and learn from my mistakes while being supportive and helping me find solutions when I got stuck. I could not have wished for a better supervisor. This endeavour would also not have been possible without my examiners, Dr. Ramaswamy and Dr. Lorenzi. Thank you for your interest in this work and its potential extensions. Your kind and constructive feedback was deeply appreciated.

I want to express my deepest gratitude to my partner Nathanël Melouki for his unwavering support during my PhD journey. Thank you for being there during all these years. I appreciate your patience with my (sometimes) hectic schedule. Thank you for the cosy evening when I was stressed and the happy dinners when I had things to celebrate.

I am thankful to Declan Collins, who took the time to proofread my thesis and many of my papers. Thank you for your insightful explanations of the various English grammar and syntax points that escaped me over the years!

Many thanks to Théophile Champion for his helpful discussions on mathematics. You gave me the courage to dig deeper into algebra and probability topics that would otherwise still elude me to this day.

Thanks should also go to Peter Clapham for his interest in my research and for proofreading my papers. I enjoyed our discussions on VAEs.

I want to thank my panel members, especially Professor Howard Bowman, for their interest in my work and encouragement, as well as the anonymous reviewers who helped improve the quality of various chapters of this thesis.

Finally, I want to thank my parents, Mercedes and Pierre, and my friend Tiphaine Bisson for their support and encouragement.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS AND ABBREVIATIONS

AE	Autoencoder	ii
a.k.a.	also known as	
BCE	Binary Cross Entropy	14
CDF	Cumulative Distribution Function	8
CKA	Centred Kernel Alignment	xi
ELBO	Evidence Lower Bound	19
FONDUE	Facilely Obtaining the Number of latent Dimensions by Unsupervised Estimation	xiii
GAN	Generative Adversarial Network	127
IB	Information Bottleneck	30
ICA	Independent Components Analysis	37
ID	Intrinsic Dimension	xiv
IDE	Intrinsic Dimension Estimate	xiv
iVAE	identifiable VAE	3
KLD	Kullback-Liebler Divergence	viii
MI	Mutual Information	v

MLE	Maximum Likelihood Estimation	26
MSE	Mean Squared Error	20
NLP	Natural Language Processing	132
OOD	Out Of Distribution	132
PCA	Principal Components Analysis	2
PPCA	Probabilistic Principal Components Analysis	26
PDF	Probability Density Function	xxii
PMF	Probability Mass Function	8
RBF	Radial Basis Function	101
RD	Rate-Distortion	viii
RV	Rho Vector	11
SGD	Stochastic Gradient Descent	20
SPI	Structure Preservation Index	151
s.t.	such that	
SVD	Singular Value Decomposition	27
SVCCA	Singular Value Canonical Correlation Analysis	77
TC	Total Correlation	x
TwoNN	Two Nearest Neighbours	97
VAE	Variational Autoencoder	ii
VGON	Variational Gradient Origin Network	133
VI	Variational Inference	18
w.r.t.	with respect to	20

LIST OF SYMBOLS

General symbols

x	A scalar.

$\mathbf{x}$	A vector.
$a \triangleq b$	a is defined as b.
$\forall x$	For all x.
$\exists x$	There exists an x.
$\log(\cdot)$	Natural logarithm unless otherwise specified.
$\hat{\mathbf{x}}$	An approximation of the vector $\mathbf{x}$.
$\mathbf{x}^{(i)}$	A specific data point.
$\boldsymbol{X}$	A matrix.
$\boldsymbol{I}_n$	The $n \times n$ identity matrix.
$\mathbb{X}$	A set.
$\mathbb{R}$	The set of real numbers.
$\mathrm{diag}[\boldsymbol{X}]$	The diagonal of $\boldsymbol{X}$.
$\mathrm{diag}[\mathbf{x}]$	The matrix $\boldsymbol{X}$ with $\mathbf{x}$ on its diagonal.

CALCULUS

$\int f(\mathbf{x}) d\mathbf{x}$	Integral over the entire domain of $\mathbf{x}$.
∇	Gradient.
$\frac{df}{dx}$	Derivative of f w.r.t. x.
$\frac{\delta f}{\delta x}$	Partial derivative of f w.r.t. x.

LINEAR ALGEBRA

$\mathrm{Tr}(\boldsymbol{X})$	The trace of $\boldsymbol{X}$.
$\det(\boldsymbol{X})$	The determinant of $\boldsymbol{X}$.
$\lVert\cdot\rVert_F$	The Frobenius norm.
$\lVert\cdot\rVert_*$	The nuclear norm.
$\boldsymbol{A} \odot \boldsymbol{B}$	The Hadamard product.

PROBABILITY

$\mathbf{X}$	A random variable.
$\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$	$\mathbf{X}$ is conditionally independent of $\mathbf{Y}$ given $\mathbf{Z}$.
P	A probability measure, or a probability distribution.
$p(\mathbf{x})$	A Probability Density Function (PDF).
$p_{\boldsymbol{\theta}}(\cdot)$	A PDF parametrised by $\boldsymbol{\theta}$.
$\mathbb{E}_{p(\mathbf{x})}[f(\mathbf{x})]$	The expected value of $f(\mathbf{x})$ w.r.t. the random variable $\mathbf{X}$ with PDF $p(\mathbf{x})$.
$Var[\mathbf{x}]$	The variance of $\mathbf{x}$.
$Cov[\mathbf{x}, \mathbf{y}]$	The covariance between $\mathbf{x}$ and $\mathbf{y}$.
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$	The normal distribution with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}$.

1 INTRODUCTION

1.1 MOTIVATION

How can we convert our human knowledge into a form that is understandable by computers? Representation learning has been trying to answer this question for decades. While some progress has been made, the best state-of-the-art deep learning models are still far away from "intelligent machines" able to understand their environment. These powerful models can learn representations that are stored in embeddings and reused as input for various downstream tasks. One of the advantages of this method is that the models performing the downstream tasks will be able to focus on the task objective without having to learn the representations needed for that task. For example, a model whose learning objective is to recognise pictures of dogs and cats will be able to distinguish more quickly a cat from a dog if it is given pre-computed representations. Indeed, it will be able to focus on its classification task without needing to learn what an image is and what is important in it. Thus, if a representation is easy enough to interpret, the model can have a simple architecture (e.g., decision tree, logistic regression) and provide fast and efficient results. However, what can be easily interpreted by an algorithm may not be semantically meaningful—and thus, easily understandable—to humans. This lack of transparency is a critical issue, as it can result in undetected biases in the representations that impact the downstream task models (Bolukbasi et al., 2016; Caliskan, Bryson and Narayanan, 2017; Wilson, Hoffman and Morgenstern, 2019) and in turn people's life. For example, self-driving cars are less likely to detect people with darker skin (Wilson, Hoffman and Morgenstern, 2019), Amazon automated hiring algorithm (now taken down) favoured male applicants (Martin, 2022, Chap. 7.1), and facial recognition could not distinguish non-white persons properly (Martin, 2022, Chap. 3.6).

To counter this problem, a very active area of research is dedicated to the post-hoc evaluation of models' fairness (Barocas, Hardt and Narayanan, 2019; Garg, Villasenor and Foggo, 2020). However, biased representations can be very hard to debias after training (Gonen and Goldberg, 2019). Therefore, improving the way representations are learned is crucial for trustworthy machine learning applications. A first step towards this goal would be to have more transparent representations to make it easier to decide whether an algorithm can be trusted or not. This is the objective pursued by disentangled representation learning, which aims to provide representations where each factor is encoded separately. Let us consider the example of the automated hiring algorithm and imagine that the latent representation learned contains two main features: programming expertise and gender. An entangled model will encode both features at the same time such that programming expertise will vary with gender, while a disentangled model will encode programming expertise and gender separately. One can directly see that giving entangled representations to an automated hiring algorithm will almost surely result in gender bias. However, given disentangled representations, one can remove the sensitive attribute gender by pruning the dimensions of the embedding encoding it and avoid biasing any downstream algorithm. In fact, disentangled representations have been empirically demonstrated to be fairer, even without any post-processing (Locatello et al., 2019b).

The most studied models in the context of disentangled representation learning are VAEs (Kingma and Welling, 2014). These deep learning models are composed of an encoder and a decoder. The former learns to encode the latent factors and the latter to generate data from this latent representation. While initially thought to natively learn unsupervised disentangled representations (Higgins et al., 2017; Burgess et al., 2018; Kim and Mnih, 2018; Chen et al., 2018), this have been shown to be impossible without further inductive biases (Locatello et al., 2019b). Disentangled representation learning was later proved to be possible when introducing some level of supervision (Locatello et al., 2020; Khemakhem et al., 2020; Mita, Filippone and Michiardi, 2021), and is conjectured to be possible in unsupervised models for datasets with specific structures (Zietlow, Rolinek and Martius, 2021; Reizinger et al., 2022). This eluding property of VAEs, along with other unexpected behaviours such as posterior collapse (Bowman et al., 2016; Lucas et al., 2019a; He et al., 2019; Lucas et al., 2019b; Dai, Wang and Wipf, 2020), led to more in-depth analysis of the model's behaviour (Dai et al., 2017; Dai and Wipf, 2018; Dai et al., 2018; Lucas et al., 2019b; Rolinek, Zietlow and Martius, 2019) which revealed a sparsity inducing mechanism present in VAEs models: the polarised regime. This mechanism is akin to a local Principal Components Analysis (PCA) behaviour where any superfluous dimensions of the latent representations are discarded while the remaining active variables are used by the decoder. While very interesting, this property has been scarcely studied and, generally, only in the narrow context of posterior collapse.

Because the behaviour of VAEs is yet far from understood (Dai, Wang and Wipf, 2020; Zietlow, Rolinek and Martius, 2021; Reizinger et al., 2022), we believe that gaining more insights into the underlying mechanisms of these models is a necessary step to improve the quality of latent representations. Thus, this thesis will focus on further studying the polarised regime and its implications for VAEs.

1.2 RESEARCH GOAL AND THESIS STRUCTURE

As stated above, the goal of this thesis is to analyse VAEs through the lens of the polarised regime. Specifically, we will demonstrate that the polarised regime is a generalisable and consistent mechanism that can be used as a tool to understand and improve the latent representations learned by VAEs. Overall, our contributions are as follows:

- In Chapter 5, we demonstrate that the polarised regime, which is only studied in the context of VAEs with a standard Gaussian prior, is also present in VAEs whose Gaussian prior's parameters are learned, such as identifiable VAEs (iVAEs).
- In Chapter 6, we will study the polarised regime with multiple data examples and use the insights gained from this extended version to explain the discrepancies in disentanglement scores observed by Locatello et al. (2019b) between the mean and sampled representations of VAEs.
- In Chapter 7, we will observe the representational similarity of the representations learned by different VAEs and show that, when the latent representations of different models are highly similar, they have the same number of active and passive variables.
- Finally, in Chapter 8, we will show that the polarised regime can also be used as a tool to improve VAEs. Specifically, we will combine the results of Chapters 6 and 7 to design an unsupervised algorithm which can determine a good number of dimensions to use for the latent representations of VAEs trained on a specific dataset. Furthermore, we will show that this number of dimensions can also be applied to deterministic AEs and extended in a number of ways.

The thesis is divided into three parts: Part I introduces the concepts that will be used in the rest of the thesis and review the existing literature on latent representation learning with VAEs, Part II presents the contributions listed above, and Part III summarises our findings and discusses possible extensions of this work. The global organisation of the thesis and the interaction between the different chapters is illustrated in Figure 1.1.

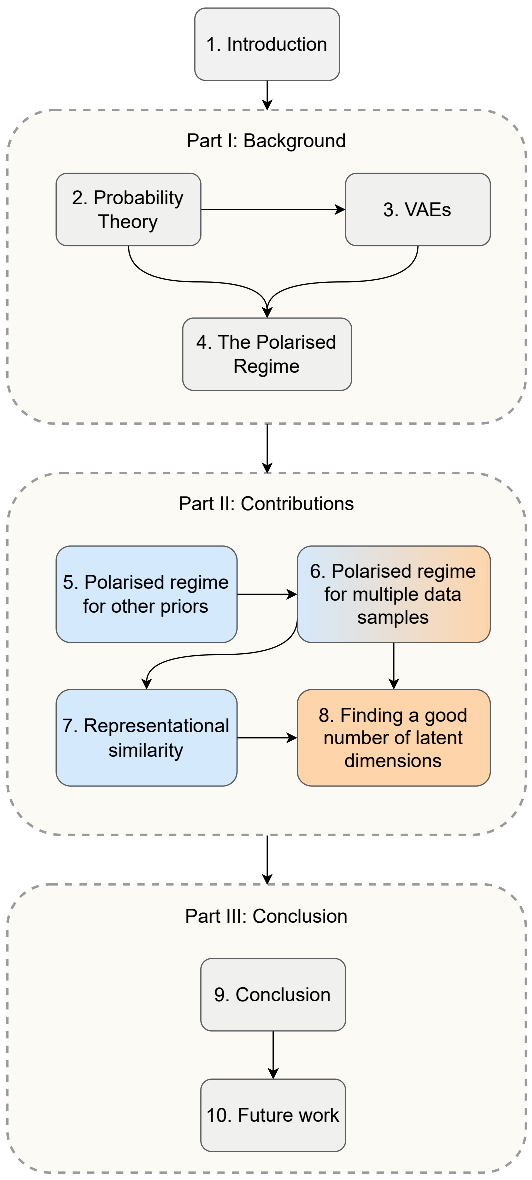

1. Introduction
Part I: Background
2. Probability Theory
3. VAEs
4. The Polarised Regime
Part II: Contributions
5. Polarised regime for other priors
6. Polarised regime for multiple data samples
7. Representational similarity
8. Finding a good number of latent dimensions
Part III: Conclusion
9. Conclusion
10. Future work

Figure 1.1: The overall organisation of this thesis. Any arrow pointing from some chapter A to another chapter B indicates that concepts seen in Chapter A will inform the results obtained in Chapter B. Grey nodes indicate chapters introducing or summarising existing work, blue nodes indicate contributions extending or analysing the polarised regime, and orange nodes indicate contributions to the analysis and improvements of VAEs using the polarised regime. Any node with multiple colours combines the topics with the corresponding colours.

Part I

BACKGROUND

2

PROBABILITY THEORY

As mentioned in Chapter 1, this thesis will focus on VAEs (Kingma and Welling, 2014), which are probabilistic generative models. To allow an accurate study of these models, this section will formalise a number of concepts from probability theory that will be used in the remaining chapters.

2.1 PROBABILITY SPACE

Let us consider the simple experiment of a single coin toss. The outcome can be either heads or tails, thereafter denoted h and t. Together, the set of all outcomes forms the *sample space* $\Omega = \{h, t\}$. One could list all the potential *events* as neither heads nor tails, heads, tails, and either heads or tails. Putting together all these events in set notation, we can define the *event space* $\mathcal{A} = \{\emptyset, \{h\}, \{t\}, \{h, t\}\}$. To evaluate our degree of belief in each event $\mathbb{A}$, we can associate a *probability measure* $P(\mathbb{A})$ to them s.t. all the values of $P(\mathbb{A})$ lie in the interval $[0, 1]$ and $P(\Omega) = 1$. Assuming that the coin is fair, we have $P(\mathbb{A} = \{h\}) = P(\mathbb{A} = \{t\}) = 0.5$ (i.e., the toss has equal chances of being heads or tails), $P(\mathbb{A} = \{h, t\}) = P(\Omega) = 1$ (i.e., the toss will always result in either heads or tails), and $P(\mathbb{A} = \emptyset) = 0$ (i.e., the toss will always have an outcome). The triplet $(\Omega, \mathcal{A}, P)$ forms the *probability space*. We will now formally define these concepts.

Definition 2.1.1 (Sample space)**.** The sample space Ω is the set of all possible outcomes of an experiment.

Using the sample space, one can then define events.

Definition 2.1.2 (Event)**.** An event $\mathbb{A}$ is a set of outcomes s.t. $\mathbb{A} \subseteq \Omega$.

The collection of events will form the event space.

Definition 2.1.3 (Event space)**.** The event space $\mathcal{A}$ is the σ-algebra of the set Ω. It means that $\mathcal{A}$ is a set of events $\mathbb{A}$ with the following properties:

- $\Omega \in \mathcal{A}$,
- *Closure under complement*: if $\mathbb{A} \in \mathcal{A}$ then its complement $\mathbb{A}^C \in \mathcal{A}$,
- *Closure under countable unions*: given the collection $\{\mathbb{A}_i\}_{i\in\mathbb{N}}$, $\bigcup_{i\in\mathbb{N}} \mathbb{A}_i \in \mathcal{A}$.

Note that the event space associated with a discrete sample space is generally its power set $\wp(\Omega)$, as illustrated by our previous example. However, when Ω is the set of real numbers, $\mathcal{A}$ will be the Borel σ-algebra: the smallest σ-algebra containing all open intervals on $\mathbb{R}$ (Deisenroth, Faisal and Ong, 2020). Any set in the Borel σ-algebra is called a Borel set. The pair $(\Omega, \mathcal{A})$ will form a measurable space on which we can apply a probability measure.

Definition 2.1.4 (Probability measure)**.** A probability measure P is a set function that assigns a real number $P(\mathbb{A})$ to each event and has the following properties:

- $P(\mathbb{A}) \in [0, 1]\ \forall \mathbb{A} \in \mathcal{A}$,
- $P(\emptyset) = 0$ and $P(\Omega) = 1$,
- given a collection of pairwise disjoint sets $\{\mathbb{A}_i\}_{i=1}^{\infty}$, $P\Big(\bigcup_{i=1}^{\infty} \mathbb{A}_i\Big) = \sum_{i=1}^{\infty} P(\mathbb{A}_i)$.

The triplet $(\Omega, \mathcal{A}, P)$ will form a probability space that can be used to model experiments. Note that we will also refer to P as the probability distribution of a random variable in Section 2.2.

2.2 RANDOM VARIABLES

In machine learning we rarely directly refer to the probability space. Instead, we use random variables to transform one probability space into another. For example, let us consider an experiment where we toss two coins, $\Omega = \{hh, ht, th, tt\}$. If we were interested in counting the number of heads obtained, we could define a target sample space $\Psi = \{0, 1, 2\}$, a mapping $\mathbf{X} : \Omega \to \Psi$ s.t. $\mathbf{X}(hh) = 2$, $\mathbf{X}(tt) = 0$, $\mathbf{X}(ht) = \mathbf{X}(th) = 1$ and define $\mathcal{T} = \wp(\Psi)$. This function $\mathbf{X}$ is called a *random variable*.

Definition 2.2.1 (Random Variable)**.** Given a probability space $(\Omega, \mathcal{A}, P)$ and a measurable space $(\Psi, \mathcal{T})$, a random variable is a measurable function $\mathbf{X} : \Omega \to \Psi$. It means that $\forall\ \mathbb{T} \in \mathcal{T}, \mathbf{X}^{-1}(\mathbb{T}) \in \mathcal{A}$.

We can endow this newly created measurable space with a probability measure using the pushforward of P:

Definition 2.2.2 (Induced probability measure)**.** Let $\mathbf{X}$ be a random variable from $(\Omega, \mathcal{A}, P)$ to $(\Psi, \mathcal{T})$, and let $Q(\mathbb{T}) = P(\mathbf{X}^{-1}(\mathbb{T}))$ where $\mathbb{T} \in \mathcal{T}$. Q is the probability measure induced by X from P, and $(\Psi, \mathcal{T}, Q)$ is a probability space.

Remark. In this thesis, we will only consider continuous random variables where $\Psi = \mathbb{R}$ or $\Psi = \mathbb{R}^d$. This implies that $\mathcal{T}$ will always be the Borel σ-algebra of Ψ. From now on, unless otherwise specified, we will assume that the random variables considered are always continuous and will refer to them as random variables instead of continuous random variables. Furthermore, we will not present concepts specific to discrete random variables like Probability Mass Functions (PMFs).

2.3 DISTRIBUTION OF A RANDOM VARIABLE

Random variables have associated PDFs and Cumulative Distribution Functions (CDFs) from which one can express its distribution. For clarity, we will present the univariate case, but the multivariate case is obtained in the same way by using a vector of $x \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^d$. Let us first define the PDF.

Definition 2.3.1 (Probability density function)**.** A function $p : \mathbb{R} \to \mathbb{R}$ is a probability density function if:

- $p(x) \geqslant 0 \ \forall x \in \mathbb{R}$,
- p is integrable on $\mathbb{R}$,
- $\int_{\mathbb{R}} p(x)dx = 1$.

For conciseness, $\int_{\mathbb{R}}$ will simply be denoted as $\int$. The CDF can be expressed as the integral of the PDF as follows:

Definition 2.3.2 (Cumulative distribution function)**.** A function $F : \mathbb{R} \to [0, 1]$ is a cumulative distribution function if:

- $F(x) = \int_{-\infty}^{x} p(x)dx$,
- $\lim\limits_{x \to -\infty} F(x) = 0$,
- $\lim\limits_{x \to \infty} F(x) = 1$.

From Definitions 2.3.2 and 2.3.1, we can now define the absolutely continuous probability distribution P of a random variable.

Definition 2.3.3 (Absolutely continuous distribution)**.** The probability distribution of a random variable $\mathbf{X}$ is absolutely continuous if there is a PDF p and a CDF F s.t. for each $[a, b] \subset \mathbb{R}$:

$$P(a \leqslant x \leqslant b) = F(b) - F(a) = \int_a^b p(x)dx, \tag{2.1}$$

where $a, b \in \mathbb{R}$.

Here, because P is the probability measure from Definitions 2.1.4 and 2.2.2, we could have written $P(\mathbf{X}^{-1}([a, b]))$ instead of $P(a \leqslant x \leqslant b)$. An example of a well-known absolutely continuous probability distribution is the Gaussian distribution presented in Section 2.7.

Remark. We will generally write $\mathbf{X} \sim p(x)$ to indicate that the PDF of the univariate random variable $\mathbf{X}$ is $p(x)$ and $\mathbf{X} \sim p(\mathbf{x})$ for a multivariate $\mathbf{X}$. In the same way, $\mathbf{X} \sim P(x)$ and $\mathbf{X} \sim P(\mathbf{x})$ indicate the probability distributions of the univariate or multivariate random variable $\mathbf{X}$.

2.4 PROPERTIES OF THE PROBABILITY DENSITY FUNCTION

We will now consider several fundamental properties of the PDF that will frequently be used in the rest of this work. Let us consider two random variables, $\mathbf{X} \sim p(\mathbf{x})$ and $\mathbf{Y} \sim p(\mathbf{y})$, their joint PDF $p(\mathbf{x}, \mathbf{y})$, and the conditional PDF of $\mathbf{x}$ given $\mathbf{y}$ as $p(\mathbf{x}|\mathbf{y})$. The joint PDF can be decomposed using the product rule:

Definition 2.4.1 (Product rule)**.** $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}|\mathbf{y})p(\mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$.

Using Definition 2.4.1, we can readily obtain Bayes' theorem:

Definition 2.4.2 (Bayes theorem)**.**

$$\underbrace{p(\mathbf{x}|\mathbf{y})}_{Posterior} = \frac{\overbrace{p(\mathbf{y}|\mathbf{x})}^{Likelihood}\ \overbrace{p(\mathbf{x})}^{Prior}}{\underbrace{p(\mathbf{y})}_{Evidence}}$$

The joint and conditional PDF can also be related to the PDFs of both random variables using the sum rule.

Definition 2.4.3 (Sum rule)**.** $p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{y})d\mathbf{y} = \int p(\mathbf{x}|\mathbf{y})p(\mathbf{y})d\mathbf{y}$.

In this context, we say that $p(\mathbf{x})$ is a marginal distribution. When $\mathbf{x}$ does not provide any information about $\mathbf{y}$ and vice-versa, we say that $\mathbf{X}$ and $\mathbf{Y}$ are statistically independent and write $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}$.

Definition 2.4.4 (Statistical independence)**.** Two random variables $\mathbf{X}$ and $\mathbf{Y}$ are statistically independent if and only if for all $\mathbf{x}, \mathbf{y}$ we have $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$. Thus, $p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x})$ and $p(\mathbf{y}|\mathbf{x}) = p(\mathbf{y})$.

When $\mathbf{X}$ and $\mathbf{Y}$ have no influence on each other given a third random variable $\mathbf{Z}$, we say that they are conditionally independent and write $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$. Note that statistically independent variables are generally not conditionally independent and vice versa, as illustrated in Figure 2.1.

Definition 2.4.5 (Conditional independence)**.** Two random variables $\mathbf{X}$ and $\mathbf{Y}$ are conditionally independent given a third random variable $\mathbf{Z}$ if and only if for all $\mathbf{x}, \mathbf{y}, \mathbf{z}$ we have $p(\mathbf{x}, \mathbf{y}|\mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{y}|\mathbf{z})$. Thus, $p(\mathbf{x}|\mathbf{y}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})$ and $p(\mathbf{y}|\mathbf{x}, \mathbf{z}) = p(\mathbf{y}|\mathbf{z})$.

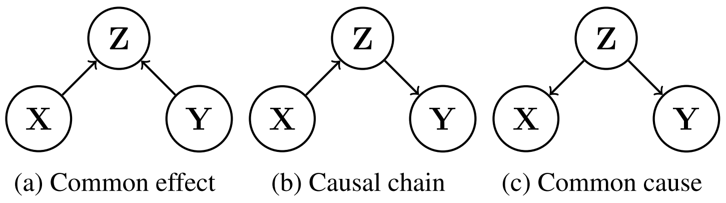


Figure 2.1: Examples of statistical and conditional independence. In (a) $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}$, but $\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}$.
In (b) and (c) $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}$, but $\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y}$.

2.5 SUMMARY STATISTICS

Summary statistics are used to compare and summarise the properties of random variables. A fundamental summary statistic for machine learning is the expected value (a.k.a. expectation).

Definition 2.5.1 (Expected value)**.** The expected value of a random variable $\mathbf{X} \sim p(x)$ is $\mathbb{E}_{\mathbf{X}}[x] = \int xp(x)dx$. When $\mathbf{x} \in \mathbb{R}^d$, the expected value of a random variable $\mathbf{X} \sim p(\mathbf{x})$ is $\mathbb{E}_{\mathbf{X}}[\mathbf{x}] = \left[\int x_1 p(x_1)dx_1, \cdots, \int x_d p(x_d)dx_d\right]$, where x_i denotes the i^{th} dimension of $\mathbf{x}$.

This will generally be referred to as the mean of a random variable and can be approximated by averaging the observed $\mathbf{x}$. The expected value is a linear operator. Thus, given two random variables $\mathbf{X} \sim p(\mathbf{x})$ and $\mathbf{Y} \sim p(\mathbf{y})$ and two constants a and b, $\mathbb{E}[a\mathbf{x} + b\mathbf{y}] = a\mathbb{E}[\mathbf{x}] + b\mathbb{E}[\mathbf{y}]$. The expected value can be generalised to measurable functions thanks to the law of the unconscious statistician:

Definition 2.5.2 (Law of the unconscious statistician)**.** Given a measurable function f of a random variable $\mathbf{X} \sim p(\mathbf{x})$,

$$\mathbb{E}_{\mathbf{X}}[f(\mathbf{x})] = \left[\int f(x_1)p(x_1)dx_1, \cdots, \int f(x_d)p(x_d)dx_d \right].$$

As the expectation is an integral, we can use Jensen's inequality to relate it with convex and concave functions.

Definition 2.5.3 (Jensen's inequality for random variables)**.** Given a random variable $\mathbf{X} \sim p(\mathbf{x})$, a convex function f, and a concave function g, we have $f(\mathbb{E}[\mathbf{x}]) \leqslant \mathbb{E}[f(\mathbf{x})]$ and $g(\mathbb{E}[\mathbf{x}]) \geqslant \mathbb{E}[g(\mathbf{x})]$.

This is frequently applied to the concave $\log$ function in probabilistic settings, as we will see in Chapter 3. Having information about one random variable is good, but one may also want to monitor the relationship between two random variables. One way to do this is to measure their joint variability.

Definition 2.5.4 (Covariance)**.** The covariance of two random variables, $\mathbf{X} \sim p(\mathbf{x})$ and $\mathbf{Y} \sim p(\mathbf{y})$, is defined as $\mathrm{Cov}[\mathbf{x}, \mathbf{y}] = \mathbb{E}[\mathbf{x}\mathbf{y}^T] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{y}]^T = \mathrm{Cov}[\mathbf{y}, \mathbf{x}]^T$.

The spread of a random variable can be measured in a similar way by the variance.

Definition 2.5.5 (Variance)**.** The variance of a random variable $\mathbf{X} \sim p(\mathbf{x})$ is defined as $\mathrm{Cov}[\mathbf{x}, \mathbf{x}] = \mathrm{Var}[\mathbf{x}] = \mathbb{E}[\mathbf{x}\mathbf{x}^T] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{x}]^T$.

The matrices obtained in Definitions 2.5.4 and 2.5.5 are both symmetric positive semi-definite (i.e., all their eigenvalues are non-negative) and are often further assumed to be full rank, becoming positive definite (i.e., all their eigenvalues are positive). This gives them some convenient properties that can be exploited during derivations, as, for example, in Chapter 6. We can combine Definitions 2.5.4 and 2.5.5 to obtain a standardised measure of the relationship between two random variables.

Definition 2.5.6 (Correlation)**.** The correlation ρ of two random variables $\mathbf{X} \sim p(\mathbf{x})$ and $\mathbf{Y} \sim p(\mathbf{y})$ is defined as $\rho[\mathbf{x}, \mathbf{y}] = \frac{\mathrm{Cov}[\mathbf{x}, \mathbf{y}]}{\sqrt{\mathrm{Var}[\mathbf{x}]\mathrm{Var}[\mathbf{y}]}}$.

For centred variables, Definition 2.5.6 corresponds to the cosine of the angle between $\mathbf{x}$ and $\mathbf{y}$. It can be further generalised to matrices using the Rho Vector (RV) coefficient (Escoufier, 1973; Robert and Escoufier, 1976; Josse and Holmes, 2016). In the deep learning community, this metric is also known as linear CKA (Cortes, Mohri and Rostamizadeh, 2012; Kornblith et al., 2019).

Definition 2.5.7 (RV coefficient)**.** The RV coefficient is a generalisation of the correlation coefficient to multivariate data $\boldsymbol{X}$ and $\boldsymbol{X}$ such that

$$\rho V(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\|\boldsymbol{X}^T\boldsymbol{Y}\|_F^2}{\|\boldsymbol{X}^T\boldsymbol{X}\|_F\|\boldsymbol{Y}^T\boldsymbol{Y}\|_F}, \tag{2.2}$$

where $\|\cdot\|_F$ is the Frobenius norm.

The RV coefficient can be thought of as the cosine of the angle between matrices, where $0 \leqslant \rho V \leqslant 1$ and $\rho V(\boldsymbol{X}, \boldsymbol{Y}) = 0$ if and only if there is no covariance between $\boldsymbol{X}$ and $\boldsymbol{Y}$. Furthermore, for one-dimensional $\boldsymbol{X}$ and $\boldsymbol{Y}$, the RV coefficient is equal to the square value of the correlation coefficient. The RV coefficient is also invariant to orthogonal transformations and isotropic scaling. In other words, $\rho V(\boldsymbol{X}, a\boldsymbol{B}\boldsymbol{X} + \boldsymbol{c}) = 1$, where $\boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{B}^T\boldsymbol{B} = \boldsymbol{I}$, a is a constant, and $\boldsymbol{c}$ is a constant vector. Non-linear versions of the RV coefficient have also been developed using kernels (Purdom, 2006; Cortes, Mohri and Rostamizadeh, 2012), however, we will see in Chapter 7 that linear CKA (a.k.a. RV coefficient) is sufficient for monitoring representational similarity of deep neural networks.

2.6 STATISTICAL DIVERGENCES

In addition to the summary statistics, one may be interested in assessing how different two probability distributions are from each other. This can be measured by statistical divergences.

Definition 2.6.1 (Statistical divergence)**.** A statistical divergence $D(P \parallel Q)$ measures the discrepancies between two probability distributions, P and Q, s.t.:

- $D(P \parallel Q) \geqslant 0$,
- $D(P \parallel Q) = 0 \iff P = Q$,
- The dual divergence of $D(P \parallel Q)$ is $D^*(P \parallel Q) \triangleq D(Q \parallel P)$,
- The symmetrised divergence of $D(P \parallel Q)$ is $D^S(P \parallel Q) \triangleq \frac{1}{2}\Big(D^*(P \parallel Q) + D(P \parallel Q)\Big)$.

Remark. In general, statistical divergences are not metrics. Indeed, they can be asymmetric, with $D(P \parallel Q) \neq D(Q \parallel P)$. Moreover, in most of the cases, they do not fulfill the triangle inequality and we have $D(P \parallel Q) \nleqslant D(P \parallel R) + D(R \parallel Q)$.

An example of statistical divergence is the f-divergence (Ali and Silvey, 1966; Csiszár, 1967), which generalises many other divergences such as the Hellinger distance (Hellinger, 1909; Kailath, 1967), the Jeffreys divergence (Jeffreys, 1946), the Jensen-Shannon divergence (Lin, 1991), or the Kullback-Liebler Divergence (Kullback and Leibler, 1951). Note that other generalisations of these divergences exist (Rényi, 1961), but we will not present them here for conciseness.

Definition 2.6.2 (f-divergence)**.** Given a convex function $\mathrm{f} : [0, \infty) \to \mathbb{R}$ satisfying $\mathrm{f}(1) = 0$ and $\mathrm{f}(0) = \lim_{t \to 0} \mathrm{f}(t)$ and two continuous probability distributions P and Q, the f-divergence $D_{\mathrm{f}}(P \parallel Q)$ is defined as

$$D_{\mathrm{f}}(P \parallel Q) \triangleq \int \mathrm{f}\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right) q(\mathbf{x}) d\mathbf{x}. \tag{2.3}$$

The f-divergence has the following properties:

- *Convexity* $D_{\mathrm{f}}(P \parallel Q)$ is convex w.r.t. P and Q.
- *Joint convexity* For any $0 \leq \lambda \leq 1$, $D_{\mathrm{f}}\Big(\lambda P_1 + (1-\lambda) P_2 \parallel \lambda Q_1 + (1-\lambda) Q_2\Big) \leqslant \lambda D_{\mathrm{f}}(P_1 \parallel Q_1) + (1 - \lambda) D_{\mathrm{f}}(P_2 \parallel Q_2)$.
- *Duality* Given $\mathrm{g}(t) = t\mathrm{f}\left(\frac{1}{t}\right)$, $D_{\mathrm{g}}(P \parallel Q) = D_{\mathrm{f}}^{*}(P \parallel Q) = D_{\mathrm{f}}(Q \parallel P)$.

A divergence that will recurrently be used in this work is the KLD, $D_{\mathrm{KL}}(P \parallel Q)$.

Definition 2.6.3 (KL divergence)**.** Given two continuous probability distributions P and Q, the KLD is defined as

$$D_{\mathrm{KL}}(P \parallel Q) \triangleq \int \log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right) p(\mathbf{x}) d\mathbf{x}. \tag{2.4}$$

It can be re-expressed in terms of entropy and interpreted as the information gain one would achieve using P (the true distribution) instead of Q,

$$D_{\mathrm{KL}}(P \parallel Q) = \mathrm{H}(P, Q) - \mathrm{H}(P). \tag{2.5}$$

Furthermore, we can directly see from Equation 2.5 that one can reduce the divergence between the data distribution P and its approximation Q by minimising the cross entropy $\mathrm{H}(P, Q)$, as we do not have any control over the entropy of the data $\mathrm{H}(P)$. When P and Q are Bernoulli distributions, $\mathrm{H}(P, Q)$ is the Binary Cross Entropy (BCE) and has the following form:

$$H(P, Q) = \sum_{i=1}^{n} x^{(i)} \log \hat{x}^{(i)} + (1 - x^{(i)}) \log(1 - \hat{x}^{(i)}), \tag{2.6}$$

where the $x^{(i)}$ are sampled from P and the $\hat{x}^{(i)}$ from Q.

Using Equation 2.4, we can easily show the following:

Proposition 2.6.1. *The forward* KLD *is an* f*-divergence.*

Proof. Starting from Equation 2.4

$$\begin{aligned} D_{\mathrm{KL}}(P \parallel Q) &= \int \log \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} \right) p(\mathbf{x}) d\mathbf{x}, && (2.7)\\ &= \int \log \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} \right) \frac{p(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) d\mathbf{x}, && (2.8)\\ &= \int \mathrm{f}_{\mathrm{KL}} \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} \right) q(\mathbf{x}) d\mathbf{x}, && (2.9)\\ &= D_{\mathrm{f}_{\mathrm{KL}}}(P \parallel Q), && (2.10) \end{aligned}$$

where $\mathrm{f}_{\mathrm{KL}}(t) = t \log t$, and Equation 2.9 is the equation of the f-divergence. □

In addition to the properties inherited from the f-divergence, the KLD is asymmetric and has no upper bound. However, when P and Q are Gaussian distributions, we will see in Section 2.7 that we can derive an analytical solution for Equation 2.4.

Let us now find the expression of the reverse KLD, $D_{\mathrm{KL}}(Q \parallel P)$. Recall the duality property of the f-divergence: given $\mathrm{g}(t) = t\mathrm{f}\left(\frac{1}{t}\right)$, $D_{\mathrm{g}}(P \parallel Q) = D_{\mathrm{f}}(Q \parallel P)$. For the KLD, $\mathrm{f}_{\mathrm{KL}}(t) = t \log t$, so $\mathrm{g}_{\mathrm{KL}}(t) = t\frac{1}{t} \log \frac{1}{t} = -\log t$.
Thus, we can obtain the reverse KLD as follows:

$$\begin{aligned} D^{*}_{\mathrm{KL}}(P \parallel Q) &= \int \mathrm{g}_{\mathrm{KL}} \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} \right) q(\mathbf{x}) d\mathbf{x}, && (2.11)\\ &= \int -\log \frac{p(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) d\mathbf{x}, && (2.12)\\ &= \int \log \frac{q(\mathbf{x})}{p(\mathbf{x})} q(\mathbf{x}) d\mathbf{x}, && (2.13)\\ &= D_{\mathrm{KL}}(Q \parallel P). && (2.14) \end{aligned}$$

As illustrated in Figure 2.2, the forward and reverse KLDs have very different behaviours on multimodal distributions. Because the $\log$ of Equation 2.13 will be infinite when $p(\mathbf{x}) = 0$ and $q(\mathbf{x}) > 0$, the reverse KLD will restrict Q to a specific mode to avoid this case. On the other hand, the log of Equation 2.4 will be infinite when $q(\mathbf{x}) = 0$ and $p(\mathbf{x}) > 0$. Thus the forward KLD will seek to cover any part of the space where p is non-zero (Murphy, 2022, pp. 208-209). We will see in Chapter 3 that the reverse KLD plays an important role in the learning objective of VAEs.

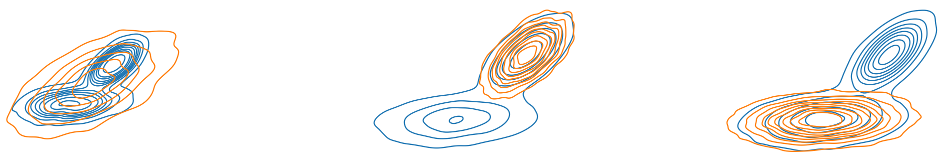

(a) Forward KLD (b) Reverse KLD locked on the first mode (c) Reverse KLD locked on the second mode

Figure 2.2: The difference between the zero-avoiding forward KLD shown in (a) and the mode-seeking reverse KLDs shown in (b) and (c). On all figures P is in blue and Q is in orange. This illustration is inspired by Bishop (2006); Murphy (2022).

2.7 THE MULTIVARIATE GAUSSIAN DISTRIBUTION

As we will see in Chapter 3, the multivariate Gaussian distribution (a.k.a. multivariate normal distribution) has a very important place in the study of VAEs. This section will present some properties of this distribution which will regularly be used in the following chapters.

Definition 2.7.1 (Gaussian distribution)**.** The Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$ is a continuous probability distribution with the following PDF:

$$p(\mathbf{x}) = (2\pi)^{-d/2} \det(\boldsymbol{\Sigma})^{-1/2} \exp\Big(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\Big). \tag{2.15}$$

The standard Gaussian distribution is defined as $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, where $\mathbf{0}$ is a vector of zeros and $\boldsymbol{I}$ is the identity matrix. The isotropic Gaussian distribution is defined as $\mathcal{N}(\mathbf{0}, \gamma\boldsymbol{I})$, where $\gamma > 0$.

Any Gaussian distribution can be seen as a standard Gaussian distribution that has been shifted, stretched, and spread according to some new mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Thus, if a random variable is distributed according to a Gaussian with known $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, it is possible to sample from this distribution using reparametrisation (Gentle, 2009, pp. 315-316). We will see in Section 3.3 that Equation 2.16 is crucial for the stability of VAEs.

Definition 2.7.2 (Reparametrisation of multivariate Gaussians)**.** Given a random variable $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and a random variable $\mathbf{E} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, we can sample $\mathbf{x}$ from the distribution of $\mathbf{X}$ by reparametrisation:

$$\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{\epsilon}, \tag{2.16}$$

where $\boldsymbol{A}$ is the real matrix obtained from Choleski decomposition $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{\Sigma}$, allowing the usage of non-diagonal covariance.

As mentioned in Section 2.6, there is also an analytical solution for the KLD between two multivariate Gaussian distributions. Let us consider two n-variate Gaussian distributions $P \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ and $Q \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and their KLD $D_{\mathrm{KL}}\big(P \parallel Q\big)$. Using Equations 2.7 and 2.15, we have:

$$\begin{aligned} D_{\mathrm{KL}}\big(P \parallel Q\big) &= \mathbb{E}_{p(\mathbf{x})}[\log p(\mathbf{x}) - \log q(\mathbf{x})], \\ &= \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\Big[(\mathbf{x}-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1) - (\mathbf{x}-\boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}_0^{-1}(\mathbf{x}-\boldsymbol{\mu}_0)\Big] \\ &\quad + \frac{1}{2}\log\frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)}. \end{aligned}$$

Using the cyclical property of the trace and the fact that the trace of a scalar is the scalar itself, we get

$$\begin{aligned} D_{\mathrm{KL}}\big(P \parallel Q\big) =& \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\Big[\operatorname{Tr}\big((\mathbf{x}-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)\big) \\ &\quad - \operatorname{Tr}\big((\mathbf{x}-\boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}_0^{-1}(\mathbf{x}-\boldsymbol{\mu}_0)\big)\Big] + \frac{1}{2}\log\frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)}, \\ =& \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\Big[\operatorname{Tr}\big(\boldsymbol{\Sigma}_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)(\mathbf{x}-\boldsymbol{\mu}_1)^T\big) \\ &\quad - \operatorname{Tr}\big(\boldsymbol{\Sigma}_0^{-1}(\mathbf{x}-\boldsymbol{\mu}_0)(\mathbf{x}-\boldsymbol{\mu}_0)^T\big)\Big] + \frac{1}{2}\log\frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)}. \end{aligned} \tag{2.17}$$

Using the linearity of the trace and expected value, we can push the expectations inside the trace terms and integrate:

$$\begin{aligned} D_{\mathrm{KL}}\big(P \parallel Q\big) =& \frac{1}{2}\operatorname{Tr}\big(\boldsymbol{\Sigma}_1^{-1}(\mathbb{E}_{p(\mathbf{x})}[\mathbf{x}\mathbf{x}^T] - \mathbb{E}_{p(\mathbf{x})}[2\boldsymbol{\mu}_1\mathbf{x}^T] + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T)\big) \\ &- \frac{1}{2}\operatorname{Tr}\Big(\boldsymbol{\Sigma}_0^{-1}\mathbb{E}_{p(\mathbf{x})}\big[(\mathbf{x}-\boldsymbol{\mu}_0)(\mathbf{x}-\boldsymbol{\mu}_0)^T\big]\Big) + \frac{1}{2}\log\frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)}, \\ =& \frac{1}{2}\operatorname{Tr}\big(\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\Sigma}_0 + \boldsymbol{\mu}_0\boldsymbol{\mu}_0^T - 2\boldsymbol{\mu}_1\boldsymbol{\mu}_0^T + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T)\big) - \frac{1}{2}\operatorname{Tr}\Big(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\Sigma}_0\Big) \end{aligned}$$

$$+ \frac{1}{2} \log \frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)}.$$

Factorising and using once more the trace properties, we finally obtain

$$\begin{aligned} D_{\mathrm{KL}}\big(P \parallel Q\big) =& \frac{1}{2}\Bigg(\operatorname{Tr}(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0) + \operatorname{Tr}\big(\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\big) \\ &+ \log \frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)} - n\Bigg), \\ =& \frac{1}{2}\Bigg(\operatorname{Tr}(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \\ &+ \log \frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)} - n\Bigg). \end{aligned} \tag{2.18}$$

When $Q \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, Equation 2.18 further simplifies to the traditional analytical solution used in VAEs, which we will study in Chapter 3.

$$D_{\mathrm{KL}}\big(P \parallel Q\big) = \frac{1}{2}\Bigg(\operatorname{Tr}(\boldsymbol{\Sigma}_0) + \|\boldsymbol{\mu}_0\|_2^2 - \log\det(\boldsymbol{\Sigma}_0) - n\Bigg). \tag{2.19}$$

3

VARIATIONAL AUTOENCODERS

3.1 WHAT ARE VARIATIONAL AUTOENCODERS?

VAEs (Kingma and Welling, 2014; Rezende, Mohamed and Wierstra, 2014) are probabilistic generative models based on Variational Inference (VI). Akin to AEs, they map some input $\mathbf{x}$ to a latent representation $\mathbf{z}$, which is used by the decoder to generate an output $\hat{\mathbf{x}}$ similar to $\mathbf{x}$. However, we will see in Section 3.2 that in opposition to the deterministic representations of AEs, the latent representations of VAEs are probabilistic, as shown in Figure 3.1.

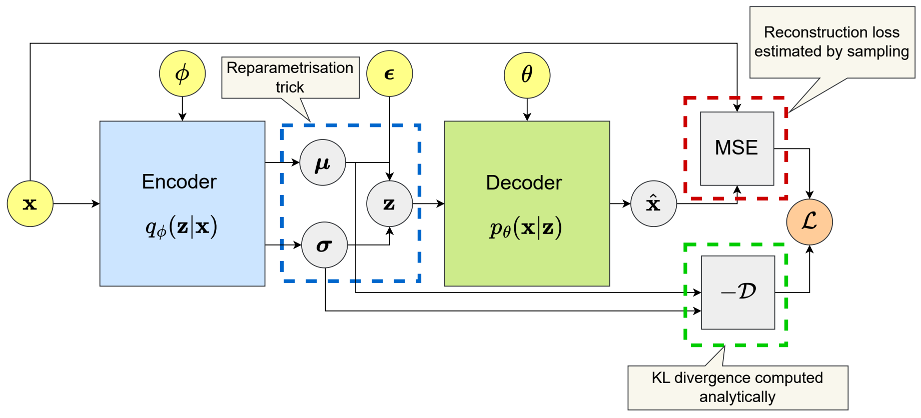
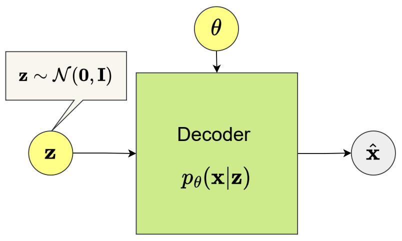
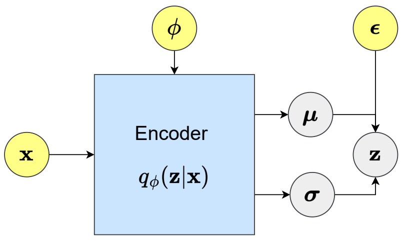


Figure 3.1: Illustration of a VAE during the training process. The distributions are assumed to be multivariate Gaussian. $\boldsymbol{\mu}$ is the mean layer and $\boldsymbol{\sigma}$ is the variance layer. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the parameters of the posterior over $\mathbf{z}$.

After training, one can use the encoder and decoder separately for inference or generation. For generation (Figure 3.2a), only the decoder is used, and any input $\mathbf{z}$ will be sampled from a standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. For inference (Figure 3.2b), the encoder will output the mean, variance and sampled representations corresponding to $\mathbf{x}$, where the deterministic mean representation is generally used as input for downstream tasks (Higgins et al., 2017; Kim and Mnih, 2018; Locatello et al., 2019a).


θ
$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$\mathbf{z}$
Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\hat{\mathbf{x}}$

(a) Generation



(b) Inference

Figure 3.2: Different usages of a VAE after training.

3.2 THE LEARNING OBJECTIVE

Now that we have presented a high-level overview of VAEs, let us provide a more in-depth analysis of their learning objective. The idea is to maximise the probability of $\mathbf{x}$ under the model parametrised by $\boldsymbol{\theta}$, where the latent factors $\mathbf{z}$ can be marginalised out using Definition 2.4.3:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) d\mathbf{z}, \tag{3.1}$$

where $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ is assumed to be computable and continuous on $\boldsymbol{\theta}$. However, one would need to obtain all possible values of $\mathbf{z}$ to solve Equation 3.1, which is, in general, intractable. One can, however, obtain the lower bound of the likelihood, a.k.a. the Evidence Lower Bound (ELBO), as follows:

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\mathbf{x}) &= \log \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} d\mathbf{z}, && (3.2) \\
&\geqslant \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \right], && (3.3) \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \right] - \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z})} \right]. && (3.4)
\end{aligned}$$

The transition from Equation 3.2 to the inequality in Equation 3.3 is obtained by using the concavity of the log function to apply Jensen's inequality, as explained in Section 2.5 and Definition 2.5.3. By realising that the second term of Equation 3.4 corresponds to the KLD from Equation 2.4, the learning objective can be reformulated as a maximisation of the Evidence Lower Bound (ELBO):

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z}) \right)}_{\text{regularisation term}}. \tag{3.5}$$

On one hand, the reconstruction term of Equation 3.5 will penalise unlikely representations of $\mathbf{x}$, ensuring that realistic data is generated. On the other hand, the regularisation term will enforce the posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ to stay reasonably close to the prior $p_{\boldsymbol{\theta}}(\mathbf{z})$, preventing overfitting. $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ will typically be approximated by deep neural networks trained using Stochastic Gradient Descent (SGD). We will thus seek to minimise the negative ELBO, $-\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$.

3.3 THE REPARAMETRISATION TRICK

Usually, $p_{\boldsymbol{\theta}}(\mathbf{z})$ and $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ are assumed to be the PDFs of multivariate Gaussian distributions to permit an analytical computation of the regularisation term from Equation 2.18 (Doersch, 2016). To allow further simplifications from Equation 2.18 to Equation 2.19, $p_{\boldsymbol{\theta}}(\mathbf{z})$ is often set to the PDF of the standard multivariate Gaussian distribution. The reconstruction term is estimated by sampling using Mean Squared Error (MSE) or BCE, depending whether we assume that $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ comes from a multivariate Gaussian distribution or a multivariate Bernoulli (Kingma and Welling, 2014, Appendix C). Thus, the gradient used for the SGD is as follows:

$$\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} - \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = -\underbrace{\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big]}_{\text{estimated by sampling}} + \underbrace{\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)}_{\text{computed analytically}}. \tag{3.6}$$

Because $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ does not depend on $\boldsymbol{\theta}$, the gradient of the first term of Equation 3.6 with respect to (w.r.t.) $\boldsymbol{\theta}$ can be pushed inside the expectation, making it straightforward to evaluate:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big]. \tag{3.7}$$

However, the same thing cannot be done for the gradient of $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ w.r.t. $\boldsymbol{\phi}$, as the expectation is over $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, which is parametrised by $\boldsymbol{\phi}$. Recalling that the derivative of $\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ w.r.t. $\boldsymbol{\phi}$ is $\nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \frac{\nabla_{\boldsymbol{\phi}} q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}$, this gradient can be transformed into a computable form:

$$\begin{aligned}
\nabla_{\boldsymbol{\phi}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big] &= \int \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \nabla_{\boldsymbol{\phi}} q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) d\mathbf{z}, \\
&= \int \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \frac{\nabla_{\boldsymbol{\phi}} q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} d\mathbf{z}, \\
&= \int \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) d\mathbf{z},
\end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})} \left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\right], \\
&\approx \frac{1}{n} \sum_{i=0}^{n} \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}^{(i)}) \nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\mathbf{z}^{(i)}|\mathbf{x}).
\end{aligned}
$$

However, as pointed out by Paisley, Blei and Jordan (2012), this estimator—called the *score function estimator*—can have a high variance[1], making it unusable for VAEs. Thus, Kingma and Welling (2014) used a different approach and isolated the randomness generated by the sampling process using the reparametrisation trick seen in Definition 2.7.2, which has lower variance (Xu et al., 2019). Its goal is to express the latent variable $\mathbf{z}$ as the result of a deterministic function $g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x})$, where $\boldsymbol{\epsilon}$ is sampled from a Gaussian distribution with PDF $p(\boldsymbol{\epsilon})$, providing a way to reformulate the reconstruction term of 3.5 as:

$$
\begin{aligned}
\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big] &= \int \log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) d\mathbf{z}, \\
&= \int \log p_{\boldsymbol{\theta}}(\mathbf{x}|g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}, \\
&= \mathbb{E}_{p(\boldsymbol{\epsilon})}\big[p_{\boldsymbol{\theta}}(\mathbf{x}|g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x}))\big].
\end{aligned}
$$

Therefore, if there exists a differentiable function $g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x})$, the gradient can be computed as:

$$
\nabla_{\boldsymbol{\phi}} \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x}))\right] = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\nabla_{\boldsymbol{\phi}} \log p_{\boldsymbol{\theta}}(\mathbf{x}|g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x}))\right]. \tag{3.8}
$$

Assuming that $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ is the PDF of $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, we have $g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x}) = \boldsymbol{\mu} + \boldsymbol{L}\boldsymbol{\epsilon}$, where $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T$. To further simplify Equation 2.16 and avoid the need for a Choleski decomposition, $\boldsymbol{\Sigma}$ is generally considered diagonal. One can thus simplify the reparametrisation to $g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \mathbf{x}) = \boldsymbol{\mu} + \boldsymbol{\sigma}^{1/2} \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\sigma}^{1/2} = [\sqrt{\Sigma_{11}}, \cdots, \sqrt{\Sigma_{kk}}]$ and $\odot$ is the Hadamard product, leading to the final version of VAE represented in Figure 3.1.

1 We refer the reader to Mohamed et al. (2020, Section 4.3.3) for a detailed discussion on the sources of variance of the score function estimator.

4

THE POLARISED REGIME OF VARIATIONAL AUTOENCODERS

4.1 WHAT IS THE POLARISED REGIME?

The polarised regime—also known as selective posterior collapse—is the ability of VAEs to 'shut down' superfluous dimensions of their sampled latent representations while providing a high precision on other dimensions (Rolinek, Zietlow and Martius, 2019; Dai, Wang and Wipf, 2020). As a result, the sampled representation can be separated into two subsets of variables: active and passive. The active variables correspond to the subset of the sampled latent representation that is needed for reconstruction. They have low variance and are close to the mean representations. The passive variables correspond to the superfluous dimensions that are discarded by the VAE. They follow a zero-mean unit-variance Gaussian distribution to optimally match the prior and are ignored by the decoder, which only uses the variables that help to reconstruct the input.

Based on the definition of Rolinek, Zietlow and Martius (2019), we can characterise the active and passive variables of sampled representations as follows.

Definition 4.1.1 (Polarised regime)**.** When a VAE learns in a polarised regime, for a given data example $\mathbf{x}^{(i)} \in \boldsymbol{X}$, its mean, variance, and sampled representations, $\boldsymbol{\mu}^{(i)} \triangleq \boldsymbol{\mu}(\mathbf{x}^{(i)};\phi)$, $\boldsymbol{\sigma}^{(i)} \triangleq \operatorname{diag}[\boldsymbol{\Sigma}(\mathbf{x}^{(i)};\phi)]$, and $\mathbf{z}^{(i)} \triangleq \boldsymbol{\mu}^{(i)} + \boldsymbol{\epsilon}^{(i)} \odot (\boldsymbol{\sigma}^{(i)})^{1/2}$, respectively, are composed of a set of passive and active variables, $\mathbb{V}_p^{(i)} \cup \mathbb{V}_a^{(i)}$, such that, for each data example $\mathbf{x}^{(i)}$:

1. $|\boldsymbol{\mu}_j^{(i)}| \ll 1$, $\boldsymbol{\sigma}_j^{(i)} \approx 1$, and $\mathbf{z}_j^{(i)} \approx \boldsymbol{\epsilon}_j^{(i)} \quad \forall\, j \in \mathbb{V}_p^{(i)}$,
2. $\boldsymbol{\sigma}_j^{(i)} \ll 1$ and $\mathbf{z}_j^{(i)} \approx \boldsymbol{\mu}_j^{(i)} \quad \forall\, j \in \mathbb{V}_a^{(i)}$,

where $\boldsymbol{\epsilon}^{(i)}$ is sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, j is the jth index of a vector-valued representation, and $|\cdot|$ denotes the absolute value.

The polarised regime can also be seen as a sparsity-inducing mechanism which will prune the superfluous columns of the weights of the first layer of the decoder (Dai et al., 2017, 2018). As the corresponding dimensions of the mean and variance representations will not have any influence on the decoder (i.e., they are passive), they will only be optimised with respect to the KLD, thus becoming close to 0 and 1, respectively.

4.2 A CONSEQUENCE OF THE LEARNING OBJECTIVE AND STANDARD VAE IMPLEMENTATION

The polarised regime has been shown to occur naturally in VAEs as a result of optimising Equation 3.5 with $p_{\boldsymbol{\theta}}(\mathbf{z}) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$, $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}[\boldsymbol{\sigma}])$, and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}})$ (Dai and Wipf, 2018; Dai, Wang and Wipf, 2020). These assumptions about the prior, posterior, and likelihood distributions reflect the "standard" VAE implementation for continuous data. Indeed, in "standard" VAEs, the posterior covariance is represented by a linear layer enforcing its diagonality (Dai et al., 2017; Rolinek, Zietlow and Martius, 2019), and the reconstruction loss is measured by MSE indicating a likelihood with Gaussian distribution (Kingma and Welling, 2014). Moreover, as mentioned in Section 3.1, the prior is set to $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$ to permit the usage of the analytical solution from in Equation 2.19 for the KLD.

To analyse the impact of these assumptions, we will start by deriving an analytical solution for linear VAEs with $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is not enforced to be diagonal in Section 4.2.1. Then, in Section 4.2.2, we will analyse the impact of enforcing a diagonal $\boldsymbol{\Sigma}$.

4.2.1 *Analytical solution of the ELBO for linear VAEs*

The assumptions stated above allow the derivation of a closed form learning objective in the linear case which can be used to gain some insight on the polarised regime defined in Section 4.1. Note that in this section, we do not enforce $\boldsymbol{\Sigma}$ to be diagonal and will compare these results with Section 4.2.2, where $\boldsymbol{\Sigma}$ is diagonal, to assess the impact of enforcing a diagonal covariance.

Using Equation 2.19, we have

$$D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big) = \frac{1}{2}\big(\operatorname{Tr}(\boldsymbol{\Sigma}) + \|\boldsymbol{\mu}\|_2^2 - \log\det(\boldsymbol{\Sigma}) - n\big), \tag{4.1}$$

where n is the dimensionality of $\mathbf{z}$.

Given a linear decoder $Dec(\mathbf{z};\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{W}\mathbf{z}+\mathbf{b}, \gamma\boldsymbol{I}_m)$, where $\mathbf{b} \in \mathbb{R}^{m\times 1}$ is the bias vector and m is the dimensionality of the input $\mathbf{x}$, we have

$$\begin{aligned} p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) =& (2\pi)^{-m/2}\det(\gamma\boldsymbol{I}_m)^{-1/2}\exp\Big(-\frac{1}{2}(\mathbf{x}-\boldsymbol{W}\mathbf{z}-\mathbf{b})^T(\gamma\boldsymbol{I}_m)^{-1} \\ &(\mathbf{x}-\boldsymbol{W}\mathbf{z}-\mathbf{b})\Big), \\ =& (2\gamma\pi)^{-m/2}\exp\Big(-\frac{1}{2\gamma}\|\mathbf{x}-(\boldsymbol{W}\mathbf{z}+\mathbf{b})\|_2^2\Big). \end{aligned} \tag{4.2}$$

Thus,

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) = -\frac{1}{2}\Big(m\log(2\pi) + m\log\gamma + \frac{1}{\gamma}\|\mathbf{x}-(\boldsymbol{W}\mathbf{z}+\mathbf{b})\|_2^2\Big). \tag{4.3}$$

Dropping any constant that cannot be optimised in Equation 4.3, we can reduce the reconstruction term of Equation 3.5 to

$$\begin{aligned} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big] =& -\frac{1}{2}\Big(m\log\gamma + \frac{1}{\gamma}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\|\mathbf{x}-\boldsymbol{W}\mathbf{z}-\mathbf{b}\|_2^2\Big]\Big), \\ =& -\frac{1}{2}\Big(m\log\gamma + \frac{1}{\gamma}\Big(\|\mathbf{x}-\mathbf{b}\|_2^2 + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\mathbf{z}^T\boldsymbol{W}^T\boldsymbol{W}\mathbf{z}\Big] \\ &- \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[(\mathbf{x}-\mathbf{b})^T\boldsymbol{W}\mathbf{z} + \mathbf{z}^T\boldsymbol{W}^T(\mathbf{x}-\mathbf{b})\Big]\Big)\Big), \\ =& -\frac{1}{2}\Big(m\log\gamma + \frac{1}{\gamma}\Big(\|\mathbf{x}-\mathbf{b}\|_2^2 + \mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{W}) \\ &+ \boldsymbol{\mu}^T\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{\mu} - (\mathbf{x}-\mathbf{b})^T\boldsymbol{W}\boldsymbol{\mu} - \boldsymbol{\mu}^T\boldsymbol{W}^T(\mathbf{x}-\mathbf{b})\Big)\Big), \\ =& -\frac{1}{2}\Big(m\log\gamma + \frac{1}{\gamma}\|\mathbf{x}-(\boldsymbol{W}\boldsymbol{\mu}+\mathbf{b})\|_2^2 \\ &+ \frac{1}{\gamma}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{W})\Big), \end{aligned} \tag{4.4}$$

where we obtained the results of the last two lines by recalling that $\mathbb{E}[\mathbf{z}^T\boldsymbol{A}\mathbf{z}^T] = \mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{A}) + \boldsymbol{\mu}^T\boldsymbol{A}\boldsymbol{\mu}$ and factorising. Replacing the regularisation and reconstruction terms of Equation 3.5 by Equations 4.1 and 4.4 and removing constants that cannot be optimised, we obtain the following ELBO for linear VAEs

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}) = -\Big(&m\log\gamma + \frac{1}{\gamma}\|\mathbf{x}-(\boldsymbol{W}\boldsymbol{\mu}+\mathbf{b})\|_2^2 + \frac{1}{\gamma}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{W}) \\ &+ \mathrm{Tr}(\boldsymbol{\Sigma}) + \|\boldsymbol{\mu}\|_2^2 - \log\det(\boldsymbol{\Sigma})\Big). \end{aligned} \tag{4.5}$$

Following Dai et al. (2017) and Lucas et al. (2019a), using Equation 4.5, we can now minimise $-\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$ w.r.t. $\boldsymbol{\Sigma}$,

$$\begin{aligned} -\frac{\partial \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})}{\partial \boldsymbol{\Sigma}} &= \frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n - \boldsymbol{\Sigma}^{-1} = 0, \\ \Leftrightarrow \boldsymbol{\Sigma} &= \left(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\right)^{-1}. \end{aligned} \tag{4.6}$$

Rearranging Equation 4.5 and replacing $\boldsymbol{\Sigma}$ with the result of Equation 4.6, the ELBO can further be simplified to:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = &- \Bigg(\log\left(\gamma^n \det(\boldsymbol{\Sigma})^{-1}\right) + \mathrm{Tr}\left(\boldsymbol{\Sigma}\Big(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\Big)\right) + \|\boldsymbol{\mu}\|_2^2 \\ &+ \frac{1}{\gamma}\|\mathbf{x} - (\boldsymbol{W}\boldsymbol{\mu} + \mathbf{b})\|_2^2 + (m-n)\log\gamma \Bigg), \\ = &- \Bigg(\log\left(\gamma^n \det\Big(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\Big)\right) + \|\boldsymbol{\mu}\|_2^2 + \frac{1}{\gamma}\|\mathbf{x} - (\boldsymbol{W}\boldsymbol{\mu} + \mathbf{b})\|_2^2 \\ &+ \mathrm{Tr}\left(\Big(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\Big)^{-1}\Big(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\Big)\right) + (m-n)\log\gamma \Bigg), \\ = &- \Big(\log\left(\det(\boldsymbol{W}^T\boldsymbol{W} + \gamma\boldsymbol{I}_n)\right) + \|\boldsymbol{\mu}\|_2^2 + \frac{1}{\gamma}\|\mathbf{x} - (\boldsymbol{W}\boldsymbol{\mu} + \mathbf{b})\|_2^2 + n \\ &+ (m-n)\log\gamma \Big), \\ = &- \Big(\log\left(\det(\boldsymbol{W}\boldsymbol{W}^T + \gamma\boldsymbol{I}_m)\right) + \|\boldsymbol{\mu}\|_2^2 + \frac{1}{\gamma}\|\mathbf{x} - (\boldsymbol{W}\boldsymbol{\mu} + \mathbf{b})\|_2^2 \Big), \end{aligned} \tag{4.7}$$

where the last line uses Sylvester's determinant theorem (Pozrikidis, 2014, p.271) to obtain $\det(\gamma\boldsymbol{I}_m + \boldsymbol{W}\boldsymbol{W}^T) = \gamma^{(m-n)}\det(\gamma\boldsymbol{I}_n + \boldsymbol{W}^T\boldsymbol{W})$ and constants are dropped. In the same way, minimising $-\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$ w.r.t. $\boldsymbol{\mu}$ gives,

$$\begin{aligned} -\frac{\partial \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})}{\partial \boldsymbol{\mu}} &= (\gamma\boldsymbol{I}_n + \boldsymbol{W}^T\boldsymbol{W})\boldsymbol{\mu} - \boldsymbol{W}^T(\mathbf{x} - \mathbf{b}) = 0, \\ \Leftrightarrow \boldsymbol{\mu} &= (\gamma\boldsymbol{I}_n + \boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T(\mathbf{x} - \mathbf{b}), \\ \Leftrightarrow \boldsymbol{\mu} &= \boldsymbol{W}^T(\gamma\boldsymbol{I}_m + \boldsymbol{W}\boldsymbol{W}^T)^{-1}(\mathbf{x} - \mathbf{b}), \end{aligned} \tag{4.8}$$

where the last line is obtained from the fact that $(\boldsymbol{I}+\boldsymbol{A}\boldsymbol{B})^{-1}\boldsymbol{A} = \boldsymbol{A}(\boldsymbol{I}+\boldsymbol{B}\boldsymbol{A})^{-1}$ (Petersen, Pedersen et al., 2008; Searle and Khuri, 2017).

Let us replace $\boldsymbol{\mu}$ with the result of Equation 4.8 in the following expression, using $\boldsymbol{A} = \boldsymbol{W}\boldsymbol{W}^T$ for readability

$$\begin{aligned}
\|\boldsymbol{\mu}\|_2^2 + \frac{1}{\gamma}\|\mathbf{x} - \boldsymbol{W}\boldsymbol{\mu} - \mathbf{b}\|_2^2 =& \boldsymbol{\mu}^T\boldsymbol{\mu} + \frac{1}{\gamma}\boldsymbol{\mu}^T\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{\mu} - \frac{1}{\gamma}\boldsymbol{\mu}^T\boldsymbol{W}^T(\mathbf{x} - \mathbf{b}) \\
& - \frac{1}{\gamma}(\mathbf{x} - \mathbf{b})^T\boldsymbol{W}\boldsymbol{\mu} + (\mathbf{x} - \mathbf{b})^T(\mathbf{x} - \mathbf{b}), \\
=& (\mathbf{x} - \mathbf{b})^T\Bigg(\frac{1}{\gamma}\boldsymbol{I}_m + (\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} \\
& + \frac{1}{\gamma}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\boldsymbol{A}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} \\
& - \frac{1}{\gamma}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\boldsymbol{A} - \frac{1}{\gamma}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\Bigg)(\mathbf{x} - \mathbf{b}), \\
=& (\mathbf{x} - \mathbf{b})^T\Bigg((\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} \\
& - \frac{1}{\gamma}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\boldsymbol{A} - \frac{1}{\gamma}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} + \frac{1}{\gamma}\boldsymbol{I}_m\Bigg) \\
& (\mathbf{x} - \mathbf{b}), \\
=& (\mathbf{x} - \mathbf{b})^T\Bigg(\frac{1}{\gamma}(\gamma\boldsymbol{I}_m + \boldsymbol{A})(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} \\
& - \frac{1}{\gamma}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\Bigg)(\mathbf{x} - \mathbf{b}), \\
=& (\mathbf{x} - \mathbf{b})^T\Bigg((\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} + \frac{1}{\gamma}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1} \\
& - \frac{1}{\gamma}\boldsymbol{A}(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}\Bigg)(\mathbf{x} - \mathbf{b}), \\
=& (\mathbf{x} - \mathbf{b})^T(\gamma\boldsymbol{I}_m + \boldsymbol{A})^{-1}(\mathbf{x} - \mathbf{b}). \qquad (4.9)
\end{aligned}$$

As in Dai and Wipf (2018, Appendix B), plugging Equation 4.9 back into Equation 4.7, we get

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) =& -\log\big(\det(\boldsymbol{W}\boldsymbol{W}^T + \gamma\boldsymbol{I}_m)\big) \\
& - (\mathbf{x} - \mathbf{b})^T(\boldsymbol{W}\boldsymbol{W}^T + \gamma\boldsymbol{I}_m)^{-1}(\mathbf{x} - \mathbf{b}), \qquad (4.10)
\end{aligned}$$

which corresponds to the learning objective of Probabilistic Principal Components Analysis (PPCA), where Σ and $\boldsymbol{\mu}$ have identical Maximum Likelihood Estimations (MLEs) (Tipping and Bishop, 1999). Moreover, we can see that Equation 4.10 is invariant to any orthogonal transformation $\boldsymbol{R}$ (e.g., permutation and rotation) as $\boldsymbol{W}\boldsymbol{R}\boldsymbol{R}^T\boldsymbol{W}^T = \boldsymbol{W}\boldsymbol{W}^T$. Thus, if $\boldsymbol{W}^*$ is a maximum, any $\boldsymbol{W}^*\boldsymbol{R}$ will also be a maximum.

4.2.2 *Impact of the diagonal covariance assumption*

So far, we have not used the diagonal covariance assumption where $\boldsymbol{\Sigma} \triangleq \text{diag}[\boldsymbol{\sigma}]$ in our derivations. Following Dai et al. (2017), Dai and Wipf (2018), and Rolinek, Zietlow and Martius (2019), we will now analyse the impact of a diagonal covariance on the analytical result obtained in Section 4.2.1.

Let us first modify Equation 4.6 such that the variance is enforced to be diagonal

$$\boldsymbol{\Sigma} = \left(\frac{1}{\gamma} \text{diag}\left[\text{diag}[\boldsymbol{W}^T\boldsymbol{W}]\right] + \boldsymbol{I}_n\right)^{-1}. \tag{4.11}$$

As all the equations seen in Section 4.2.1 apply, using the Singular Value Decomposition (SVD) of $\boldsymbol{W} \triangleq \boldsymbol{U\Lambda V}^T$, we can further simplify the log determinant term of Equation 4.10, such that for diagonal covariance,

$$\begin{aligned}
\mathcal{L}_{\text{diag}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) &= -\sum_{i=1}^{m} \log\left((\boldsymbol{WW}^T)_{ii} + \gamma\right) \\
&\quad - (\mathbf{x} - \mathbf{b})^T(\boldsymbol{U\Lambda V}^T\boldsymbol{V\Lambda U}^T + \gamma\boldsymbol{I}_m)^{-1}(\mathbf{x} - \mathbf{b}), \\
&= -\left(\sum_{i=1}^{a} \log\left(\frac{1}{\gamma}(\boldsymbol{WW}^T)_{ii} + 1\right) + m\log\gamma\right) \\
&\quad - (\mathbf{x} - \mathbf{b})^T(\boldsymbol{U\Lambda}^2\boldsymbol{U}^T + \gamma\boldsymbol{I}_m)^{-1}(\mathbf{x} - \mathbf{b}), \\
&= -\left(\sum_{i=1}^{a} \log\left(\frac{1}{\gamma}(\boldsymbol{WW}^T)_{ii} + 1\right) + m\log\gamma\right) - \boldsymbol{\Omega},
\end{aligned} \tag{4.12}$$

where a is the number of non-zero singular values and $\boldsymbol{\Omega} \triangleq (\mathbf{x} - \mathbf{b})^T(\boldsymbol{U\Lambda}^2\boldsymbol{U}^T + \gamma\boldsymbol{I}_m)^{-1}(\mathbf{x} - \mathbf{b})$. We know from Hadamard's inequality (Hadamard, 1893) that the determinant of a positive definite matrix is lower than or equal to the product of its diagonal values, with equality if and only if the matrix is diagonal. Thus, we have

$$\sum_{i=1}^{n} \log\left((\boldsymbol{WW}^T)_{ii} + \gamma\right) \geqslant \log\left(\det(\boldsymbol{WW}^T + \gamma\boldsymbol{I})\right). \tag{4.13}$$

Using the invariance to orthogonal transformation of Equation 4.10 and Sylvester's determinant theorem once again, we can choose $\boldsymbol{W}^* = \boldsymbol{WV}$ as a maximiser of Equation 4.10

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) &= -\log\left(\det(\boldsymbol{W}^*(\boldsymbol{W}^*)^T + \gamma\boldsymbol{I}_m)\right) - \boldsymbol{\Omega}, \\
&= -\left(\log\left(\det\left((\boldsymbol{WV})^T\boldsymbol{WV} + \gamma\boldsymbol{I}_n\right)\right) + (m-n)\log\gamma\right) - \boldsymbol{\Omega},
\end{aligned}$$

$$
\begin{aligned}
&= -\left(\log\left(\det\left(\boldsymbol{V}^T\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T\boldsymbol{V} + \gamma\boldsymbol{I}_n\right)\right) + (m-n)\log\gamma\right) \\
&\quad - \boldsymbol{\Omega}, \\
&= -\left(\log\left(\det(\boldsymbol{\Lambda}^2 + \gamma\boldsymbol{I}_n)\right) + (m-n)\log\gamma\right) - \boldsymbol{\Omega}, \\
&= -\left(\log\left(\det\left(\frac{1}{\gamma}\boldsymbol{\Lambda}^2 + \boldsymbol{I}_n\right)\right) + m\log\gamma\right) - \boldsymbol{\Omega}, \\
&= -\left(\sum_{i=1}^{a}\log\left(\frac{1}{\gamma}\boldsymbol{\Lambda}_{ii}^2 + 1\right) + m\log\gamma\right) - \boldsymbol{\Omega}. \qquad (4.14)
\end{aligned}
$$

Furthermore, $\boldsymbol{W}^*$ is also a maximiser of Equation 4.12 as we reach equality for Equation 4.13. Any signed permutation $\boldsymbol{P}$ of an optimal $\boldsymbol{W}^*$ will also be a maximiser of Equation 4.12. However, in opposition to Equation 4.10, any rotation of $\boldsymbol{W}^*$ is suboptimal for $\mathcal{L}_{\text{diag}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$, as we will get a lower bound due to Equation 4.13. To summarise, the SVD of any maximiser $\boldsymbol{W}$ of Equation 4.10 can be $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T$ for Equation 4.10, but is restricted to $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{P}^T$, where $\boldsymbol{P}$ is a signed permutation matrix, for Equation 4.12.

An important consequence is that when we assume diagonal covariance, $\boldsymbol{W}$ cannot have more than a non-zero columns, where $a \leqslant n$ corresponds to the number of non-zero singular values. Using Equation 4.11, it means that the diagonal covariance will be of the form:

$$
\begin{aligned}
\boldsymbol{\Sigma} &= \left[\boldsymbol{P}\,\text{diag}\left[\frac{\boldsymbol{\Lambda}_{11}^2}{\gamma} + 1, \cdots, \frac{\boldsymbol{\Lambda}_{aa}^2}{\gamma} + 1, 1, \cdots, 1\right]\boldsymbol{P}^T\right]^{-1}, \\
&= \boldsymbol{P}\,\text{diag}\left[\frac{1}{\frac{\boldsymbol{\Lambda}_{11}^2}{\gamma} + 1}, \cdots, \frac{1}{\frac{\boldsymbol{\Lambda}_{aa}^2}{\gamma} + 1}, 1, \cdots, 1\right]\boldsymbol{P}^T. \qquad (4.15)
\end{aligned}
$$

When $\gamma = 1$, we can see from Equation 4.15 that the more variance is explained by each $\boldsymbol{\Lambda}_{ii}$, the lower its corresponding variance index will be. On one hand, the more precise our decoder is, the closer to 0 γ will be, and the a values of $\boldsymbol{\Sigma}$ depending on γ will converge to 0. On the other hand, the less precise the decoder, the larger the value of γ, and when $\gamma \gg \boldsymbol{\Lambda}_{ii}$, the corresponding variance index will collapse to 1. To summarise, we have recovered the variance properties of active and passive variables stated in Definition 4.1.1 and observed that, for sufficiently high γ, a soft-threshold was operated on the variance indexes with the smallest singular values. For example, given $\boldsymbol{\Sigma} = \boldsymbol{P}\,\text{diag}\left[\frac{1}{\frac{\boldsymbol{\Lambda}_{11}^2}{\gamma}+1}, \frac{1}{\frac{\boldsymbol{\Lambda}_{22}^2}{\gamma}+1}\right]\boldsymbol{P}^T$ and $\boldsymbol{\Lambda}^2 = \text{diag}[4, 1]$, if we have a highly precise reconstruction with $\gamma = 0.1$, we will have $\boldsymbol{\Sigma} = \boldsymbol{P}\,\mathrm{diag}[0.02, 0.09]\boldsymbol{P}^T$, making both variables active. However, for a very large $\gamma = 100$, we will have $\boldsymbol{\Sigma} = \boldsymbol{P}\,\mathrm{diag}[0.96, 0.99]\boldsymbol{P}^T$, making both variables passive. We will see in Section 4.3 that this thresholding mechanism is critical to explain posterior collapse of β-VAEs (i.e., when all latent variables become passive).

Using Equation 4.8, let us now observe the behaviour of the mean representation. We have

$$\begin{aligned}
\boldsymbol{\mu} &= \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{U}^T(\gamma\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{\Lambda}^2\boldsymbol{U}^T)^{-1}(\mathbf{x}-\mathbf{b}),\\
&= \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{U}^T\boldsymbol{U}\,\mathrm{diag}\left[\frac{1}{\boldsymbol{\Lambda}_{11}^2+\gamma},\cdots,\frac{1}{\boldsymbol{\Lambda}_{aa}^2+\gamma},\frac{1}{\gamma},\cdots,\frac{1}{\gamma}\right]\boldsymbol{U}^T(\mathbf{x}-\mathbf{b}),\\
&= \boldsymbol{P}\,\mathrm{diag}\left[\frac{\boldsymbol{\Lambda}_{11}}{\boldsymbol{\Lambda}_{11}^2+\gamma},\cdots,\frac{\boldsymbol{\Lambda}_{aa}}{\boldsymbol{\Lambda}_{aa}^2+\gamma},0,\cdots,0\right]\boldsymbol{U}^T(\mathbf{x}-\mathbf{b}). \qquad (4.16)
\end{aligned}$$

We can directly see from Equation 4.16 that any passive variable will be set to 0 in the mean representation, consistent with Definition 4.1.1. As for the variance, a soft-thresholding mechanism will progressively collapse the mean indexes with small $\boldsymbol{\Lambda}_{ii}$ to 0 as γ increases because $\lim_{\gamma\to+\infty}\frac{\boldsymbol{\Lambda}_{ii}}{\boldsymbol{\Lambda}_{ii}^2+\gamma} = 0$. On the other hand, when γ is very small, we will retain all the singular values as $\lim_{\gamma\to 0}\frac{\boldsymbol{\Lambda}_{ii}}{\boldsymbol{\Lambda}_{ii}^2+\gamma} = \frac{1}{\boldsymbol{\Lambda}_{ii}}$.

In summary, by deriving an analytical solution for the "standard" VAE implementation, we have seen that the polarised regime naturally emerges from the combination of assumptions usually made about the diagonality of the posterior covariance, as well as the distribution of the prior, posterior, and likelihood. While we have restricted our analysis to the linear case, this phenomenon has also been demonstrated to generalise to non-linear models (Dai et al., 2017; Dai and Wipf, 2018; Rolinek, Zietlow and Martius, 2019; Zietlow, Rolinek and Martius, 2021).

4.3 MECHANISMS INFLUENCING THE POLARISED REGIME

We have seen in Section 4.2 that the polarised regime is a natural consequence of optimising the $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$ with the standard assumptions about the prior, posterior, and likelihood distributions (Dai and Wipf, 2018; Dai, Wang and Wipf, 2020). However, we will now see that specific forms of the ELBO given in Equation 3.5 encourage VAEs to encode a greater number of passive variables. First, let us consider VAEs through the lens of information bottleneck (Tishby, Pereira and Bialek, 2000; Tishby and Zaslavsky, 2015).

4.3.1 *Variational autoencoders and information bottleneck*

Equation 3.5 has been reformulated from the information theory point of view by Alemi et al. (2017, Appendix B), Alemi et al. (2018, Appendix D), and Burgess et al. (2018) as being a lower bound of the Information Bottleneck (IB) objective (Tishby, Pereira and Bialek, 2000) for unsupervised learning (Slonim et al., 2005). This objective aims to maximally compress the latent representation, while retaining sufficient mutual information between the input and the latent representation to reconstruct the input.

First, let us define $\boldsymbol{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{n}$, the set of observed variables sampled from the true data distribution $p(\mathbf{x})$, and $\boldsymbol{Z} = \{\mathbf{z}^{(i)}\}_{i=1}^{n}$, the corresponding set of latent variables. The IB can be expressed using the following equation, where we make the distinction between the generative mutual information I_d, which is based on $p_{\boldsymbol{\theta}}(\cdot)$, and the representational mutual information I_e, which is based on $q_{\boldsymbol{\phi}}(\cdot)$:

$$\mathrm{I}_d(\boldsymbol{X};\boldsymbol{Z}) - \beta \mathrm{I}_e(\boldsymbol{X};\boldsymbol{Z}). \tag{4.17}$$

Letting $p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x}) \approx p(\mathbf{x})q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, the generative mutual information can be expressed as:

$$\begin{aligned}
\mathrm{I}_d\left(\boldsymbol{X};\boldsymbol{Z}\right) &= \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}\left[\log\frac{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}{p(\mathbf{x})p_{\boldsymbol{\theta}}(\mathbf{z})}\right],\\
&= \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}\left[\log\frac{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\right] - \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}\left[\log p(\mathbf{x})\right],\\
&\approx \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big] + \mathrm{H}(\boldsymbol{X}),\\
&\approx \mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]\right],
\end{aligned} \tag{4.18}$$

where $\mathrm{H}(\cdot)$ denotes the entropy. Note that we drop $\mathrm{H}(\boldsymbol{X})$ in Equation 4.18 because the entropy of the data is out of our control and cannot be optimised.
Letting $q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x}) \approx p(\mathbf{x})q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, the representational mutual information can be expressed as:

$$\begin{aligned}
\mathrm{I}_e(\mathrm{X};\mathrm{Z}) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x})}\left[\log\frac{q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x})}{p(\mathbf{x})q_{\boldsymbol{\phi}}(\mathbf{z})}\right],\\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x})}\left[\log\frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{q_{\boldsymbol{\phi}}(\mathbf{z})}\right],\\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x})}\left[\log\frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})p_{\boldsymbol{\theta}}(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z})}\right],\\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x})}\left[\log\frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\right] - \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z},\mathbf{x})}\left[\log\frac{q_{\boldsymbol{\phi}}(\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\right],
\end{aligned}$$

$$\begin{aligned}
&\approx \mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\right]\right] - \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\right], \\
&= \mathbb{E}_{p(\mathbf{x})}\left[D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)\right] - D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big), && (4.19) \\
&\leqslant \mathbb{E}_{p(\mathbf{x})}\left[D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)\right]. && (4.20)
\end{aligned}$$

We will see in Section 4.3.3 that dropping $D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)$ in Equation 4.19 has important implications in the reconstruction quality.
Putting Equations 4.18 and 4.20 together, we have:

$$\mathrm{I}_d(\boldsymbol{X};\boldsymbol{Z}) - \beta \mathrm{I}_e(\boldsymbol{X};\boldsymbol{Z}) \geqslant \mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - \beta D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)\right].$$

Thus, Equation 3.5 is a lower bound of Equation 4.17, where $\beta = 1$ and the entropy of the data, which cannot be optimised, is dropped. We can further express this lower bound in terms of negative distortion (i.e., how well the input is reconstructed) and rate (i.e., the amount of information transmitted through $\mathbf{z}$).

$$\mathcal{L}_{\mathrm{IB}}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}) = \underbrace{\mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]\right]}_{\text{Negative distortion}} - \beta\,\underbrace{\mathbb{E}_{p(\mathbf{x})}\left[D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)\right]}_{\text{Rate}}. \tag{4.21}$$

Maximising Equation 4.21 can thus be thought of as limiting the distortion of the reconstructed output while restricting the amount of information that can be encoded. In other words, we aim to minimise the distortion of $\mathbf{x}$ and the rate of $\mathbf{z}$. The β term can thus regulate this capacity, and—as suggested by Burgess et al. (2018)—the greater the value of β, the more efficient the encoding must be. Indeed, higher β will decrease the maximum number of active variables that a VAE can learn, as we will see in Section 4.3.2.

As shown by Alemi et al. (2018), different models can result in the same ELBO value while having different rate and distortion, depending on their computational capacity. As shown in Figure 4.1, one can obtain the same score given a model with low rate but high distortion (see dashed orange curve), a model with high rate but low distortion (see dotted green curve), or a model with similar rate and distortion (see plain blue curve). Different models will thus allow us to navigate through different RD curves. However, changing β will allow us to navigate through the same RD curve.

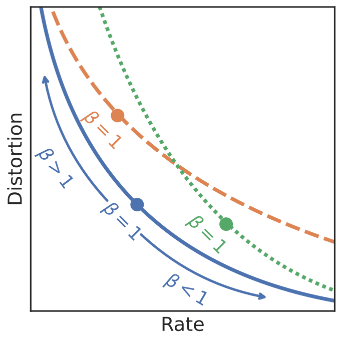


Figure 4.1: Schematic representation of the RD plane. Different models result in different curves, and different β values allow us to navigate on a given curve. The plain blue curve represents a model with similar rate and distortion scores, the dashed orange curve a model with low rate and high distortion, and the dotted green curve a model with high distortion and low rate. This figure is inspired by the great illustration proposed in Alemi et al. (2018, Figure 1).

4.3.2 *How does the rate influence the polarised regime?*

Let us consider $\mathcal{L}_{\text{IB}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$ from Equation 4.21. It can be reformulated as:

$$\mathcal{L}_{\text{IB}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \beta\bigg(\mathbb{E}_{p(\mathbf{x})}\Big[\frac{1}{\beta}\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]\Big] - \mathbb{E}_{p(\mathbf{x})}\Big[D_{\text{KL}}\big(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big)\Big]\bigg).$$

Using the same factorisation on the analytical solution of linear VAEs obtained in Equation 4.5, we can directly see that β is tightly linked with the variance of the decoder, γ.

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = -\beta\Big(&\frac{m}{\beta}\log\gamma + \frac{1}{\gamma\beta}\|\mathbf{x} - (\boldsymbol{W}\boldsymbol{\mu} + \mathbf{b})\|_2^2 + \frac{1}{\gamma\beta}\operatorname{Tr}(\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{W}) \\ &+ \operatorname{Tr}(\boldsymbol{\Sigma}) + \|\boldsymbol{\mu}\|_2^2 - \log\det(\boldsymbol{\Sigma})\Big). \end{aligned} \tag{4.22}$$

This is especially apparent when we derive the re-weighted equations of the mean and variance. Using Equation 4.22, the variance from Equations 4.6 and 4.15 becomes

$$\begin{aligned}\boldsymbol{\Sigma} &= \Big(\frac{1}{\beta\gamma}\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\Big)^{-1}, \\ &= \boldsymbol{P}\operatorname{diag}\left[\frac{1}{\frac{\boldsymbol{\Lambda}_{11}^2}{\beta\gamma} + 1}, \cdots, \frac{1}{\frac{\boldsymbol{\Lambda}_{aa}^2}{\beta\gamma} + 1}, 1, \cdots, 1\right]\boldsymbol{P}^T. \end{aligned} \tag{4.23}$$

On one hand, $\beta > 1$ will result in a more aggressive thresholding mechanism, pruning more dimensions away (i.e., allowing less active variables). This is consistent with the information theory point of view that increasing β decreases the rate, and, as a result, the channel capacity of the latent representation. On the other hand, $\beta < 1$ will soften the thresholding behaviour and keep active variables with lower singular values. To summarise, higher β leads to more passive variables in the variance representation, while lower β allows for more active variables. We also observe this phenomenon on the mean from Equations 4.8 and 4.16 once updated with the β term

$$\begin{aligned}\boldsymbol{\mu} &= \boldsymbol{W}^T(\beta\gamma\boldsymbol{I}_m + \boldsymbol{W}\boldsymbol{W}^T)^{-1}(\mathbf{x}-\mathbf{b}),\\ &= \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{U}^T(\beta\gamma\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{\Lambda}^2\boldsymbol{U}^T)^{-1}(\mathbf{x}-\mathbf{b}),\\ &= \boldsymbol{P}\,\mathrm{diag}\left[\frac{\boldsymbol{\Lambda}_{11}}{\boldsymbol{\Lambda}_{11}^2+\beta\gamma},\cdots,\frac{\boldsymbol{\Lambda}_{aa}}{\boldsymbol{\Lambda}_{aa}^2+\beta\gamma},0,\cdots,0\right]\boldsymbol{U}^T(\mathbf{x}-\mathbf{b}).\end{aligned} \tag{4.24}$$

As for the variance, β will modulate the agressiveness of the thresholding with higher $\beta > 1$ resulting in more passive variables and $\beta < 1$ allowing for more active variables. To summarise, the β term introduced in $\mathcal{L}_{\text{IB}}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x})$, which modulates the rate, is tightly linked with the polarised regime. Lower rate results in a more agressive thresholding behaviour and more passive variables, while a higher rate allows us to encode more active variables. Looking at Equations 4.23 and 4.24, we can also clearly see that β values that are too high will push the model towards posterior collapse, with latent representations composed only of passive variables.

4.3.3 *Moving through the rate-distortion curve with the β-VAE family*

We will now present the β-VAE family: a family of methods proposed to allow an efficient modulation of the RD tradeoff seen in Equation 4.21. As we have seen above, selecting different β values in such models allow us to change the maximum number of active variables that can be encoded in $\mathbf{z}$ and will thus serve as reference models throughout our study of the polarised regime in Part II.

β-VAE Higgins et al. (2017) introduced a learning objective similar to Equation 4.21 whose goal was to bias the RD tradeoff by penalising the regularisation (i.e., rate) term more strongly. Specifically, they focused on use cases where $\beta > 1$. This is formulated as the following learning objective:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\big] - \beta D_{\text{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big). \tag{4.25}$$

One of the downsides of penalising the encoding more strongly is that the reconstruction is of lower quality.

Annealed VAE Burgess et al. (2018) provided an analysis of β-VAE disentangled representations through the lens of information theory, based on the learning objective described by Alemi et al. (2017). They argued that because β-VAEs increase the pressure on the channel capacity of the network (i.e., the maximum rate at which the information can be transmitted through $\mathbf{z}$), the optimal way to encode information would be on separate dimensions, leading to disentanglement. They hypothesised that β-VAEs will learn the latent variables having the most impact on the reconstruction first, then gradually optimise less critical variables. To ease the learning of these less important latent variables, they propose gradually increasing the encoding capacity during the training process, relaxing the initial constraint. This leads to the following objective, where $\boldsymbol{C}$ is a parameter that can be understood as a channel capacity and α is a hyper-parameter penalising the divergence (similar to β in β-VAE):

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - \alpha \left| D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\right) - \boldsymbol{C} \right| . \tag{4.26}$$

As the training progresses, the channel capacity $\boldsymbol{C}$ is increased, going from zero to its maximum channel capacity $\boldsymbol{C}_{max}$. For example, given a maximum channel capacity of 100, during the first training step, any deviation from the KLD will be penalised similarly to β-VAE because $\boldsymbol{C} = 0$. After n steps, once the channel capacity is annealed to its maximum value, the KLD will be penalised only when it is higher than 100.

Factor VAE In Factor VAE (Kim and Mnih, 2018), the approach is slightly different from the β-VAE objective presented in Equation 4.25. Instead of using the lower bound Equation 4.20, Kim and Mnih (2018) tried to approximate the exact value of $\mathrm{I}_e(\mathrm{X};\mathrm{Z})$ using Equation 4.19 to provide better reconstruction quality. Indeed, the authors argued that to prevent blurry output, only the distance between the estimated latent factors and the prior should be penalised, and they proposed a new objective to this end:

$$\mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\right)\right] - \alpha D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\right). \tag{4.27}$$

Here, $D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}) \parallel p_\theta(\mathbf{z})\big)$ is approximated by penalising the dependencies between the dimensions of $q_\phi(\mathbf{z})$:

$$\frac{1}{n}\sum_{i=1}^{n}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right] - D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) \parallel p_\theta(\mathbf{z})\big)\right] \\ - \alpha \underbrace{D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}) \,\middle\|\, \prod_{j=1}^{\mathrm{d}} q_\phi(\mathbf{z}_j)\right)}_{\text{total correlation}}. \tag{4.28}$$

As the computation of the total correlation defined in Equation 4.28 is intractable, it is estimated by sampling a batch from $q_\phi(\mathbf{z})$ and shuffling the values of each dimension of the latent variables to obtain the samples for $\prod_{j=1}^{\mathrm{d}} q_\phi(\mathbf{z}_j)$. A binary classifier is then trained to recognise the samples belonging to $q_\phi(\mathbf{z})$, and the density ratio is computed using the probability $p_{classif}(\mathbf{z})$ given by the classifier that the samples belong to $q_\phi(\mathbf{z})$:

$$D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}) \,\middle\|\, \prod_{j=1}^{\mathrm{d}} q_\phi(\mathbf{z}_j)\right) \approx \mathbb{E}_{q_\phi(\mathbf{z})}\left[\log \frac{p_{classif}(\mathbf{z})}{1 - p_{classif}(\mathbf{z})}\right].$$

β-TC VAE As Kim and Mnih (2018), Chen et al. (2018) proposed to optimise Equation 4.28. The main difference is that Chen et al. (2018) approximated the total correlation using mini-batch weighted sampling. Here, the estimation is computed over a mini-batch of samples $\{\mathbf{x}^{(i)}\}_{i=1}^{m}$ as follows:

$$\mathbb{E}_{q_\phi(\mathbf{z})}[\log q_\phi(\mathbf{z})] \approx \frac{1}{m}\sum_{i=1}^{m}\left(\log \frac{1}{nm}\sum_{k=1}^{m} q_\phi(\mathbf{z}^{(i)}|\mathbf{x}^{(k)})\right), \tag{4.29}$$

where m is the number of samples in the mini-batch and n the total number of input examples. $\mathbb{E}_{q_\phi(\mathbf{z}_j)}[\log q_\phi(\mathbf{z}_j)]$ can be computed in a similar way. We refer the reader to Chen et al. (2018, Appendix C.1) for the detailed derivation of Equation 4.29.

DIP-VAE As Kim and Mnih (2018) and Chen et al. (2018), Kumar, Sattigeri and Balakrishnan (2018) proposed to regularise the distance between $q_\phi(\mathbf{z})$ and $p(\mathbf{z})$ using Equation 4.27. The main difference here is that $D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}) \parallel p(\mathbf{z})\big)$ is measured by matching the moments of the learned distribution $q_\phi(\mathbf{z})$ and its prior $p(\mathbf{z})$. The second moment of the learned distribution is given by:

$$\mathrm{Cov}_{q_\phi(\mathbf{z})}[\mathbf{z}] = \mathrm{Cov}_{p(\mathbf{x})}\left[\boldsymbol{\mu}\right] + \mathbb{E}_{p(\mathbf{x})}\left[\boldsymbol{\sigma}\right]. \tag{4.30}$$

Two divergences are then defined. The first, DIP-VAE I, penalises only the first term of Equation 4.30:

$$\lambda D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big) = \lambda_{od} \sum_{i \neq j} \left(\mathrm{Cov}_{p(\mathbf{x})}\left[\boldsymbol{\mu}\right]\right)_{ij}^{2} + \lambda_{d} \sum_{i} \left(\mathrm{Cov}_{p(\mathbf{x})}\left[\boldsymbol{\mu}\right]_{ii} - 1\right)^{2},$$

where λ_d and λ_{od} are the diagonal and off-diagonal regularisation terms, respectively. The second, DIP-VAE II, penalises both terms of Equation 4.30:

$$\lambda D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big) = \lambda_{od} \sum_{i \neq j} \left(\mathrm{Cov}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\mathbf{z}\right]\right)_{ij}^{2} + \lambda_{d} \sum_{i} \left(\mathrm{Cov}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\mathbf{z}\right]_{ii} - 1\right)^{2}.$$

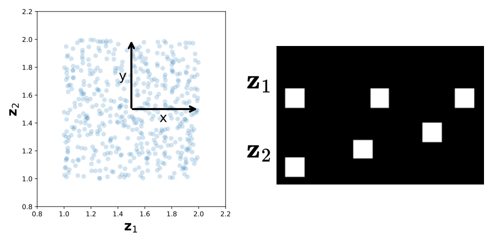
In practical terms, it means that DIP-VAE I enforces the covariance matrix of the mean representation to be diagonal, while DIP-VAE II explicitly regularises the covariance matrix of the sampled representation. While the difference may seem minor, we will show in Chapter 6 that only enforcing diagonal covariance of the sampled representation can result in discrepancies between the mean and sampled representations which are not observed in DIP-VAE I (Locatello et al., 2019a).

4.4 RELATIONSHIP BETWEEN POLARISED REGIME AND DISENTANGLEMENT

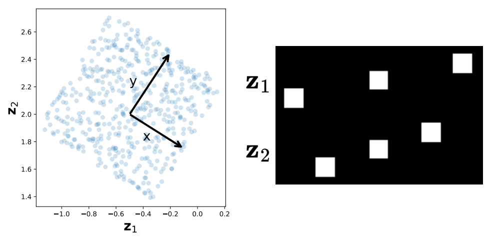
The family of methods presented in Section 4.3.3 efficiently enforce lower rate, or equivalently from the polarised regime point of view, more passive variables. However, the original goal the β-VAE family of methods was not the RD tradeoff. They aimed to provide disentangled representations; that is, latent representations where one variable will only encode information about one generative factor, as illustrated in Figure 4.2. Disentangled latent representations are semantically meaningful, fair, and beneficial for abstract reasoning (Locatello et al., 2019a; van Steenkiste et al., 2019), which makes them attractive for downstream tasks applications. In this section we will explore the relationship between the polarised regime and disentangled representation learning.

4.4.1 *Unsupervised disentanglement is impossible without further inductive bias*

We have shown in Section 4.2.1 that the ELBO of linear VAEs from Equation 4.10 was invariant to rotations when the covariance of the posterior was not enforced to be diagonal, making VAEs unidentifiable. In other words, entangled and disentangled latent representations such as those illustrated in Equation 4.2 could provide equally



(a) Disentangled representations



(b) Entangled representations

Figure 4.2: Examples of disentangled and entangled representations. In (a) the latent representations are disentangled: $\mathbf{z}_1$ and $\mathbf{z}_2$ are aligned with the generative factors (x and y position). In (b) the latent representations are entangled: $\mathbf{z}_1$ and $\mathbf{z}_2$ are not aligned with the generative factors.

good reconstructed images, with no mechanism in place to favour one over the other. Because of this unidentifiability, even with the remaining distributional assumptions about the prior and likelihood, we cannot hope to recover disentangled representations. This observation was the basis of multiple demonstrations (Locatello et al., 2019a; Dai and Wipf, 2018; Rolinek, Zietlow and Martius, 2019) and is also referred to as the *impossibility result*.

When the variance is enforced to be diagonal, VAEs become sensitive to rotations of the latent space. They develop a thresholding behaviour (the polarised regime) which is akin to a local PCA. While this encourages active variables to be orthogonal to each others, it still cannot guarantee that this is aligned with the generative factors, and thus semantically meaningful. Indeed, we have seen in Equations 4.24 and 4.23 that active variables are selected according to their singular values. A thresholding mechanism prunes variables with low singular values and is made more aggressive by higher β. It means that VAEs encode active variables that explain the most variance in the latent space (Zietlow, Rolinek and Martius, 2021). Intuitively, if a generative factor is not linked with high local variance, it will not be encoded at all and multiple generative factors may be conflated in one variable.

Overall, the "standard" implementation of VAEs is sufficient to obtain uncorrelated active variables but cannot guarantee disentanglement. Indeed, VAEs have a PCA-like behaviour instead of an Independent Components Analysis (ICA)-like behaviour.

4.4.2 *Towards identifiable VAEs*

Despite the unidentifiability issues previously discussed, models from the β-VAE family display empirical evidence of disentangled latent representations (Higgins et al., 2017; Burgess et al., 2018; Kim and Mnih, 2018; Chen et al., 2018; Kumar, Sattigeri and Balakrishnan, 2018). Zietlow, Rolinek and Martius (2021) empirically demonstrated that in most popular datasets used in the disentanglement literature the directions of largest variance in the latent space were aligned with the generative factors. In this context, the polarised regime of VAEs will thus result in disentangled latent representations, as it will recover the semantically meaningful generative factors. However, one can wonder whether this alignment between the variance of the latent space and the generative factors is likely to be systematically encountered in other datasets. Interestingly, Gresele et al. (2021) recently conjectured that such directions of variance in the latent space were likely to correspond to generative factors, and Reizinger et al. (2022) demonstrated that under some assumptions this conjecture could be applied to VAEs learning in a polarised regime.

While the ongoing work on identifiability in the unsupervised setting is very promising, the conjecture of Gresele et al. (2021) has yet to be proved, and the theoretical analysis of Reizinger et al. (2022) required the dimensionality of the latent space to be equal to the dimensionality of the output space, which is unrealistic. In the meantime, Khemakhem et al. (2020) and Mita, Filippone and Michiardi (2021) proposed provably identifiable VAEs for the semi-supervised and self-supervised settings.

iVAEs (Khemakhem et al., 2020) rely on a conditionally factorial prior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{u})$ from the exponential family where $\mathbf{u}$ is observed. In practice, the conditional prior is chosen to be Gaussian with diagonal covariance as the prior must factorise. These models maximise

$$\mathcal{L}_{\text{iVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - D_{\text{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, \mathbf{u}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{u})\big). \quad (4.31)$$

We can see in Equation 4.31 that the prior departs from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, which changes the derivation of the analytical results of the KLD in Section 4.2. Thus, while we have identifiable representations, one can wonder whether we can still benefit from the sparsity induced by the polarised regime. We will explore this question in Chapter 5 and show that iVAEs still benefit from the polarised regime, although their passive variables have different variance values.

Part II

EXTENSIONS AND APPLICATIONS OF THE POLARISED REGIME

5

GENERALISATION OF THE POLARISED REGIME TO OTHER TYPES OF VAES

5.1 INTRODUCTION

As we have seen in Section 4.4, iVAEs (Khemakhem et al., 2020; Mita, Filippone and Michiardi, 2021) are models which provide transparent embeddings in exchange for some degree of supervision. In opposition to standard VAEs, whose prior is fixed to $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, the prior of iVAEs is learned and conditioned on additional observed variables. Given the increased transparency of the learned representations, iVAEs are attractive models for downstream tasks applications. For example, using the learned embeddings for classification with a logistic regression would allow the user to more easily interpret the variables that are important (or not) for the given task. Indeed, each weight would correspond to a semantically meaningful feature. As a result, one could analyse and explain the decisions made by the classifier while benefiting from deep embeddings.

We have seen in Section 4.2 that standard VAEs, while being non-identifiable, benefit from a sparsity-inducing mechanism called the polarised regime, which encourages the compression of latent representation. Specifically, once the model cannot improve the reconstruction by learning additional active variables, the remaining variables are not used by the decoder and collapse to the prior to lower the KLD. This pruning capacity is very useful as we generally do not know how many latent variables should be used for a specific dataset. Thus, in standard VAEs, if the number of latent variables is overestimated, the superfluous variables will just be made passive by the model.

However, the polarised regime has only been studied in standard VAEs so far, and the existing literature (Dai et al., 2017; Dai and Wipf, 2018; Dai, Wang and Wipf, 2020; Rolinek, Zietlow and Martius, 2019; Zietlow, Rolinek and Martius, 2021) relies on the prior being $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ to derive proofs of these behaviour. One can thus wonder: can iVAEs have the best of both worlds and provide transparent embeddings while still benefiting from the sparsity-inducing property of the polarised regime? Answering this question is especially important for downstream task applications where the multicollinearity of redundant latent variables or the resulting highly dimensional embeddings could hinder downstream task performances and interpretability.

In this Chapter, we will prove that linear iVAEs (Khemakhem et al., 2020) with a fixed prior mean of 0 do learn in a polarised regime and provide further empirical evidence that this phenomenon also holds when the prior mean is learned and when the model is not linear.

5.2 THE POLARISED REGIME OF LINEAR IVAES

As in Section 4.2, we will first derive an analytical solution for the linear case of iVAEs to gain some insights into whether these models learn in a polarised regime. In this section, we will make the following assumptions about our model: $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{u}) \sim \mathcal{N}(\mu^\dagger, \boldsymbol{\Sigma}^\dagger)$, $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, \mathbf{u}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \sim \mathcal{N}(\boldsymbol{W}\mathbf{z} + \mathbf{b}, \gamma \boldsymbol{I}_m)$. As Khemakhem et al. (2020) and Mita, Filippone and Michiardi (2021), we further assume that $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^\dagger$ are diagonal. Using Equation 2.18, we can replace the KLD from the learning objective of iVAEs in Equation 4.31 with

$$D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, \mathbf{u}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{u})\big) = \frac{1}{2}\Bigg(\mathrm{Tr}\left((\boldsymbol{\Sigma}^\dagger)^{-1}\boldsymbol{\Sigma}\right) + (\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})^T (\boldsymbol{\Sigma}^\dagger)^{-1} (\boldsymbol{\mu}^\dagger - \boldsymbol{\mu}) + \log \frac{\det(\boldsymbol{\Sigma}^\dagger)}{\det(\boldsymbol{\Sigma})} - n \Bigg). \tag{5.1}$$

The reconstruction term of iVAEs is similar to VAEs, thus, for the linear case, we have

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},\mathbf{u})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] = -\frac{1}{2}\Big(m \log\gamma + \frac{1}{\gamma}\|\mathbf{x} - \boldsymbol{W}\boldsymbol{\mu} - \mathbf{b}\|_2^2 + \frac{1}{\gamma}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{W}) \Big). \tag{5.2}$$

Overall, combining Equation 5.1 and Equation 5.2, the learning objective of linear iVAEs, denoted liVAE thereafter, is

$$\begin{aligned}
\mathcal{L}_{\mathrm{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = -\frac{1}{2}\Big(& m \log\gamma + \frac{1}{\gamma}\|\mathbf{x} - \boldsymbol{W}\boldsymbol{\mu} - \mathbf{b}\|_2^2 + \frac{1}{\gamma}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{W}) \\
& + \mathrm{Tr}\left((\boldsymbol{\Sigma}^\dagger)^{-1}\boldsymbol{\Sigma}\right) + (\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})^T (\boldsymbol{\Sigma}^\dagger)^{-1} (\boldsymbol{\mu}^\dagger - \boldsymbol{\mu}) \\
& + \log\det(\boldsymbol{\Sigma}^\dagger) - \log\det(\boldsymbol{\Sigma}) - n \Big), \\
= -\frac{1}{2}\Bigg(& m \log\gamma + \frac{1}{\gamma}\|\mathbf{x} - \boldsymbol{W}\boldsymbol{\mu} - \mathbf{b}\|_2^2
\end{aligned}$$

$$+ \operatorname{Tr}\left(\boldsymbol{\Sigma}\Big(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + (\boldsymbol{\Sigma}^\dagger)^{-1}\Big)\right)$$
$$+ (\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})^T(\boldsymbol{\Sigma}^\dagger)^{-1}(\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})$$
$$\left. + \log\det(\boldsymbol{\Sigma}^\dagger) - \log\det(\boldsymbol{\Sigma}) - n\right), \tag{5.3}$$

and we retrieve the learning objective of classical linear VAEs obtained in Equation 4.5 when fixing $\boldsymbol{\Sigma}^\dagger = \boldsymbol{I}_n$ and $\boldsymbol{\mu}^\dagger = \mathbf{0}$, as expected.

Khemakhem et al. (2020, Thm. 2, Thm. 3) show that for VAEs to be identifiable when the conditional prior is Gaussian, one can learn both the variance and the mean of the prior or only the variance. However, models where only the prior's mean is learned are not identifiable. Given Equation 5.3, we will now analyse whether the polarised regime hold these two configurations.

Specifically, in Section 5.2.1 we will consider the case where both $\boldsymbol{\Sigma}^\dagger$ and $\boldsymbol{\mu}^\dagger$ are learned, perform a MLE of $\boldsymbol{\mu}$, Σ, $\boldsymbol{\mu}^\dagger$, and $\Sigma^\dagger$, and assess whether given these results we can conclude that iVAEs in this configuration learn in a polarised regime. Then, in Section 5.2.2, we will follow the same process to analyse the case where $\boldsymbol{\Sigma}^\dagger$ is learned and $\boldsymbol{\mu}^\dagger$ is fixed to $\mathbf{0}$, which mirrors the implementation proposed in the experimental section of Khemakhem et al. (2020).

5.2.1 *Do iVAEs whose prior mean and variance are learned behave in a polarised regime?*

As our objective is to investigate whether iVAEs whose prior mean and variance are learned behave in a polarised regime, we will start by deriving the MLEs of $\boldsymbol{\mu}^\dagger$ and $\boldsymbol{\Sigma}^\dagger$. From Equation 5.3, we have

$$-\frac{\partial \mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u})}{\partial \boldsymbol{\mu}^\dagger} = 2(\boldsymbol{\Sigma}^\dagger)^{-1}(\boldsymbol{\mu}^\dagger - \boldsymbol{\mu}) = 0,$$
$$\Leftrightarrow \boldsymbol{\mu}^\dagger = \boldsymbol{\mu}. \tag{5.4}$$

Following the same process for $\boldsymbol{\Sigma}^\dagger$, we get

$$-\frac{\partial \mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u})}{\partial \boldsymbol{\Sigma}^\dagger} = -(\boldsymbol{\Sigma}^\dagger)^{-1}\boldsymbol{\Sigma}(\boldsymbol{\Sigma}^\dagger)^{-1} - (\boldsymbol{\Sigma}^\dagger)^{-1}(\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})(\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})^T(\boldsymbol{\Sigma}^\dagger)^{-1} + (\boldsymbol{\Sigma}^\dagger)^{-1}.$$

So,

$$\boldsymbol{\Sigma}^\dagger = \boldsymbol{\Sigma} + (\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})(\boldsymbol{\mu}^\dagger - \boldsymbol{\mu})^T. \tag{5.5}$$

Thus, around the optimal $\boldsymbol{\mu}^\dagger$, we have

$$\boldsymbol{\Sigma}^\dagger = \boldsymbol{\Sigma}. \tag{5.6}$$

Now, let us consider the MLE of the posterior's variance $\boldsymbol{\Sigma}$. From Equation 5.3, we have

$$\begin{aligned} -\frac{\partial \mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u})}{\partial \boldsymbol{\Sigma}} &= \frac{1}{\gamma}\boldsymbol{W}\boldsymbol{W}^T + (\boldsymbol{\Sigma}^\dagger)^{-1} - \boldsymbol{\Sigma}^{-1} = 0, \\ \Leftrightarrow \boldsymbol{\Sigma} &= \left(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + (\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}. \end{aligned} \tag{5.7}$$

We can already see that the problem is ill-posed as we get two different solutions for $\boldsymbol{\Sigma}$ from Equations 5.6 and 5.7.

Using Equation 5.6 and recalling that $\boldsymbol{\Sigma}$ is diagonal, it means that when $\boldsymbol{\Sigma}^\dagger$ is optimal,

$$\boldsymbol{\Sigma} = \text{diag}\left[\frac{1}{\frac{\boldsymbol{A}_{11}}{\gamma} + \frac{1}{\boldsymbol{\Sigma}_{11}}}, \cdots, \frac{1}{\frac{\boldsymbol{A}_{nn}}{\gamma} + \frac{1}{\boldsymbol{\Sigma}_{nn}}}\right], \tag{5.8}$$

where $\boldsymbol{A}_{ii} \triangleq (\boldsymbol{W}^T\boldsymbol{W})_{ii}$.
Given that

$$-\frac{\partial \mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u})}{\partial \boldsymbol{\mu}} = -\frac{2}{\gamma}\boldsymbol{W}^T(\mathbf{x} - \mathbf{b}) + \frac{2}{\gamma}\boldsymbol{W}^T\boldsymbol{W}\boldsymbol{\mu} - 2(\boldsymbol{\Sigma}^\dagger)^{-1}(\boldsymbol{\mu}^\dagger - \boldsymbol{\mu}).$$

The MLE of $\boldsymbol{\mu}$ is,

$$\begin{aligned} \boldsymbol{\mu} =& \frac{1}{\gamma}\left(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + (\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}\boldsymbol{W}^T(\mathbf{x} - \mathbf{b}) \\ &+ \left(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + (\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}(\boldsymbol{\Sigma}^\dagger)^{-1}\boldsymbol{\mu}^\dagger, \\ =& \left(\boldsymbol{W}^T\boldsymbol{W} + \gamma(\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}\boldsymbol{W}^T(\mathbf{x} - \mathbf{b}) + \left(\frac{1}{\gamma}\boldsymbol{\Sigma}^\dagger\boldsymbol{W}^T\boldsymbol{W} + \boldsymbol{I}_n\right)^{-1}\boldsymbol{\mu}^\dagger. \end{aligned} \tag{5.9}$$

When $\boldsymbol{\mu}^\dagger$, $\boldsymbol{\Sigma}^\dagger$, and $\boldsymbol{\Sigma}$ are optimal, it means that

$$\boldsymbol{\mu} = (\boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T(\mathbf{x} - \mathbf{b}). \tag{5.10}$$

Based on Equations 5.8 and 5.10, we can show that

Theorem 5.2.1. *An iVAE with a highly precise reconstruction cannot simultaneously maximise the likelihood of $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\mu}^\dagger$, and $\boldsymbol{\Sigma}^\dagger$.*

Proof. We know from Equation 5.8 that given $\boldsymbol{A}_{ii} = (\boldsymbol{W}^T\boldsymbol{W})_{ii}$, we have $\frac{1}{\boldsymbol{\Sigma}_{ii}} = \frac{\boldsymbol{A}_{ii}}{\gamma} + \frac{1}{\boldsymbol{\Sigma}_{ii}}\ \forall i \in \{1, \cdots, n\}$. Thus, we must have $\frac{\boldsymbol{A}_{ii}}{\gamma} = 0\ \forall i \in \{1, \cdots, n\}$ which is only possible if $\boldsymbol{A} = \mathbf{0}$ or $\gamma \to +\infty$. If $\boldsymbol{A} = \mathbf{0}$, then $\boldsymbol{W}^T\boldsymbol{W}$ is not invertible and the likelihood of $\boldsymbol{\mu}$ cannot be maximal. Thus, the only way to fulfill Equations 5.10 and 5.8 is to have $\gamma \to +\infty$, which contradicts the fact that the reconstruction of the model is highly precise. $\square$

From Theorem 5.2.1, we can see that there is no solution for which the likelihood of all the parameters (i.e., $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\mu}^\dagger$, and $\boldsymbol{\Sigma}^\dagger$) can be maximised simultaneously. Indeed, if all the parameters take the values obtained from MLE, the model cannot reach a good reconstruction. While extending this analysis is left for future work, in Section 5.3 we will provide empirical evidence that iVAE still learn in a polarised regime when $\boldsymbol{\mu}^\dagger$ and $\boldsymbol{\Sigma}^\dagger$ are learned.

5.2.2 *Do iVAEs whose prior mean is fixed and variance is learned behave in a polarised regime?*

In this section, we are interested in the configuration used in the original implementation of Khemakhem et al. (2020), where $\boldsymbol{\mu}^\dagger = \mathbf{0}$ and $\boldsymbol{\Sigma}^\dagger$ is learned. That is, when the learning objective from Equation 5.3 can be reduced to

$$\begin{aligned}\mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = -\frac{1}{2}\Bigg(& m \log \gamma + \frac{1}{\gamma}\|\mathbf{x} - \boldsymbol{W}\boldsymbol{\mu} - \mathbf{b}\|_2^2 \\ & + \operatorname{Tr}\left(\boldsymbol{\Sigma}\left(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + (\boldsymbol{\Sigma}^\dagger)^{-1}\right)\right) \\ & + \boldsymbol{\mu}^T(\boldsymbol{\Sigma}^\dagger)^{-1}\boldsymbol{\mu} + \log\det(\boldsymbol{\Sigma}^\dagger) - \log\det(\boldsymbol{\Sigma}) - n\Bigg). \end{aligned} \tag{5.11}$$

We can easily obtain the updated MLE of $\boldsymbol{\Sigma}$ by replacing $\boldsymbol{\mu}^\dagger$ with $\mathbf{0}$ in Equation 5.5 and recalling that $\boldsymbol{\Sigma}^\dagger$ is diagonal,

$$\begin{aligned}\boldsymbol{\Sigma}^\dagger &= \boldsymbol{\Sigma} + \boldsymbol{\mu}\boldsymbol{\mu}^T, \\ \boldsymbol{\Sigma}^\dagger &\approx \operatorname{diag}\left[\boldsymbol{\Sigma}_{11} + \boldsymbol{\mu}_{11}^2, \cdots, \boldsymbol{\Sigma}_{nn} + \boldsymbol{\mu}_{nn}^2\right].\end{aligned} \tag{5.12}$$

Using Equation 5.8, the updated MLE of $\boldsymbol{\Sigma}$ around the optimal solution of $\boldsymbol{\Sigma}^\dagger$ is thus,

$$\boldsymbol{\Sigma} = \left(\frac{1}{\gamma}\boldsymbol{W}^T\boldsymbol{W} + (\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1},$$

$$= \operatorname{diag}\left[\frac{1}{\frac{\boldsymbol{A}_{11}}{\gamma}+\frac{1}{\boldsymbol{\Sigma}_{11}+\boldsymbol{\mu}_{11}^2}}, \cdots, \frac{1}{\frac{\boldsymbol{A}_{nn}}{\gamma}+\frac{1}{\boldsymbol{\Sigma}_{nn}+\boldsymbol{\mu}_{nn}^2}}\right], \tag{5.13}$$

where $\boldsymbol{A}_{ii} \triangleq (\boldsymbol{W}^T\boldsymbol{W})_{ii}$ as before.

In the same way, from Equation 5.9 we obtain

$$\begin{aligned}
\boldsymbol{\mu} &= \left(\boldsymbol{W}^T\boldsymbol{W} + \gamma(\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}\boldsymbol{W}^T(\mathbf{x}-\mathbf{b}), \\
&= \operatorname{diag}\left[\operatorname{diag}\left[\frac{1}{\boldsymbol{A}_{11}+\frac{\gamma}{\boldsymbol{\Sigma}_{11}+\boldsymbol{\mu}_{11}^2}}, \cdots, \frac{1}{\boldsymbol{A}_{nn}+\frac{\gamma}{\boldsymbol{\Sigma}_{nn}+\boldsymbol{\mu}_{nn}^2}}\right]\right]\boldsymbol{W}^T(\mathbf{x}-\mathbf{b}). \qquad (5.14)
\end{aligned}$$

Using Equations 5.12, 5.13, and 5.14, we can further simplify Equation 5.11 to

$$\begin{aligned}
\mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x},\mathbf{u}) = -\frac{1}{2}\Bigg(& m\log\gamma + \sum_{i=1}^{n}\frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2+\boldsymbol{\Sigma}_{ii}} + \log\left(\prod_i\left(\boldsymbol{\mu}_{ii}^2+\boldsymbol{\Sigma}_{ii}\right)\right) \\
& + \frac{1}{\gamma}\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\left(\boldsymbol{W}^T\boldsymbol{W}+\gamma(\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}\boldsymbol{W}^T(\mathbf{x}-\mathbf{b})\right\|_2^2 \\
& - \log\left(\prod_i\frac{1}{\frac{\boldsymbol{A}_{ii}}{\gamma}+\frac{1}{\boldsymbol{\mu}_{ii}^2+\boldsymbol{\Sigma}_{ii}}}\right)\Bigg), \\
= -\frac{1}{2}\Bigg(& m\log\gamma + \sum_{i=1}^{n}\frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2+\boldsymbol{\Sigma}_{ii}} \\
& + \frac{1}{\gamma}\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\left(\boldsymbol{W}^T\boldsymbol{W}+\gamma(\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}\boldsymbol{W}^T(\mathbf{x}-\mathbf{b})\right\|_2^2 \\
& + \sum_{i=1}^{n}\log\left(\frac{1}{\gamma}\left(\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii}+\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^2\right)+\frac{\boldsymbol{\Sigma}_{ii}+\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\Sigma}_{ii}+\boldsymbol{\mu}_{ii}^2}\right)\Bigg), \\
= -\frac{1}{2}\Bigg(& (m-n)\log\gamma + \sum_{i=1}^{n}\frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2+\boldsymbol{\Sigma}_{ii}} \\
& + \frac{1}{\gamma}\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\left(\boldsymbol{W}^T\boldsymbol{W}+\gamma(\boldsymbol{\Sigma}^\dagger)^{-1}\right)^{-1}\boldsymbol{W}^T(\mathbf{x}-\mathbf{b})\right\|_2^2
\end{aligned}$$

$$+\sum_{i=1}^{n}\log\left(\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii}+\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^{2}+\gamma\right)\Bigg). \tag{5.15}$$

When $\boldsymbol{A}$ has $k=n$ non-zero diagonal values and an iVAE reconstruction is precise, we have

$$\lim_{\gamma\to 0}\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\left(\boldsymbol{W}^{T}\boldsymbol{W}+\gamma(\boldsymbol{\Sigma}^{\dagger})^{-1}\right)^{-1}\boldsymbol{W}^{T}(\mathbf{x}-\mathbf{b})\right\|_{2}^{2}=\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\boldsymbol{W}^{+}(\mathbf{x}-\mathbf{b})\right\|_{2}^{2}, \tag{5.16}$$

where $\boldsymbol{W}^{+}\triangleq(\boldsymbol{W}^{T}\boldsymbol{W})^{-1}\boldsymbol{W}^{T}$ is the left pseudoinverse of $\boldsymbol{W}$, indicating that we obtain the closest possible approximation of $(\mathbf{x}-\mathbf{b})$ resulting in a reconstruction loss close to 0. Furthermore,

$$\lim_{\gamma\to 0}\sum_{i=1}^{n}\log\left(\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii}+\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^{2}+\gamma\right)=\sum_{i=1}^{n}\log\left(\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii}+\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^{2}\right). \tag{5.17}$$

From Equations 5.16 and 5.17, we can conclude that

Proposition 5.2.1 (Active variables of linear iVAEs)**.** *When a variable with index i is used for reconstruction (i.e., the variable is active), we have:*

- $\boldsymbol{\Sigma}_{ii}\ll 1$
- $\boldsymbol{\Sigma}^{\dagger}_{ii}\approx\boldsymbol{\mu}_{ii}^{2}$.

Proof. We can see that as $\gamma\to 0$ one could further reduce Equation 5.17 by either setting $\boldsymbol{\mu}\ll 1$, or $\boldsymbol{\Sigma}\ll 1$.

Case 1 If we set $\boldsymbol{\mu}\ll 1$, then

$$\lim_{\gamma\to 0}\frac{1}{\gamma}\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\boldsymbol{\mu}\right\|_{2}^{2}=\frac{1}{\gamma}\left\|\mathbf{x}-\mathbf{b}\right\|_{2}^{2}=+\infty, \tag{5.18}$$

which contradicts the fact that we obtain a highly precise reconstruction when $\gamma\to 0$. Thus, we cannot have $\boldsymbol{\mu}\ll 1$.

Case 2 If we set $\boldsymbol{\Sigma}\ll 1$, then

$$\mathcal{L}_{\overline{\text{liVAE}}}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x},\mathbf{u})\approx-\frac{1}{2}\Bigg((m-n)\log\gamma+n+\sum_{i=1}^{n}\log\left(\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^{2}+\gamma\right)$$
$$+\frac{1}{\gamma}\left\|\mathbf{x}-\mathbf{b}-\boldsymbol{W}\left(\boldsymbol{W}^{T}\boldsymbol{W}+\gamma(\boldsymbol{\mu}\boldsymbol{\mu}^{T})^{-1}\right)^{-1}\boldsymbol{W}^{T}(\mathbf{x}-\mathbf{b})\right\|_{2}^{2}\Bigg), \tag{5.19}$$

and the ELBO is maximised as $\lim_{\gamma \to 0} \mathcal{L}_{\overline{\text{liVAE}}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = +\infty$.
Furthermore, $\mathcal{L}_{\overline{\text{liVAE}}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) \geqslant \mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u})$, so we must have $\boldsymbol{\Sigma} \ll 1$, which proves the first point of Proposition 5.2.1.
We know from Equation 5.12 that $\boldsymbol{\Sigma}^{\dagger}_{ii} \approx \boldsymbol{\Sigma}_{ii} + \boldsymbol{\mu}^2_{ii}$. Given that $\boldsymbol{\Sigma} \ll 1$, we have $\boldsymbol{\Sigma}^{\dagger}_{ii} \approx \boldsymbol{\mu}^2_{ii}$, which proves the second point of Proposition 5.2.1. □

So far, we have studied the case where we needed $k = n$ active variables to correctly reconstruct the output. Now, we will consider the case where only $k < n$ variables are sufficient to provide a good reconstruction. First, let us partition the matrix $\boldsymbol{W}$, such that $\boldsymbol{W} = \begin{bmatrix} \overline{\boldsymbol{W}}_1 & \overline{\boldsymbol{W}}_2 \end{bmatrix}$, where $\overline{\boldsymbol{W}}_1 \in \mathbb{R}^{m \times k}$ is the block containing the first k columns of $\boldsymbol{W}$ and $\overline{\boldsymbol{W}}_2 \in \mathbb{R}^{m \times (n-k)}$ the remaining $n - k$ columns. Moreover, because $\boldsymbol{W}^T\boldsymbol{W}$ has only k non-zero values, $\overline{\boldsymbol{W}}_2^T \overline{\boldsymbol{W}}_2 = \mathbf{0}$, which implies that $\overline{\boldsymbol{W}}_2 = \mathbf{0}$. So we can rewrite $\boldsymbol{W} = \begin{bmatrix} \overline{\boldsymbol{W}}_1 & 0 \end{bmatrix}$. In the same way, we can partition $\boldsymbol{\Sigma}^{\dagger}$ into $\boldsymbol{\Sigma}^{\dagger} = \begin{bmatrix} \overline{\boldsymbol{\Sigma}}^{\dagger}_1 & \mathbf{0} \\ \mathbf{0} & \overline{\boldsymbol{\Sigma}}^{\dagger}_2 \end{bmatrix}$, where $\overline{\boldsymbol{\Sigma}}^{\dagger}_1 \in \mathbb{R}^{k \times k}$ is the square block containing the first k rows and columns of $\boldsymbol{W}$ and $\overline{\boldsymbol{\Sigma}^{\dagger}}_2 \in \mathbb{R}^{(n-k) \times (n-k)}$ the remaining $n - k$ rows and columns. Using these two partitioned matrices and Equation 5.14, we can rewrite $\boldsymbol{W}\boldsymbol{\mu}$ as

$$\begin{aligned}
\boldsymbol{W}\boldsymbol{\mu} &= \begin{bmatrix} \overline{\boldsymbol{W}}_1 & 0 \end{bmatrix} \begin{bmatrix} \left(\overline{\boldsymbol{W}}_1^T \overline{\boldsymbol{W}}_1 + \gamma \left(\overline{\boldsymbol{\Sigma}}^{\dagger}_1\right)^{-1}\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \frac{1}{\gamma}\overline{\boldsymbol{\Sigma}}^{\dagger}_2 \end{bmatrix} \begin{bmatrix} \overline{\boldsymbol{W}}_1^T \\ \mathbf{0} \end{bmatrix} (\mathbf{x} - \mathbf{b}), \\
&= \begin{bmatrix} \overline{\boldsymbol{W}}_1 & 0 \end{bmatrix} \begin{bmatrix} \left(\overline{\boldsymbol{W}}_1^T \overline{\boldsymbol{W}}_1 + \gamma \left(\overline{\boldsymbol{\Sigma}}^{\dagger}_1\right)^{-1}\right)^{-1} \overline{\boldsymbol{W}}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} (\mathbf{x} - \mathbf{b}), \\
&= \begin{bmatrix} \overline{\boldsymbol{W}}_1 \left(\overline{\boldsymbol{W}}_1^T \overline{\boldsymbol{W}}_1 + \gamma \left(\overline{\boldsymbol{\Sigma}}^{\dagger}_1\right)^{-1}\right)^{-1} \overline{\boldsymbol{W}}_1^T & 0 \end{bmatrix} (\mathbf{x} - \mathbf{b}).
\end{aligned} \tag{5.20}$$

When $\gamma \to 0$, we retrieve

$$\boldsymbol{W}\boldsymbol{\mu} = \begin{bmatrix} \overline{\boldsymbol{W}}_1 \overline{\boldsymbol{W}}_1^+ & 0 \end{bmatrix} (\mathbf{x} - \mathbf{b}). \tag{5.21}$$

Thus, if having more than $k < n$ active variables does not change the reconstruction, $\begin{bmatrix} \overline{\boldsymbol{W}}_1 & 0 \end{bmatrix} \boldsymbol{\mu}$ will be the closest possible approximation of $(\mathbf{x} - \mathbf{b})$ resulting in a reconstruction loss close to 0. Updating the ELBO for $k < n$, and letting

$$\boldsymbol{B} \triangleq \begin{bmatrix} \overline{\boldsymbol{W}}_1 \left(\overline{\boldsymbol{W}}_1^T \overline{\boldsymbol{W}}_1 + \gamma \left(\overline{\boldsymbol{\Sigma}}^{\dagger}_1\right)^{-1}\right)^{-1} \overline{\boldsymbol{W}}_1^T & 0 \end{bmatrix},$$

we get

$$\mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = -\frac{1}{2}\Bigg(m \log \gamma + \sum_{i=1}^{k} \frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2 + \boldsymbol{\Sigma}_{ii}} + \sum_{i=k+1}^{n} \frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2 + \boldsymbol{\Sigma}_{ii}} + \frac{1}{\gamma}\Big\| \mathbf{x} - \mathbf{b} - \boldsymbol{B}(\mathbf{x} - \mathbf{b}) \Big\|_2^2 + \sum_{i=1}^{k} \log\Big(\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii} + \boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^2 + 1 \Big) \Bigg). \tag{5.22}$$

We can directly see that Equation 5.22 can further be reduced by setting $\boldsymbol{\mu}_{ii}^2 = \mathbf{0}\ \forall i \in \{k+1, \cdots, n\}$, such that

$$\mathcal{L}_{\text{liVAE}}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = -\frac{1}{2}\Bigg(m \log \gamma + \sum_{i=1}^{k} \frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2 + \boldsymbol{\Sigma}_{ii}} + \frac{1}{\gamma}\Big\| \mathbf{x} - \mathbf{b} - \boldsymbol{B}(\mathbf{x} - \mathbf{b}) \Big\|_2^2 + \sum_{i=1}^{k} \log\Big(\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii} + \boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^2 + 1 \Big) \Bigg). \tag{5.23}$$

It follows that

Proposition 5.2.2 (Passive variables of linear iVAEs)**.** *When a variable with index i is not used for reconstruction (i.e., the variable is passive), we have:*

- $\boldsymbol{\mu}_{ii}^2 \approx \mathbf{0}$,
- $\boldsymbol{\Sigma}_{ii} \approx \boldsymbol{\Sigma}_{ii}^{\dagger}$.

Proof. We can see from Equations 5.22 and 5.23 that, when $k < n$, the ELBO is maximised when $\boldsymbol{\mu}_{ii}^2 = \mathbf{0}\ \forall i \in \{k+1, \cdots, n\}$, and any $\boldsymbol{\mu}_{ii}^2 \neq \mathbf{0}$ for $i \in \{k+1, \cdots, n\}$ will reduce this score, which proves the first point. The second point naturally follows. We know from Equation 5.12 that $\boldsymbol{\Sigma}_{ii}^{\dagger} = \boldsymbol{\Sigma}_{ii} + \boldsymbol{\mu}_{ii}^2$, thus if $\boldsymbol{\mu}_{ii} \approx 0$ then $\boldsymbol{\Sigma}_{ii}^{\dagger} = \boldsymbol{\Sigma}_{ii}$. □

Given Equations 5.22 and 5.23, we can further conclude that linear iVAEs behave in a polarised regime when $\boldsymbol{\mu}^{\dagger} = \mathbf{0}$.

Theorem 5.2.2 (Linear iVAEs with $\boldsymbol{\mu}^{\dagger} = \mathbf{0}$ learn in a polarised regime)**.** *Any linear iVAE with $\boldsymbol{\mu}^{\dagger} = \mathbf{0}$ whose reconstruction is not improved by using more than $k < n$ active variables will be composed of k active variables and $n - k$ passive variables.*

Proof. Let us consider the case where we have the same reconstruction loss with k latent variables and $k+1$ latent variables, and $0 < k < k+1 \leqslant n$. If the $(k+1)^{th}$ variable is active, we have

$$\mathcal{L}_{\text{liVAE}_a}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = -\frac{1}{2}\left(m \log \gamma + \sum_{i=1}^{k+1} \frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2 + \boldsymbol{\Sigma}_{ii}} + \frac{1}{\gamma}\left\|\mathbf{x} - \mathbf{b} - \boldsymbol{W}\boldsymbol{\mu}\right\|_2^2 \right.$$
$$\left. + \sum_{i=1}^{k+1} \log\left(\frac{1}{\gamma}\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii} + \frac{1}{\gamma}\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^2 + 1 \right) \right). \tag{5.24}$$

However, if the variable is passive, we have

$$\mathcal{L}_{\text{liVAE}_p}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) = -\frac{1}{2}\left(m \log \gamma + \sum_{i=1}^{k} \frac{\boldsymbol{\mu}_{ii}^2}{\boldsymbol{\mu}_{ii}^2 + \boldsymbol{\Sigma}_{ii}} + \frac{1}{\gamma}\left\|\mathbf{x} - \mathbf{b} - \boldsymbol{W}\boldsymbol{\mu}\right\|_2^2 \right.$$
$$\left. + \sum_{i=1}^{k} \log\left(\frac{1}{\gamma}\boldsymbol{A}_{ii}\boldsymbol{\Sigma}_{ii} + \frac{1}{\gamma}\boldsymbol{A}_{ii}\boldsymbol{\mu}_{ii}^2 + 1 \right) \right). \tag{5.25}$$

Because the reconstruction loss is the same in Equations 5.24 and 5.25, we will always have $\mathcal{L}_{\text{liVAE}_p}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u}) > \mathcal{L}_{\text{liVAE}_a}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{u})$. Thus, given that we seek to maximise the ELBO, any variables that do not improve the reconstruction loss must be passive and display the properties detailed in Proposition 5.2.2. □

5.3 EMPIRICAL VERIFICATION

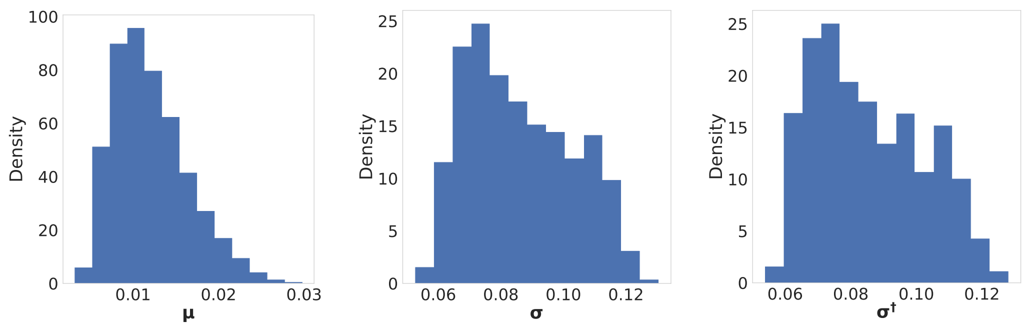
The objective of this section is twofold: (1) to empirically verify the propositions and theorems presented in Section 5.2 and (2) to provide evidences that the analysis of linear iVAEs can generalise to the non-linear case.

We use the dSprites dataset[1] (Higgins et al., 2017). Note that all the histograms presented below are computed using 10000 input examples, and the latent space is set to a larger number of dimensions than usual (30 instead of 10) to ensure the presence of superfluous (passive) variables. Further details about the model architecture and source code can be found in Appendix A.1.

1 Licensed under an Apache 2.0 licence.

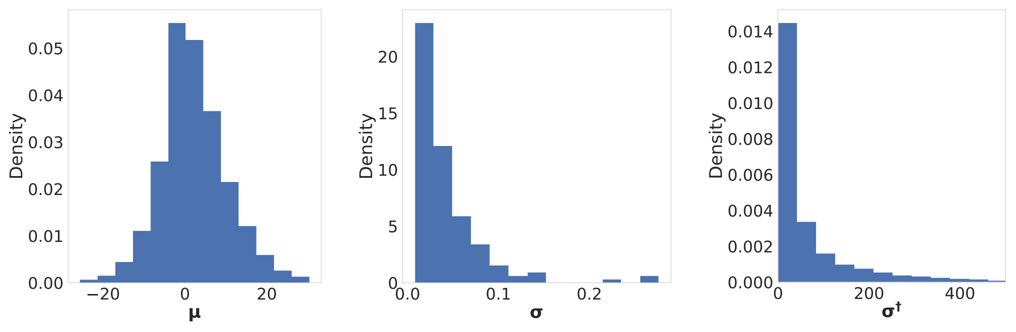
5.3.1 *Behaviour of iVAEs when the prior mean is fixed to 0*

In this section we will focus on empirically verifying the theoretical results of Section 5.2.



(a) Mean (b) Variance (c) Prior variance

Figure 5.1: Empirical distributions of a passive variable of an iVAE trained on dSprites. (a) and (b) correspond to the mean and variance of the posterior, and (c) is the variance of the prior. The same passive variable is used for all plots in this figure.



(a) Mean (b) Variance (c) Prior variance

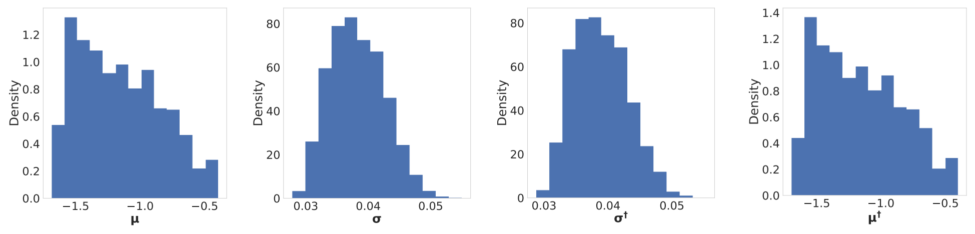
Figure 5.2: Empirical distributions of an active variable of an iVAE trained on dSprites. (a) and (b) correspond to the mean and variance of the posterior, and (c) is the variance of the prior. The same active variable is used for all plots in this figure.

Illustration of Proposition 5.2.2 We can see in Figure 5.1a that the empirical distribution of a passive variable of the mean representation consistently takes values very close to zero, which confirms Proposition 5.2.2 for the configuration where the mean is fixed. Moreover, the active variables of the mean representation tend to have higher variance as they encode more information, as illustrated in Figure 5.2a. We can also see that the passive variables of the variance representation stay close to the variance representation of the prior to minimise the KLD, as seen in Figures 5.1b and 5.1c. Interestingly, in the case of passive variables, the variance representation of the prior stays close to zero. Thus, iVAEs seem to behave in a near-deterministic way as the variance is kept low for both types of variables.

Illustration of Proposition 5.2.1 Figure 5.2b shows that the active variables of the variance representation will remain close to zero. Moreover, when compared to Figure 5.2c, we can see that the active variables of the variance representation also depart from their prior distribution, as their objective is to maximise the reconstruction by reducing the noise during the reparametrisation. Furthermore, given a passive variable with index i, the empirical distribution of $\Sigma^{\dagger}_{ii}$ closely matches what would be obtained by computing the histogram of $\boldsymbol{\mu}^2_{ii}$. For example, we can see in Figure 5.2 that we approximately have $\boldsymbol{\mu}_{ii} \in [-20, 20]$, and the largest value of $\Sigma^{\dagger}_{ii} \approx \boldsymbol{\mu}^2_{ii} = 400$. This is consistent with the second point of Proposition 5.2.1.

5.3.2 *Behaviour of iVAEs when the prior mean is learned*

In the previous section, we have provided empirical evidence of the theoretical results from Section 5.2 and shown that they were likely to generalise to non-linear models. In this section, we will provide some empirical evidence of polarised regime in non-linear iVAEs whose prior mean $\boldsymbol{\mu}^{\dagger}$ is learned.

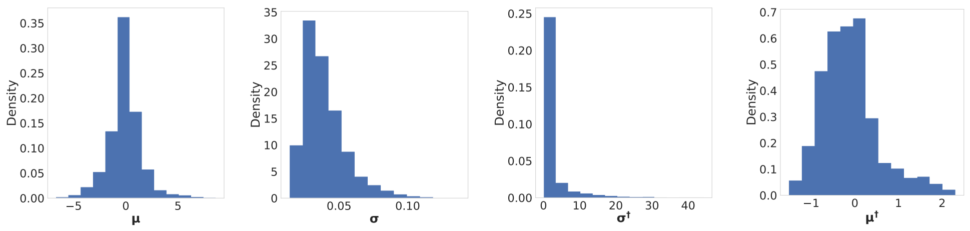


(a) Mean (b) Variance (c) Prior variance (d) Prior mean

Figure 5.3: Empirical distributions of a passive variable of an iVAE with learned $\boldsymbol{\mu}^{\dagger}$ trained on dSprites. (a) and (b) correspond to the mean and variance of the posterior; (c) and (d) are the mean and variance of the prior. The same passive variable is used for all plots in this figure.

We can see in Figure 5.3d that passive variables are still present when the prior mean is learned. In line with Equations 5.4 and 5.6, the prior and posterior mean are very close to each other, as are the prior and posterior variance. Furthermore, all the values tend to 0.

Active variables can also be found, as shown in Figure 5.4d. In this case, the posterior mean departs from its prior, as does the posterior variance. As in other types of VAEs, the posterior variance is close to 0 to increase the precision of the



(a) Mean (b) Variance (c) Prior variance (d) Prior mean

Figure 5.4: Empirical distributions of an active variable of an iVAE with learned $\boldsymbol{\mu}^\dagger$ trained on dSprites. (a) and (b) correspond to the mean and variance of the posterior; (c) and (d) are the mean and variance of the prior. The same active variable is used for all plots in this figure.

variance, and the posterior mean has a larger variance than in the passive case, as it encodes information used for reconstruction. While the $\boldsymbol{\Sigma}^\dagger$ still has a high variance for passive variables, it is not as high as in Figure 5.2c for iVAEs whose prior mean is fixed to 0.

Overall, this provides some evidence that a pruning mechanism is still in place when $\boldsymbol{\mu}^\dagger$ is learned, which is typical of the polarised regime.

5.4 CONCLUSION

In this chapter we sought to verify whether iVAEs learned in a polarised regime. To our investigation, we explored the impact of learning the mean and variance of the prior on the polarised regime. As in Chapter 4, we derived an analytical solution for the linear case in Section 5.2 to gain some insight into this phenomenon.

We provide theoretical and empirical evidence that iVAEs whose mean is fixed learn in a polarised regime while behaving in a near deterministic way, with a posterior variance consistently close to 0. Interestingly, the active variables of the prior variance $\boldsymbol{\Sigma}^\dagger$ of these models displayed a very high variance, which was directly linked to the values of the posterior mean, as seen in Proposition 5.2.1. One can thus wonder if this could lead to model instability should the posterior mean take a very large value on a specific dataset where, for example, the data is unbalanced and contains rare samples. While the theoretical study of iVAEs whose variance is learned was inconclusive, these models displayed polarised behaviour during the experimental evaluation.

Overall, we can thus conclude that iVAEs learn in a polarised regime. While these results can be generalised to VAEs with any Gaussian prior and posterior with diagonal covariances, extending this work to other prior and posterior distributions is left for future work.

6

IMPACT OF THE POLARISED REGIME ON THE LATENT REPRESENTATIONS

6.1 INTRODUCTION

As we have seen in Section 4.4, VAEs are considered state-of-the-art techniques to learn unsupervised disentangled representations; that is, representations encoding separately the different factors of variations (Bengio, Courville and Vincent, 2013). Disentangled representations are very attractive in terms of interpretability and fairness (Locatello et al., 2019b) and can be beneficial for downstream tasks such as abstract reasoning (van Steenkiste et al., 2019).

Over the years, multiple regularisation techniques have been developed to encourage disentanglement, with a specific focus on enforcing the learned latent factors to be uncorrelated. While this regularisation is done on the sampled aggregated posterior, the learned representation is generally taken to be the mean vector of the posterior distribution. However, Locatello et al. (2019a) reported an increased TC and averaged MI over the dimensions of the mean representation compared to the results obtained on its sampled counterpart. This finding raises questions on whether mean representations would still benefit from the appealing attributes of disentanglement, since sampled representations were shown to be less correlated and, thus, more disentangled.

As shown in Chapters 4 and 5, VAEs behave in a polarised regime, where the relevant dimensions of the sampled representations (the active variables) are used by the decoder for reconstruction, while the remaining dimensions (the passive variables) are 'shut down' to closely match the prior. However, the polarised regime has only been studied in the context of single data examples for sampled representations. Therefore, in Section 6.3, we extend the existing definition of the polarised regime to multiple data examples and explore the implications for mean and variance representations. Assuming that VAEs producing disentangled representations are behaving in a polarised regime, we show, based on this extended version, that active variables of mean representations should not be more correlated than the sampled ones. Thus, we argue that the higher correlation reported by Locatello et al. (2019a) is due to the impact of the passive variables on the metrics used. We verify this hypothesis empirically in Section 6.4 and provide further analytical justifications in Section 6.5.

Our contribution is three-fold: (1) we extend the definition of the polarised regime to mean and variance representations using multiple data examples; (2) we use this extended version to show that the discrepancies between mean and sampled representations observed by Locatello et al. (2019a) are mostly due to the impact of the polarised regime, and especially of the passive variables; and (3) we explain why passive variables are leading to higher TC and averaged MI scores. The code of our experiments is available at `https://github.com/bonheml/tc_study`.

Notational considerations Throughout this chapter, we use the superscript $^{(i)}$ to denote the values obtained for the ith sample $\mathbf{x}^{(i)}$ of the random variable $\mathbf{x}$ and we represent the jth dimension of a vector representation using the subscript $_j$. For example, given a random variable $\boldsymbol{\epsilon}$ distributed according to $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, $\boldsymbol{\epsilon}_j^{(i)}$ is the jth dimension of the sample of $\boldsymbol{\epsilon}$ obtained for $\mathbf{x}^{(i)}$. While the mean and diagonal variance representations are functions of $\mathbf{x}$ and the network parameters ϕ, we use a shortened version when the meaning is clear from the context, such that $\boldsymbol{\mu} \triangleq \boldsymbol{\mu}(\mathbf{x}; \phi)$ and $\boldsymbol{\sigma} \triangleq \operatorname{diag}[\boldsymbol{\Sigma}(\mathbf{x}; \phi)]$. Similarly, for a specific sample $\mathbf{x}^{(i)}$, $\boldsymbol{\mu}^{(i)} \triangleq \boldsymbol{\mu}(\mathbf{x}^{(i)}; \phi)$ and $\boldsymbol{\sigma}^{(i)} \triangleq \operatorname{diag}[\boldsymbol{\Sigma}(\mathbf{x}^{(i)}; \phi)]$. We adopt the same notation for the sampled representation, such that $\mathbf{z} \triangleq \boldsymbol{\mu} + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}^{1/2}$ and $\mathbf{z}^{(i)} \triangleq \boldsymbol{\mu}^{(i)} + \boldsymbol{\epsilon}^{(i)} \odot (\boldsymbol{\sigma}^{(i)})^{1/2}$. For multiple data examples $\boldsymbol{X} = \{\mathbf{x}^{(i)}\}_{i=0}^{h}$, $\boldsymbol{M} \triangleq [\boldsymbol{\mu}^{(0)} \cdots \boldsymbol{\mu}^{(h)}]^T$, $\boldsymbol{S} \triangleq [\boldsymbol{\sigma}^{(0)} \cdots \boldsymbol{\sigma}^{(h)}]^T$, $\boldsymbol{E} \triangleq [\boldsymbol{\epsilon}^{(0)} \cdots \boldsymbol{\epsilon}^{(h)}]^T$, and $\boldsymbol{Z} \triangleq [\mathbf{z}^{(0)} \cdots \mathbf{z}^{(h)}]^T$.

6.2 BACKGROUND

In this chapter we are interested in investigating the discrepancies between the mean and sampled representations—$\boldsymbol{\mu}$ and $\mathbf{z}$, respectively. Specifically, our goal is to explain the higher correlation of mean representations reported by Locatello et al. (2019a). In the following sections, the representations learned by the mean layer will be referred to as mean representations, and those learned during the sampling stage as sampled representations.

6.2.1 *Disentangled representation learning with VAEs*

While various learning objectives and architectures have been proposed for VAEs, we will focus on the β-VAE family of models, presented in Section 4.3, to provide consistent results with Locatello et al. (2019a), who used the same models.

Note that, because DIP-VAE I directly encourages diagonal covariance matrices in the mean representation, it will have a low correlation in the mean representation, which, as observed by Locatello et al. (2019a), mirrors the correlations in the sampled representation. Moreover, the discrepancies between mean and sampled representations were observed by Locatello et al. (2019a) in the context of methods which explicitly regularise the disentanglement of $\mathbf{z}$, but not $\boldsymbol{\mu}$. In this study, we will thus consider DIP-VAE II, which enforces the covariance matrix of the sampled representation to be diagonal, but not DIP-VAE I, as it explicitly regularises $\boldsymbol{\mu}$.

6.2.2 *Benefits of disentanglement on downstream tasks*

Disentangled representations have been shown to reduce the sample complexity of abstract reasoning tasks (van Steenkiste et al., 2019) and improve the fairness of downstream task models (Locatello et al., 2019b; Creager et al., 2019) and their interpretability (Higgins et al., 2017; Adel, Ghahramani and Weller, 2018). However, when the mean representations are more correlated than the sampled representations on which the disentanglement is measured, it will hamper the interpretability of downstream task models (Alin, 2010; Chan et al., 2022) and reduce their fairness (Locatello et al., 2019b; Träuble et al., 2021). Moreover, we have seen in Section 4.4 that, under a certain level of supervision, VAEs can provably provide identifiable representations (Khemakhem et al., 2020; Mita, Filippone and Michiardi, 2021), and it is conjectured that under specific constraints this is also possible in the unsupervised setting (Reizinger et al., 2022). As we have shown in Chapter 5 that iVAEs also behave in a polarised regime, it is important to investigate the origin of the discrepancies between mean and sampled representations to determine if one can still benefit from disentanglement when using mean representations on downstream tasks.

6.2.3 *Related work*

The discrepancy between the TC of mean and sampled representations observed by Locatello et al. (2019a) has recently been investigated by Cheng et al. (2021) who provide a theoretical justification of the higher TC scores of the mean representations. However, they did not consider the polarised regime, which is a necessary condition for VAEs to provide good reconstruction (Dai and Wipf, 2018; Rolinek, Zietlow and Martius, 2019; Dai, Wang and Wipf, 2020). Thus, their work is complementary to ours in the case where VAEs are not learning in a polarised regime.

6.3 EXTENSION OF THE POLARISED REGIME

As discussed in Chapter 4, the polarised regime is the ability of VAEs to 'shut down' superfluous dimensions of their sampled latent representations while providing a high precision on the remaining ones (Rolinek, Zietlow and Martius, 2019; Dai, Wang and Wipf, 2020). As a result, the sampled representation can be separated into two subsets of variables, active and passive. The active variables correspond to the subset of the sampled latent representation that is needed for the reconstruction. They have a low variance, and are close to the mean variables. The passive variables correspond to the superfluous dimensions that are discarded by the VAE. They follow a zero-mean unit-variance Gaussian distribution to optimally match the prior and are ignored by the decoder, which only uses the variables that help to reconstruct the input.

We have seen in Section 4.2 that the existence of active and passive variables is a necessary condition for the VAEs to provide a good reconstruction (Dai and Wipf, 2018; Dai, Wang and Wipf, 2020). However, when the weight on the regularisation term of the ELBO given in Equation 3.5 increases, VAEs prune more active variables to minimise the regularisation loss, as discussed in Section 4.3. When this weight becomes too large, the representations collapse to the prior, containing only passive variables (Lucas et al., 2019b; Dai, Wang and Wipf, 2020).

While Definition 4.1.1 provides an overview of the polarised regime for each data example, it is not readily usable to analyse discrepancies between mean and sampled representations, as they are observed over the whole dataset. We will thus extend the definition of the polarised regime to multiple data examples in Section 6.3.1 and show that the discrepancies between mean and sampled representations can only originate from variables which are not active.

6.3.1 *Generalisation of the polarised regime to multiple data examples*

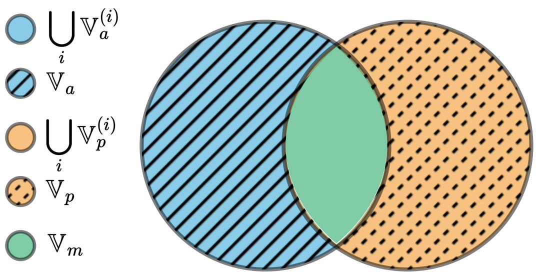
Now we will propose a new generalisation of Definition 4.1.1 to multiple data examples, which will serve as a basis for our analysis in Section 6.4. Given that a variable can either be active or passive for a given data example, when considering multiple data examples, three cases arise:

- A variable is passive for all the data examples.
- A variable is active for all the data examples.
- A variable is active for some data examples and passive otherwise.

These three types of variables are formalised in Definition 6.3.1 and illustrated in Figure 6.1.

Definition 6.3.1 (Variable types)**.** When considered over multiple data examples $\boldsymbol{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{n}$, the latent representations are composed of a set of passive, active, and mixed variables $\mathbb{V}_p \cup \mathbb{V}_a \cup \mathbb{V}_m$, which are defined as follows:

1. $\mathbb{V}_p \triangleq \bigcap_{i=1}^{n} \mathbb{V}_p^{(i)}$,
2. $\mathbb{V}_a \triangleq \bigcap_{i=1}^{n} \mathbb{V}_a^{(i)}$,
3. $\mathbb{V}_m \triangleq \left(\bigcup_{i=1}^{n} \mathbb{V}_p^{(i)}\right) \bigcap \left(\bigcup_{i=1}^{n} \mathbb{V}_a^{(i)}\right)$.



$\bigcup_i \mathbb{V}_a^{(i)}$
$\mathbb{V}_a$
$\bigcup_i \mathbb{V}_p^{(i)}$
$\mathbb{V}_p$
$\mathbb{V}_m$

Figure 6.1: A graphical representation of Definition 6.3.1.

Based on Definitions 4.1.1 and 6.3.1, we will consider the properties of the mean and variance representations over multiple data examples in Propositions 6.3.1 and 6.3.2, then describe their implications for the sampled representations in Theorem 6.3.1. Given a Gaussian decoder with diagonal covariance $\gamma\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$, Dai and Wipf (2018) proved that for any active variable j, $\lim_{\gamma\to 0} \mathbf{z}_j^{(i)} = \boldsymbol{\mu}_j^{(i)}$ (See Section 4.2 for a discussion in the linear case). In the same way, for any passive variable j, $\lim_{\gamma\to 0} \mathbf{z}_j^{(i)} = \boldsymbol{\epsilon}_j^{(i)}$. In practice, this limit is approached very early in the training (i.e., after a few epochs), as shown in Dai and Wipf (2018, Fig.1.a).

Proposition 6.3.1 (Polarised regime of $\boldsymbol{M}$)**.** *When a VAE learns in a polarised regime, its mean representation $\boldsymbol{M}$ is composed of a set of passive, active, and mixed variables $\mathbb{V}_p \cup \mathbb{V}_a \cup \mathbb{V}_m$ such that, over $\boldsymbol{X}$:*

1. $\lim_{\gamma \to 0} \bar{\boldsymbol{M}}_j = 0$ *and* $\lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{M}_j) \ll 1 \quad \forall\, j \in \mathbb{V}_p$,
2. $\lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{M}_j) > \lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{M}_k) \quad \forall\, j \in \mathbb{V}_a\,, \forall\, k \in \mathbb{V}_p$,

where $\bar{\boldsymbol{M}}_j \triangleq \frac{1}{n}\sum_{i=0}^{n} \boldsymbol{\mu}_j^{(i)}$, *and* $\mathrm{Var}(\cdot)$ *denotes the variance.*

The first point of Proposition 6.3.1 indicates that passive variables of mean representations are almost constant when considered over multiple data examples. Indeed, because they are passive for each data example, according to (1) of Definition 4.1.1 they will consistently take values close to 0. Thus, they will have a variance and expected value close to 0.

Because active variables of the mean representation encode some information about the input, their value will vary depending on the input and thus have a higher variance than their passive counterpart, as stated in (2) of Proposition 6.3.1.

Now, let us analyse the effect of the polarised regime on sampled representations over multiple data examples.

Proposition 6.3.2 (Polarised regime of $\boldsymbol{S}$)**.** *When a* VAE *learns in a polarised regime, its variance representation* $\boldsymbol{S}$ *is composed of a set of passive, active and mixed variables* $\mathbb{V}_p \cup \mathbb{V}_a \cup \mathbb{V}_m$ *such that, over* $\boldsymbol{X}$*:*

1. $\lim_{\gamma \to 0} \bar{\boldsymbol{S}}_j = 1$ *and* $\lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{S}_j) = 0 \quad \forall j \in \mathbb{V}_p$,
2. $\lim_{\gamma \to 0} \bar{\boldsymbol{S}}_j = 0$ *and* $\lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{S}_j) = 0 \quad \forall j \in \mathbb{V}_a$,
3. $\lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{S}_j) < \lim_{\gamma \to 0} \mathrm{Var}(\boldsymbol{S}_k) \quad \forall j \notin \mathbb{V}_m\,, \forall k \in \mathbb{V}_m$,

where $\bar{\boldsymbol{S}}_j \triangleq \frac{1}{n}\sum_{i=0}^{n} \boldsymbol{\sigma}_j^{(i)}$.

We know from Definition 4.1.1 that the variance representation is always close to 1 when the variables are passive and to 0 when they are active. Thus, variables that are passive (resp. active) over the whole dataset will be almost constant, with an expected value close to 1 (resp. 0), as stated in (1) and (2) of Proposition 6.3.2.

The variance representations of mixed variables will alternate between 1 and 0, depending on whether they are passive or active for the considered data examples. Thus, as described in (3) of Proposition 6.3.2, they will vary more than active and passive variables.

Theorem 6.3.1 (Polarised regime of $\boldsymbol{Z}$)**.** *When a* VAE *learns in a polarised regime, its sampled representation* $\boldsymbol{Z}$ *is composed of a set of passive, active and mixed variables* $\mathbb{V}_p \cup \mathbb{V}_a \cup \mathbb{V}_m$ *such that:*

1. $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \boldsymbol{E}_j \quad \forall\, j \in \mathbb{V}_p$,

2. $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \boldsymbol{M}_j \quad \forall\, j \in \mathbb{V}_a$,

3. $\lim_{\gamma \to 0} \boldsymbol{Z}_j^{(\boldsymbol{p})} = \boldsymbol{E}_j^{(\boldsymbol{p})}$ *and* $\lim_{\gamma \to 0} \boldsymbol{Z}_j^{(\boldsymbol{a})} = \boldsymbol{M}_j^{(\boldsymbol{a})} \quad \forall\, j \in \mathbb{V}_m$,

where $\boldsymbol{Z}_j^{(\boldsymbol{p})}$ *denotes the subsets of examples for which* $\boldsymbol{Z}_j$ *is passive, and* $\boldsymbol{Z}_j^{(\boldsymbol{a})}$ *is the subsets of examples for which* $\boldsymbol{Z}_j$ *is active.*

Proof. Let $\boldsymbol{Z}_j$ be the sampled representation variable at index j. There are three cases:

1. If $j \in \mathbb{V}_p$, then, from statement (1) of Proposition 6.3.1, $\boldsymbol{M}_j$ is almost constant with a value close to 0. Thus, $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \boldsymbol{E}_j \boldsymbol{S}_j^{1/2}$. Using statement (1) of Proposition 6.3.2, we also know that $\boldsymbol{S}_j$ is almost constant with a value close to 1, thus $\lim_{\gamma \to 0} \boldsymbol{S}_j = 1$ and $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \boldsymbol{E}_j$. It follows that $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \boldsymbol{E}_j$, which proves statement (1).

2. If $j \in \mathbb{V}_a$, then, from statement (2) of Proposition 6.3.2, $\lim_{\gamma \to 0} \boldsymbol{S}_j = 0$. Thus, $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \boldsymbol{M}_j$. It follows that $\lim_{\gamma \to 0} \boldsymbol{Z}_j = \lim_{\gamma \to 0} \boldsymbol{M}_j$, which proves statement (2).

3. If $j \in \mathbb{V}_m$, then from statement (3) of Definition 6.3.1 we know that $\boldsymbol{Z}_j$ is composed of a subset of active components and a subset of passive components, $\boldsymbol{a}$ and $\boldsymbol{p}$. Using step (1) and (2) of the proof, we know that $\lim_{\gamma \to 0} \boldsymbol{Z}_j^{(\boldsymbol{p})} = \boldsymbol{E}_j^{(\boldsymbol{p})}$ for passive variables and $\lim_{\gamma \to 0} \boldsymbol{Z}_j^{(\boldsymbol{a})} = \boldsymbol{M}_j^{(\boldsymbol{a})}$ for active variables which proves statement (3). This concludes the proof.

□

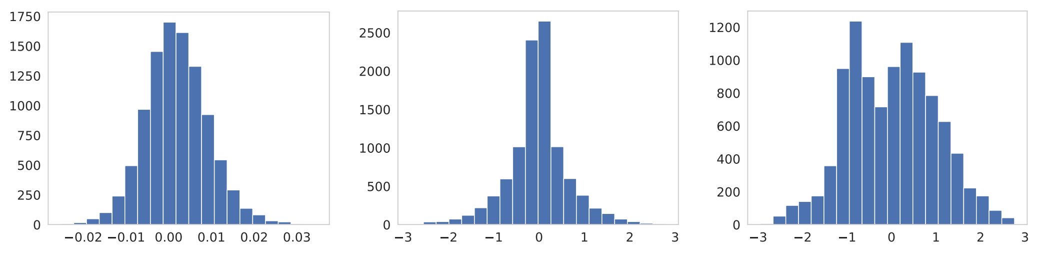
Given that active variables are the only type of variables with approximately the same distributions in mean and sampled representations, Corollary 6.3.1 immediately follows.

Corollary 6.3.1. *Any discrepancies between the distributions of the mean and sampled representations can only come from the mixed and passive variables.*

6.3.2 *Empirical demonstration of the polarised regime*

We will now verify that Theorem 6.3.1 holds empirically using a β-VAE trained on the dSprites dataset (Higgins et al., 2017). By comparing the passive variable distribution of mean and sampled representations provided in Figures 6.2a and 6.3a, we can see that both have a mean of zero, and that the variance of the variable is close to zero in the mean representation and to one in the sampled representation, consistent with statement (1) of Theorem 6.3.1 and Proposition 6.3.1. As described in statement (2) of Theorem 6.3.1, the active variable of the mean representation observed in Figure 6.2c follows a similar distribution as its sampled counterpart in Figure 6.3c. Figures 6.2b and 6.3b show that mixed variables can also be identified. Their variance in the mean representation is larger than for passive variables and increases in the sampled representation. Moreover, we can see a sharp peak around zero in the mean representation, which is smoothed out in the sampled one. This likely corresponds to the passive component of the mixed variables, which is close to zero with very low variance in the mean representation, and to $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ in the sampled representation. Overall, these observations are consistent with a mixture distribution, as per statement (3) of Theorem 6.3.1.

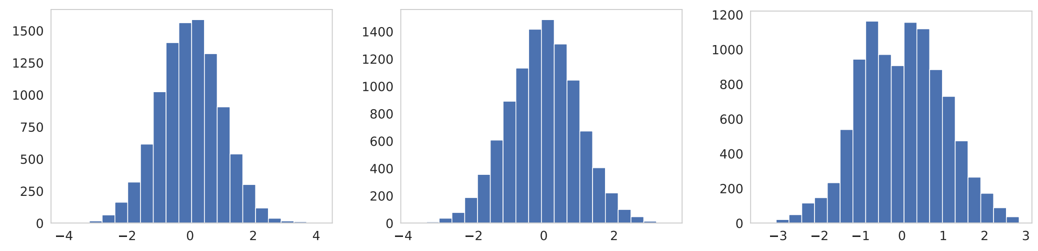
Note that, while the distribution of the passive variables of sampled representations is encouraged to be Gaussian by the KLD term of the ELBO, we can see in Figures 6.2 and 6.3 that the mixed and active variables are not guaranteed to be Gaussian in mean and sampled representations. Indeed, as they convey some information about the data, their distribution can be arbitrarily different from the prior and may or may not be Gaussian. The resulting increased KLD of active variables is then compensated by passive variables that match the prior exactly. Moreover, the distribution of the passive variables of mean representations may also not be Gaussian, as it is only optimised to have low variance.



(a) Passive variable (b) Mixed variable (c) Active variable

Figure 6.2: Empirical distributions of the 2nd, 4th, and 10th latent dimensions of the mean representations of a β-VAE trained on dSprites with $\beta = 8$, which illustrate the typical appearance of passive, mixed, and active variables. The histograms are computed using 10000 input examples.

These observations also provide empirical evidence of Corollary 6.3.1, that only the active variables are similar in mean and sampled representations. When the regularisation strength increases, the sampled and mean representations will contain more passive and mixed variables, hence *the higher the regularisation weight, the greater the difference between the mean and sampled representations*. This is the



(a) Passive variable (b) Mixed variable (c) Active variable

Figure 6.3: Empirical distributions of the 2nd, 4th, and 10th latent dimensions of the sampled representations of a β-VAE trained on dSprites with $\beta = 8$, which illustrate the typical appearance of passive, mixed, and active variables. The histograms are computed using 10000 input examples.

main insight that will allow us to explain the discrepancies between the correlation scores of the mean and sampled representations observed by Locatello et al. (2019a).

6.4 ASSESSING THE IMPACT OF MIXED AND PASSIVE VARIABLES ON THE DIFFERENCES BETWEEN MEAN AND SAMPLED REPRESENTATIONS

In Theorem 6.3.1 and Section 6.3, we have seen that while active variables are equivalent in mean and sampled representations, it is not the case for mixed and passive variables. As the number of non-active variables increases with the regularisation strength, we hypothesise that they may be the source of the stronger correlation of mean representations observed by Locatello et al. (2019a).

To verify this hypothesis, we investigate if active variables alone are equivalently correlated in mean and sampled representations and how passive variables would impact the different metrics used. To do so, we divide our experiment into four steps:

1. Using Proposition 6.3.2, identify the type of variable (active, mixed or passive) stored at each index of the representations considered.

2. Based on Proposition 6.3.1 and Theorem 6.3.1, provide a theoretical explanation of the the discrepancies between the averaged Mutual Information (MI) and TC scores of mean and sampled representations for passive variables components.

3. Verify that the correlation of mean representations increases with the number of passive and mixed variables. This will allow us to explore how the number of passive variables evolves with stronger regularisation and whether this number

can explain the discrepancies in TC and averaged MI that were reported by Locatello et al. (2019a).

4. Empirically verify the results of step (2) by comparing the impact of passive and mixed variables on averaged MI and TC scores for mean and sampled representations. By separately comparing the scores of every combination of variable type, we can further attribute the discrepancies to a specific type of variable, or a combination of variables.

These steps are implemented in Sections 6.4.1, 6.4.2, 6.4.3 and 6.4.4, respectively.

Empirical setup We based our implementation on `disentanglement_lib` (Locatello et al., 2019a), using the same datasets as the authors: dSprites (Higgins et al., 2017), smallNorb (LeCun, Huang and Bottou, 2004), cars3D (Reed et al., 2015), and the alternative versions of dSprites (Locatello et al., 2019a) color-dSprites, Scream-dSprites and Noisy-dSprites. We relied on the 9000 pre-trained models released by Locatello et al. (2019a), corresponding to the 5 VAE architectures described in Section 4.3, over the 6 datasets mentioned above, with 6 different regularisation strengths for each method and 50 seeds per (dataset, method, regularisation) triplet.

6.4.1 *Identifying variable types*

Using Proposition 6.3.2 and defining a threshold $\alpha \geqslant 0$, one can easily classify the type of variables stored at each index $j \in d$ based on the variance representations:

- **Passive variable**: $\bar{\boldsymbol{S}}_j = 1 \pm \alpha$ and $\mathrm{Var}(\boldsymbol{S}_j) = 0 \pm \alpha$,
- **Active variable**: $\bar{\boldsymbol{S}}_j = 0 \pm \alpha$ and $\mathrm{Var}(\boldsymbol{S}_j) = 0 \pm \alpha$,
- **Mixed variable**: any variable not classified as active or passive.

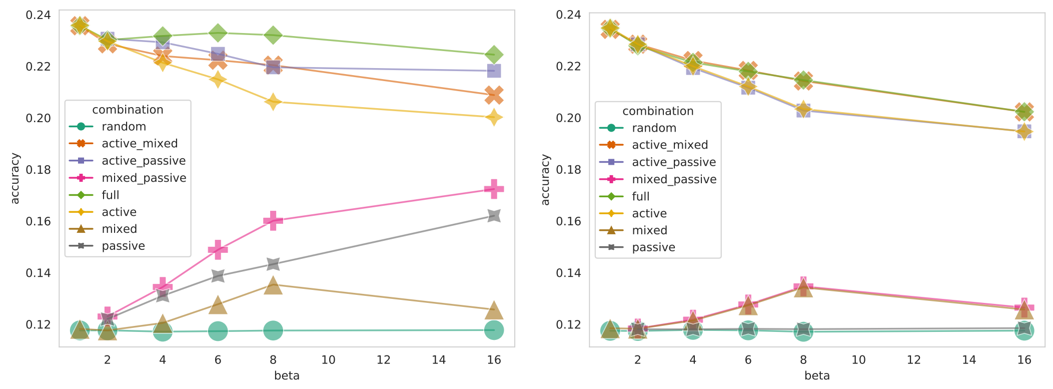
In all our experiments, α is set to 0.1. As stated above, any variable that does not satisfy both conditions required for active or passive variables becomes a mixed variable.

Note that the indexes obtained with this method are used in the same way to identify the variable types of mean and sampled representations. For example, if our procedure determines that index $j = 1$ of the variance representation corresponds to a passive variable, the variable at index $j = 1$ of the mean and sampled representations will be considered as passive.

Sanity check To verify that our thresholds α are valid and that the variable types have been mapped correctly, we compare the scores of a classifier trained on the whole representations with those of classifiers trained on every combination of variable types. Similarly to Locatello et al. (2019a), the classification models are logistic regressions that predict the labels of the dataset from which the representation was learned.

We trained the logistic regressions for 300000 steps on 10000 data examples for each variable combination and computed the average accuracy over 5000 test examples. The logistic regressions are cross-validated with 10 different regularisation strengths and 5 folds.

If our procedure to identify active, mixed and passive variables is correct, we should expect the following. Active variables should contribute the most to the predictions and give results close to those obtained by the full representation. Mixed variables should contribute less and have a much lower score but together with the active representation, they should provide the same score as the full representation. Passive variables should not contribute at all and ought to provide results close to a random classifier.



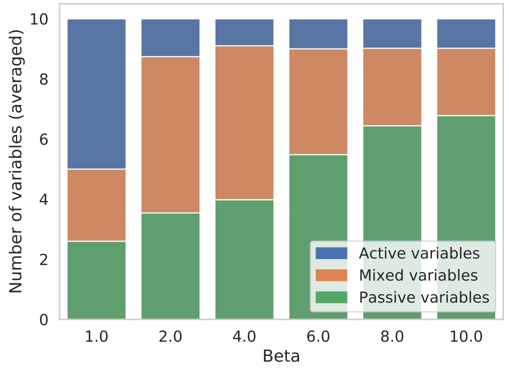
(a) Mean representation (b) Sampled representation

Figure 6.4: Average test accuracy of a logistic regression trained on the mean and sampled representations learned by a β-VAE trained on dSprites. Each figure shows the results obtained using the full representations and combinations of different variable types. This is also compared to a random classifier picking uniformly from the possible labels.

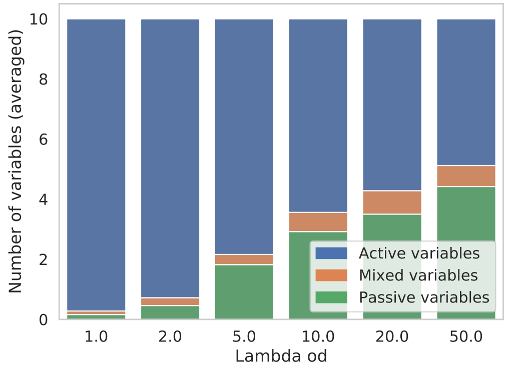
In Figure 6.4b, we can see that the results obtained for β-VAE trained on dSprites using sampled representations are exactly as expected. The passive variables gave equivalent results to a random classifier, while mixed variables performed slightly better, which indicates that we correctly identify them. The score obtained with active and mixed representations is the same as the full score in sampled representations, while active variables alone or with passive variables performed a bit worse, which confirms that mixed and active variables have also been identified correctly.

Interestingly, the results obtained for mean representations in Figure 6.4a show that passive variables are becoming more informative as the regularisation strength increases. While the score obtained with active variables is still closer to the full representation score, we can see that in opposition to sampled representations, passive and active variables perform better than mixed and active variables. Thus, *despite being close to zero and having a very low variance, passive variables of the mean representation seem to capture some information about the data*. Note that this result does not indicate a problem in the detection of the passive variables: if we had incorrectly identified any active or mixed variables as passive, they would still convey some information in the sampled representation. As seen in Figure 6.4b, it is not the case here as their performance is equivalent to the performance of a random classifier.

As discussed in Section 6.3, we can also see in Figure 6.5 that, as expected, the number of passive and mixed variables increases with the regularisation strength. The number of passive variables at the lowest regularisation strength is generally close to zero for all datasets and all models, except annealed-VAE, whose special case will be discussed in Appendix B.1.



(a) β-TC VAE trained on smallNorb.



(b) DIP-VAE II trained on dSprites.

Figure 6.5: Number of passive, mixed and active variables with increased regularisation strength averaged over 50 runs for each regularisation value.

To summarise, given that the behaviour of the passive and active variables observed in Figures 6.4 and 6.5 is consistent with the findings of Rolinek, Zietlow and Martius (2019) and Dai, Wang and Wipf (2020) regarding the polarised regime, we assume that our method to determine the type of variables is valid, and our thresholds properly set. We can thus proceed to the next steps of our experiment.

6.4.2 *Metrics used to assess mean and sampled representations*

To be consistent with the existing literature (Locatello et al., 2019a), we compared the mean and sampled representations using TC and averaged MI scores. We also used effective rank (Roy and Vetterli, 2007) as a complementary measure. Those metrics are measuring slightly different things and have different limitations that we will detail below.

6.4.2.1 *Total correlation*

The TC (Watanabe, 1960) is a measure of the amount of information shared between multiple latent variables. It is measured as the KLD between the joint distribution and the product of its marginal distributions. More formally, given a latent representation $\mathbf{r} = \mathbf{r}(\mathbf{x}; \phi)$:

$$TC(\mathbf{r}) = D_{\text{KL}}\left(p(\mathbf{r}) \,\Big\|\, \prod_{j=1}^{d} p(\mathbf{r}_j)\right). \tag{6.1}$$

In their experiment, Locatello et al. (2019a) assumed that $\mathbf{r}$ (either the mean or sampled representation) was multivariate Gaussian with a mean of zero. Thus, given the mean $\bar{\mathbf{r}}$ and covariance $\text{Cov}[\mathbf{r}]$ of $\mathbf{r}$, Equation 6.1 takes the following form:

$$TC(\mathbf{r}) = D_{\text{KL}}\left(\mathcal{N}(\bar{\mathbf{r}}, \text{Cov}[\mathbf{r}]) \,\Big\|\, \prod_{j=1}^{d} \mathcal{N}(\bar{\mathbf{r}}_j, \text{Cov}[\mathbf{r}]_{jj})\right), \tag{6.2}$$

We assume that all the distributions have a mean of zero. Thus,

$$TC(\mathbf{r}) = D_{\text{KL}}\left(\mathcal{N}(\mathbf{0}, \text{Cov}[\mathbf{r}]) \,\Big\|\, \prod_{j=1}^{d} \mathcal{N}(0, \text{Cov}[\mathbf{r}]_{jj})\right).$$

Moreover, the second term of the KLD represents the case where $\mathbf{r}$ is composed of j independent and Gaussian random variables. We can thus re-express it as a multivariate Gaussian distribution with diagonal covariance,

$$TC(\mathbf{r}) = D_{\text{KL}}\Big(\mathcal{N}(\mathbf{0}, \text{Cov}[\mathbf{r}]) \,\|\, \mathcal{N}(\mathbf{0}, \text{diag}[\text{Var}(\mathbf{r})])\Big).$$

Now, let us define $\boldsymbol{\Sigma} \triangleq \text{Cov}[\mathbf{r}]$ and $\bar{\boldsymbol{\Sigma}} \triangleq \text{diag}[\text{Var}(\mathbf{r})]$. Using Equation 2.18, we have:

$$TC(\mathbf{r}) = \frac{1}{2}\left(\log \frac{\det(\bar{\boldsymbol{\Sigma}})}{\det(\boldsymbol{\Sigma})} + \text{Tr}(\bar{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\Sigma}) - n\right).$$

Because $\mathbf{\Sigma}$ and $\bar{\mathbf{\Sigma}}$ have the same diagonal values, $\bar{\mathbf{\Sigma}}^{-1}\mathbf{\Sigma} = \boldsymbol{I}$ and $\text{Tr}(\bar{\mathbf{\Sigma}}^{-1}\mathbf{\Sigma}) = n$. Thus,

$$TC(\mathbf{r}) = \frac{1}{2}\left(\log\det(\bar{\mathbf{\Sigma}}) - \log\det(\mathbf{\Sigma})\right).$$

As $\bar{\mathbf{\Sigma}}$ is diagonal, this further simplifies to

$$\begin{aligned} TC(\mathbf{r}) &= \frac{1}{2}\left(\log\prod_{j=1}^{d}\bar{\mathbf{\Sigma}}_{jj} - \log\det(\mathbf{\Sigma})\right), \\ &= \frac{1}{2}\left(\sum_{j=1}^{d}\log\bar{\mathbf{\Sigma}}_{jj} - \log\det(\mathbf{\Sigma})\right), \\ &= \frac{1}{2}\left(\sum_{j=1}^{d}\log\text{Cov}[\mathbf{r}]_{jj} - \log\det(\text{Cov}[\mathbf{r}])\right). \end{aligned} \tag{6.3}$$

From Equation 6.3, we then obtain Theorem 6.4.1.

Theorem 6.4.1 (Impact of passive variables on $TC(\mathbf{z})$)**.** *The* TC *of sampled representations is not modified by passive variables.*

Proof. Let us consider a sampled representation $\mathbf{z}$ with n latent variables having a covariance matrix $\text{Cov}[\mathbf{z}] \in \mathbb{R}^{n\times n}$. Now, let us create a second sampled representation $\hat{\mathbf{z}}$ by concatenating the latent variables of $\mathbf{z}$ with m passive variables. The resulting covariance matrix $\text{Cov}[\hat{\mathbf{z}}]$ can be partitioned as

$$\text{Cov}[\hat{\mathbf{z}}] = \begin{bmatrix} \text{Cov}[\mathbf{z}] & \mathbf{0}_{n,m} \\ \mathbf{0}_{m,n} & \boldsymbol{I}_{m,m} \end{bmatrix}.$$

From this, we can immediately see that

$$\begin{aligned} \sum_{i=1}^{n+m}(\log\text{Cov}[\hat{\mathbf{z}}]_{ii}) &= \sum_{i=1}^{n}(\log\text{Cov}[\mathbf{z}]_{ii}) + \sum_{i=1}^{m}(\log\boldsymbol{I}_{ii}) \\ &= \sum_{i=1}^{n}(\log\text{Cov}[\mathbf{z}]_{ii}). \end{aligned} \tag{6.4}$$

Moreover, as $\text{Cov}[\mathbf{z}]$ is invertible, using Schur's identity (Brualdi and Schneider, 1983):

$$\begin{aligned} \det(\text{Cov}[\hat{\mathbf{z}}]) &= \det(\text{Cov}[\mathbf{z}])\det(\boldsymbol{I} - \mathbf{0}_{m,n}\text{Cov}[\mathbf{z}]^{-1}\mathbf{0}_{n,m}) \\ &= \det(\text{Cov}[\mathbf{z}])\det(\boldsymbol{I}) \\ &= \det(\text{Cov}[\mathbf{z}]). \end{aligned}$$

Thus,

$$\log\det(\mathrm{Cov}[\hat{\mathbf{z}}]) = \log\det(\mathrm{Cov}[\mathbf{z}]). \tag{6.5}$$

Recall from Equation 6.3 that

$$TC(\hat{\mathbf{z}}) = \frac{1}{2}\left(\sum_{i=1}^{n+m}(\log\mathrm{Cov}[\hat{\mathbf{z}}]_{ii}) - \log\det(\mathrm{Cov}[\hat{\mathbf{z}}])\right). \tag{6.6}$$

Using the result of Equation 6.4, we can replace the first term of Equation 6.6, so that

$$TC(\hat{\mathbf{z}}) = \frac{1}{2}\left(\sum_{i=1}^{n}(\log\mathrm{Cov}[\mathbf{z}]_{ii}) - \log\det(\mathrm{Cov}[\hat{\mathbf{z}}])\right). \tag{6.7}$$

Finally, we can replace the second term of Equation 6.7 using Equation 6.5 to obtain

$$\begin{aligned} TC(\hat{\mathbf{z}}) &= \frac{1}{2}\left(\sum_{i=1}^{n}(\log\mathrm{Cov}[\mathbf{z}]_{ii}) - \log\det(\mathrm{Cov}[\mathbf{z}])\right), \\ &= TC(\mathbf{z}), \end{aligned}$$

as required. □

Limitations According to Theorem 6.3.1, while the passive variables of the sampled representations follow standard Gaussian distributions, other variables of mean and sampled representations, especially mixed variables, may not be Gaussian. Moreover, the distributions of the latent variables may not be jointly Gaussian. As such, TC may provide inaccurate results. For this reason, it is important to use complementary metrics.

6.4.2.2 *Averaged mutual information*

MI is a measure of the information shared by two latent variables (Cover, 1999). Similarly to TC, which is a generalisation of MI to the multivariate case, it is measured as the KLD between the joint distribution and the product of its marginal distributions. That is, given two latent factors $\mathbf{r}_1$ and $\mathbf{r}_2$:

$$MI(\mathbf{r}_1,\mathbf{r}_2) = D_{\mathrm{KL}}\big(q(\mathbf{r}_1,\mathbf{r}_2)\parallel q(\mathbf{r}_1)q(\mathbf{r}_2)\big). \tag{6.8}$$

In their experiment, Locatello et al. (2019a) calculated the MI over discretised values using:

$$MI(\mathbf{r}_1,\mathbf{r}_2) = \sum_{i=1}^{|\boldsymbol{U}|}\sum_{j=1}^{|\boldsymbol{V}|}\frac{|\boldsymbol{U}_i\cap\boldsymbol{V}_j|}{n}\log\frac{n|\boldsymbol{U}_i\cap\boldsymbol{V}_j|}{|\boldsymbol{U}_i||\boldsymbol{V}_j|}, \tag{6.9}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ are the bins of $\mathbf{r}_1$ and $\mathbf{r}_2$ respectively, n is the number of samples, and $|\cdot|$ denotes the cardinality. They then used the averaged MI over all the latent factors:

$$MI_{avg}(\mathbf{r}) = \frac{1}{k^2 - k} \sum_{i=1}^{k} \sum_{j \neq i} MI(\mathbf{r}_i, \mathbf{r}_j), \tag{6.10}$$

where k is the dimensionality of the latent representation $\mathbf{r}$. Theorem 6.4.2 follows.

Theorem 6.4.2 (Impact of passive variables on $MI_{avg}(\mathbf{z})$)**.** *The averaged* MI *of sampled representations decreases with the number of passive variables.*

Proof. Let us consider a sampled representation $\mathbf{z} \in \mathbb{R}^2$ composed of two active variables $\mathbf{z}_1$ and $\mathbf{z}_2$ such that $MI(\mathbf{z}_1, \mathbf{z}_2) = c$ with $c > 0$. As MI is symmetric, we have $MI_{avg}(\mathbf{z}) = \frac{1}{2}2c = c$.
Now, let us consider a sampled representation $\hat{\mathbf{z}} \in \mathbb{R}^n$ composed of the two actives variables of $\mathbf{z}$ and $n-2$ additional passive variables $\{\mathbf{z}_j\}_{j=3}^{n}$.
Because passive variables do not contain any information about the input, the MI between an active variable i and a passive variable j will be zero (i.e., $MI(\mathbf{z}_i, \mathbf{z}_j) = 0$). Passive variables are also independent and normally distributed, hence the MI between two different passive variables i and j will also be zero. We thus have

$$\begin{aligned} MI_{avg}(\hat{\mathbf{z}}) =& \frac{1}{n^2 - n} \left(\sum_{i \in \mathbb{V}_a} \sum_{\substack{j \in \mathbb{V}_a \\ i \neq j}} MI(\mathbf{z}_i, \mathbf{z}_j) + 2 \sum_{i \in \mathbb{V}_a} \sum_{j \in \mathbb{V}_p} MI(\mathbf{z}_i, \mathbf{z}_j) \right. \\ & \left. + \sum_{i \in \mathbb{V}_p} \sum_{\substack{j \in \mathbb{V}_p \\ i \neq j}} MI(\mathbf{z}_i, \mathbf{z}_j) \right), \\ =& \frac{1}{n^2 - n} 2c < MI_{avg}(\mathbf{z}), \end{aligned}$$

as required. □

Limitations As it is using discretised values, averaged MI does not have the downside of TC regarding the Gaussian assumption. However, because MI is averaged, it may diminish the strong relationships between two variables if the other MI scores are close to zero, as seen in Theorem 6.4.2.

6.4.2.3 *Effective rank*

The effective rank of a matrix (Roy and Vetterli, 2007) is a real-valued generalisation of the integer-valued rank of a matrix. More formally, let us consider a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$. Its singular value decomposition is $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$ where $\boldsymbol{U} \in \mathbb{R}^{m \times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and $\boldsymbol{S} \in \mathbb{R}^{m \times n}$ is a rectangular diagonal matrix containing the l singular values $s_1 \geqslant s_2 \geqslant \cdots \geqslant s_l \geqslant 0$ where $l = \min(m, n)$. The singular value distribution is given by:

$$p_k = \frac{s_k}{\sum_{i=1}^{l} s_i} \text{ for } k = 1, 2, \cdots, l.$$

The effective rank is then defined as:

$$\text{erank}(\boldsymbol{A}) = \exp\big(\mathrm{H}(p_1, p_2, \cdots, p_l)\big),$$

where $\mathrm{H}(p_1, p_2, \cdots, p_l)$ is the spectral entropy (Campbell, 1960; Yang, Gibson and He, 2005):

$$\mathrm{H}(p_1, p_2, \cdots, p_l) = -\sum_{k=1}^{l} p_k \log p_k.$$

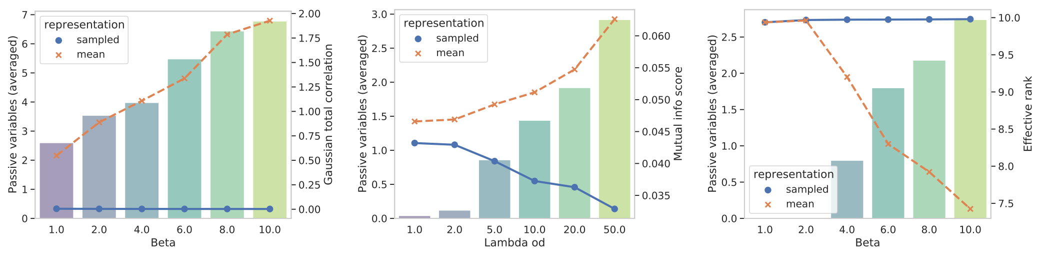
The effective rank is generally more informative than the rank as it can take all possible values in the interval $[1, l]$, whereas the rank is limited to integer values in the set $\{1, 2, ..., l\}$. Consider, for example, a Gaussian distribution of dimension two whose variables are highly correlated. The first singular value will dominate while the second will be close to zero. As both singular values are higher than zero, the matrix rank will be two. However, because the second singular value is very low, its effective rank will be only slightly above one. Overall, the effective rank tells us more about the data than the integer-valued rank.

Limitations While the effective rank does not have the weaknesses of TC regarding the Gaussian assumption nor of averaged MI regarding the dilution of a strong relationship when dimensionality is high, it would not make sense to measure it separately for each type of variables as the significance of the dimensions would be relative to the subset considered. Thus, effective rank is only included in Section 6.4.3, where we use it to analyse all variables jointly.

6.4.3 *Preliminary observations*

As we have established in Section 6.3, the active variables of mean and sampled representations should be similarly correlated. Hence, while Locatello et al. (2019a) concluded that uncorrelated sampled representations did not guarantee uncorrelated mean representations, we argue that active variables, which encode the most information, should have similar correlation in mean and sampled representations. We thus suggest that the increased correlation of mean representations may be due to mixed and passive variables. If the passive variables are, as hypothesised, responsible for the higher correlation of mean representations, one should expect the effective rank to be close to the total number of latent variables minus the passive ones.

Our preliminary observations are consistent with this hypothesis. Specifically, in sampled representations, Figures 6.6a and 6.6b show that TC does not change when we increased the number of passive variables and MI decreases, as described in Theorems 6.4.1 and 6.4.2. Moreover, Figure 6.6a shows that the discrepancies between the TC scores of the mean and sampled representations are increasing with the number of passive variables, and we can observe the same trend for MI in Figure 6.6b. The effective rank of mean representations is, as shown in Figure 6.6c, close to the total number of latent variables minus the passive ones. One can also notice that in sampled representations, the effective rank is close to the total number of variables as the passive variables are replaced by uncorrelated samples from $\mathcal{N}(0,1)$, and those uncorrelated variables cannot reduce the effective rank any more. Interestingly, in addition to showing an increased correlation of passive variables, the effective rank is also providing further confirmation that passive variables are correctly identified in Proposition 6.3.2 and Section 6.4.1.



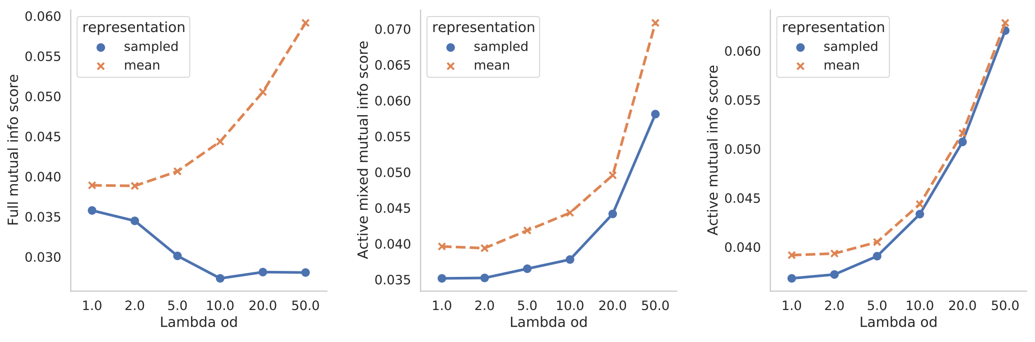
(a) Total correlation (b) Mutual information (c) Effective rank

Figure 6.6: Relationship between the number of passive variables and the TC, MI, and effective rank scores of mean and sampled representations. Figures (a), (b) and (c) use the representations learned by β-TC VAE trained on smallNorb, DIP-VAE II trained on Cars3D, and β-TC VAE trained on colour dSprites, respectively. Lines indicate the metric scores of the representation and the bars the average number of passive variables.

6.4.4 *Impact of passive and mixed variables on averaged MI and total correlation*

To further validate our hypothesis that only mixed and passive variables are responsible for the increased correlation of mean representations, we will compare TC and averaged MI scores with and without mixed and passive variables. Thus, we will be able to determine whether we can have an increased correlation between any dimensions of the mean representations, as initially inferred by Locatello et al. (2019a), or only between a specific subset corresponding to mixed and passive variables.

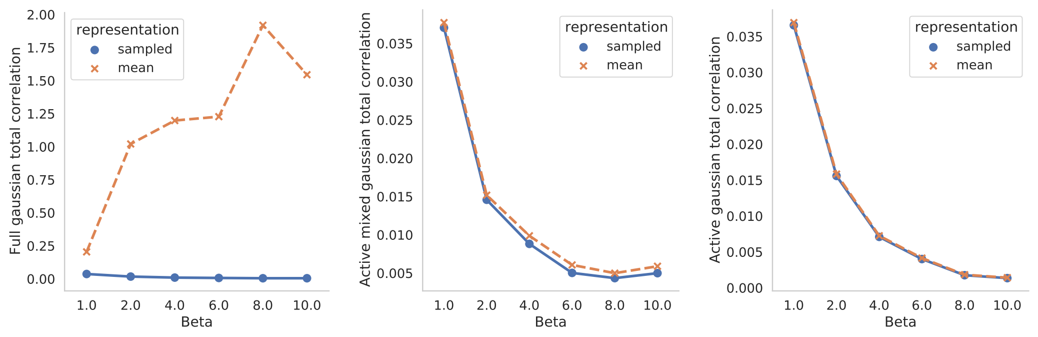
As we have mapped each index of the mean and sampled representation to a specific variable type in Section 6.4.1, we can now assess the impact of mixed and passive variables on TC and averaged MI by comparing the scores with and without them. We first calculate TC and averaged MI using the full representation and compare these scores with those obtained without passive variables and with only active variables.



(a) Full representation (b) Mixed and active variables (c) Active variables

Figure 6.7: Comparison of the averaged MI scores of the mean and sampled representations of DIP-VAE II trained on dSprites. In Figure (a), the number of passive variables increases with λ leading to lower averaged MI scores for sampled representations. The mean and sampled representation scores become more similar once passive variables are removed in Figures (b) and (c).

Mutual information and total correlation of mean representations In Figures 6.7a and 6.8a, we can see that the mixed and passive variables are raising the TC and averaged MI scores of the mean representations (the dashed orange curves). Indeed, when these variables are removed, the TC and averaged MI scores of the mean and sampled representations are quite similar, as observed in Figures 6.7b and 6.7c, and Figures 6.8b and 6.8c. While the mixed variables impact the score to a small extent, the passive variables lead to a dramatic score increase, especially for TC. These observations, thus, show that *active variables are as disentangled in mean as in sampled representations, and passive variables of the mean representations should have*



(a) Full representation (b) Mixed and active variables (c) Active variables

Figure 6.8: Comparison of the TC scores of the mean and sampled representations of β-TC VAE trained on noisy dSprites. In Figure (a), the number of passive variables increases with β leading to lower TC scores for sampled representations. The mean and sampled representation scores become more similar once passive variables are removed in Figures (b) and (c).

strong correlations with other variables. We, therefore, examine the correlation of passive variables in Section 6.5 to better understand with which type of variable they are correlated and how this correlation emerges. Even before we present this analysis, we should recall that passive variables of the mean representations performed better on downstream tasks than their sampled counterpart, as observed in Section 6.4.1. Thus, to convey useful information, the passive variables have to be correlated with known informative variables. As a result, one should expect the passive variables in the mean representations to be correlated with active ones. This is the subject of our investigation in Section 6.5.

6.5 WHERE DOES THE CORRELATION OF PASSIVE VARIABLES IN THE MEAN REPRESENTATION COME FROM?

As the passive variables of mean representations lead to higher TC and averaged MI scores, they should exhibit some correlation with other variables. This can be seen empirically in Figure 6.9 where the correlation between each latent variable of a β-VAE trained with $\beta = 16$ on colour dSprites shows that, indeed, the passive variables tend to have strong correlation scores with one or more variables, which is generally not the case for other variable types. This result should not be surprising as VAEs optimise passive variables to be close to zero with low variance, but not to be uncorrelated. However, one can ponder on the origin of these correlations. Are they present from the beginning of the learning process and lead these variables to become passive, or are the correlations the consequence of the variables being passive?

To gain some insights into this question, we trained a β-VAE with $\beta = 8$ on dSprites for 300K steps and saved a snapshot of the model parameters every 1000 steps and observed the evolution of the latent representations. Using the same technique as in Section 6.4, after the model has been trained (i.e. after 300K training steps), we determined that the variables 1, 4, and 6 were passive, the variables 0, 3 and 8 mixed, and the remaining ones active. In Figure 6.10, we can see that the correlation score of the passive variables is more often above 0.2 than the scores of the active variables across the 300 snapshots recorded during the training process. This highlights the important correlation scores of the passive variables in most of the training steps. Figure 6.11 shows that the correlation between passive variables and other variables varies significantly during the training process and can be relatively high, while the correlation scores of active variables remain very low. These high correlations are consistent with the observations of Section 6.4.1 where the logistic regression had better accuracy with passive variables than with mixed ones. While a more in-depth study of the learning dynamics of VAEs would be needed to provide a complete explanation of this phenomenon, the frequent changes of the correlation scores makes it likely to be an inherent property of the neural network training process.

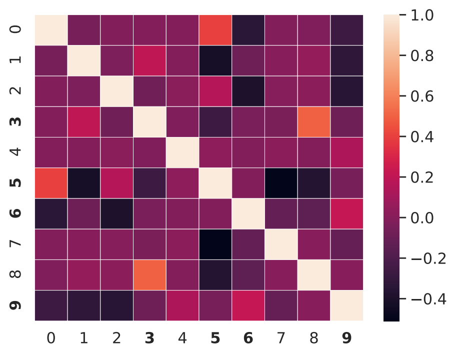


Figure 6.9: Correlation of the passive variables of β-VAE trained on colour dSprites with $\beta = 16$ with other variables. The passive variables are at indexes 3, 5, 6, and 9. Several correlations of the passive variables clearly stand out.

6.6 CONCLUSION

In their study, Locatello et al. (2019a) have reported that mean representations seemed to be more correlated than sampled ones in a large number of experiments. They concluded that enforcing uncorrelated sampled representations may not be sufficient to obtain uncorrelated mean representations. By extending the definition of the po-

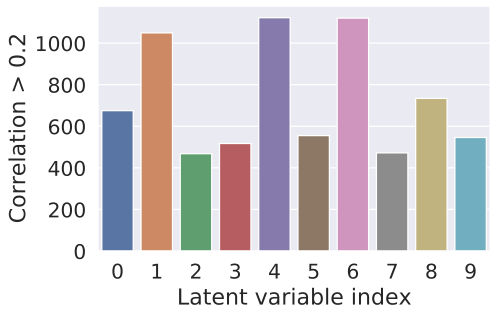


Figure 6.10: Number of times where the absolute value of the correlation of each latent variable with another variable in the mean representation was above 0.2 in the 300 snapshots recorded during the training of a β-VAE with $\beta = 8$ on dSprites. The passive variables are at indexes 1, 4, and 6, and are the most often correlated variables.

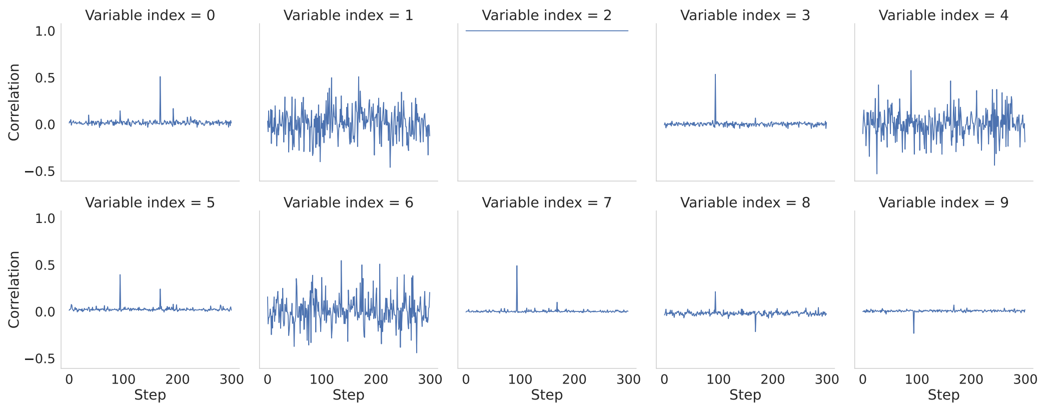


Figure 6.11: The correlation scores of the active variable at index 2 of the mean representation with all the other variables during the 300K training steps of a β-VAE with $\beta = 8$ trained on dSprites. We can see an increased correlation with all the passive variables (indexes 1, 4, and 6).

larised regime to mean representations, we have shown that while aiming to optimise uncorrelated sampled representations is not sufficient to guarantee completely uncorrelated mean representations, it is sufficient to obtain uncorrelated *active variables* in mean representations. We thus hypothesised that the increasing discrepancies between the two representations should only be attributed to a subset of variables: the passive ones. This hypothesis was consistent across different levels of regularisation and has been further confirmed by empirical observations showing an increased correlation of passive variables. By considering the latent representations over the whole dataset, we have also introduced mixed variables, a type of variable that can either be active or passive depending on the input example provided. Our empirical results confirmed the existence and importance of such variables.

Should we use mean representations for downstream tasks? One of the concerns that was raised by the findings of Locatello et al. (2019a) was that mean representations, which are generally used for downstream tasks, would not benefit from the disentanglement that was exhibited by sampled representations. However, we showed that active variables, the relevant part of the representations, are quite similar in both representations. Thus, we can expect mean representations to be as useful as sampled ones for downstream tasks. We established that the passive variables of mean representations are near zero with low variance and seem to have an arbitrary high correlation with other variables. Thus, one may want to remove them, especially when feeding the mean representations into algorithms that are sensitive to near-zero or highly correlated features.

Generalisation of our results to other types of VAEs Because this chapter explains the reason of the discrepancies between the mean and sampled representations observed by Locatello et al. (2019a), we chose to remain as close as possible to their experimental protocol, and used the same models to obtain consistent results. However, the authors proved that unsupervised disentanglement learning was not possible without further inductive biases. Thus one should keep in mind that disentangled representations obtained from the models used in this chapter can only be selected post-hoc based on disentanglement metrics, which may not always be practical for downstream task applications. Despite this, as described in Section 6.3, our explanation mainly relies on the polarised regime, which, as discussed in Chapters 4 and 5, occurs in any well-behaved VAE as long as their prior and posterior distributions are Gaussian with diagonal covariances. With slight updates to our filtering method, this makes our findings applicable to iVAEs which can directly provide disentangled representations (Khemakhem et al., 2020; Mita, Filippone and Michiardi, 2021).

Other applications of this study While our main focus was to explain the higher correlation observed in mean representations by Locatello et al. (2019a), our new definitions of the polarised regime may also be useful to monitor the number of passive variables and prevent posterior collapse due to over-regularisation (Lucas et al., 2019a; Dai, Wang and Wipf, 2020).

The surprising behaviour of annealed VAEs discussed in Section B.1 and the correlation of passive variables reported in Section 6.5 show that we could gain a deeper understanding of the representations learned by VAEs by studying their learning dynamics more in depth. This will be the topic of our next chapter, where we will study the similarity and generalisability of the representations learned by different VAEs.

7

SIMILARITY OF THE LATENT REPRESENTATIONS

7.1 INTRODUCTION

We have seen in Chapter 6 that while learning in a polarised regime, VAEs with different learning objectives did not always display similar numbers of active and passive variables. This was especially visible on annealed VAE, as described in Chapter 6 and Appendix B.1.
To better understand this phenomenon, we propose to explore the representational similarity of VAEs. The domain of deep representational similarity is an active area of research and metrics such as Singular Value Canonical Correlation Analysis (SVCCA) (Raghu et al., 2017; Morcos, Raghu and Bengio, 2018), Procrustes distance (Schönemann, 1966), or CKA (Kornblith et al., 2019) have already proven very useful in analysing the learning dynamics of various models (Wang et al., 2019; Kudugunta et al., 2019; Raghu et al., 2019; Neyshabur, Sedghi and Zhang, 2020). Such metrics could help identify common representations between models, which could in turn explain in which context different models do, or do not, exhibit the same number of active and passive variables.

In this chapter, our aim is to use such representational similarity techniques to analyse the representations learned by VAEs on different datasets, and use these results to explain the fluctuating number of active and passive variables observed when using different learning objectives in Chapter 6. Specifically, based on the results of CKA, we will show that the representations learned by encoders across a range of datasets are generic across seeds but may or may not diverge between learning objectives. We will further discuss the implications of these findings from the polarised regime point of view and show that different models whose mean representation retains a high representational similarity have the same number of active variables, while this is not guaranteed when the similarity decreases.

Our contributions are as follows:

1. We verify the consistency of CKA for measuring the representational similarity of VAEs by demonstrating that the similarity scores agree with several known properties of VAEs.
2. Based on these insights, we discuss the implications of high and low similarity scores for active and passive variables.

7.2 BACKGROUND

As stated in Section 7.1, our aim is to study the impact of the learning dynamics of VAEs on the latent representations learned under the polarised regime. In this section, we will thus present two well-established metrics that will be used in our experiment. Representational similarity metrics aim to compare the geometric similarity between two representations. In the context of deep learning, these representations correspond to $\mathbb{R}^{n \times p}$ matrices of activations, where n is the number of data examples and p the number of neurons in a layer. Such metrics can provide various information on deep neural networks (e.g., the training dynamics of neural networks, common and specialised layers between models).

7.2.1 *Representational similarity*

Centred Kernel Alignment We have seen in Section 2.5 that CKA measures the alignment between the $n \times n$ kernel matrices of two representations, and works well with linear kernels (Kornblith et al., 2019) for representational similarity of centred layer activations. We thus focus on the linear CKA, also known as RV coefficient (Escoufier, 1973; Robert and Escoufier, 1976) as defined in Equation 2.2. For conciseness, we will refer to linear CKA as CKA in the rest of this chapter. As CKA is invariant to any rotation $\boldsymbol{R}$ and scaling ξ, given two embedding functions E_ϕ and $E_{\bar{\phi}}$, we have $CKA(E_\phi(\boldsymbol{X}), E_{\bar{\phi}}(\boldsymbol{X})) = CKA(E_\phi(\boldsymbol{X}), \xi \boldsymbol{R} E_{\bar{\phi}}(\boldsymbol{X}))$. Thus, when the similarity score between the two embeddings is high, we have $E_\phi(\boldsymbol{X}) \approx E_{\bar{\phi}}(\boldsymbol{X})$ up to arbitrary rotation and scaling. Given that rotation and uniform scaling are angle-preserving transformations, a high CKA score also implies that angles between latent representations are preserved across E_ϕ and $E_{\bar{\phi}}$. Specifically, given the latent representations $\mathbf{e}^{(i)} \triangleq E_\phi\big(\mathbf{x}^{(i)}\big)$, and $\bar{\mathbf{e}}^{(i)} \triangleq E_{\bar{\phi}}\big(\mathbf{x}^{(i)}\big)$ we have

$$\angle\big(\mathbf{e}^{(i)}, \mathbf{e}^{(j)}\big) \approx \angle\big(\bar{\mathbf{e}}^{(i)}, \bar{\mathbf{e}}^{(j)}\big). \tag{7.1}$$

Orthogonal Procrustes The aim of orthogonal Procrustes (Schönemann, 1966) is to align a matrix $\boldsymbol{Y}$ to a matrix $\boldsymbol{X}$ using orthogonal transformations $\boldsymbol{Q}$ such that

$$\begin{aligned} \min_{\boldsymbol{Q}} \quad & \|\boldsymbol{X} - \boldsymbol{Y}\boldsymbol{Q}\|_F^2 \\ \text{s.t.} \quad & \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}. \end{aligned} \tag{7.2}$$

This can be reformulated as

$$\begin{aligned} \|\boldsymbol{X} - \boldsymbol{Y}\boldsymbol{Q}\|_F^2 &= \sum_i \|\boldsymbol{X}_{:,i} - (\boldsymbol{Y}\boldsymbol{Q})_{:,i}\|_2^2, \\ &= \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\boldsymbol{Q}\|_F^2 - 2\sum_i (\boldsymbol{Q}^T\boldsymbol{Y}^T\boldsymbol{X})_{ii}, \\ &= \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\boldsymbol{Q}\|_F^2 - 2\,\mathrm{Tr}(\boldsymbol{Q}^T\boldsymbol{Y}^T\boldsymbol{X}). \quad (7.3) \\ & \quad (7.4) \end{aligned}$$

As described in Golub and Van Loan (2013, pp. 327-328), to minimise Equation 7.3, we simply need to maximise $\mathrm{Tr}(\boldsymbol{Q}^T\boldsymbol{Y}^T\boldsymbol{X})$ while enforcing $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$. Hence, given the SVD of $\boldsymbol{Y}^T\boldsymbol{X} \triangleq \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T$, and using the cyclic property of the trace we have

$$\begin{aligned} \mathrm{Tr}(\boldsymbol{Q}^T\boldsymbol{Y}^T\boldsymbol{X}) &= \mathrm{Tr}(\boldsymbol{V}^T\boldsymbol{Q}^T\boldsymbol{U}\boldsymbol{\Lambda}), \\ &\leqslant \sum_i \boldsymbol{\Lambda}_{ii}, \\ &= \|\boldsymbol{Y}^T\boldsymbol{X}\|_*, \end{aligned} \tag{7.5}$$

where $\|\cdot\|_*$ is the nuclear norm. We can see that the upper bound is reached when $\boldsymbol{V}^T\boldsymbol{Q}^T\boldsymbol{U} = \boldsymbol{I}$, that is, when $\boldsymbol{Q}^T = \boldsymbol{V}\boldsymbol{U}^T$. The orthogonality constraint on $\boldsymbol{Q}$ is also met as $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{V}\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{V}^T = \boldsymbol{I}$.

Thus, the Procrustes distance, P_d, is the difference remaining between $\boldsymbol{X}$ and $\boldsymbol{Y}$ when $\boldsymbol{Q}$ is optimal,

$$P_d(\boldsymbol{X}, \boldsymbol{Y}) = \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\|_F^2 - 2\|\boldsymbol{Y}^T\boldsymbol{X}\|_*. \tag{7.6}$$

To easily compare the results of Equation 7.6 with CKA, we first bound its results between 0 and 2 using normalised $\dot{\boldsymbol{X}}$ and $\dot{\boldsymbol{Y}}$.

The normalisation process is similar to Ding, Denain and Steinhardt (2021). Given an activation matrix $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ containing n samples and m features, we compute the vector $\bar{\boldsymbol{x}} \in \mathbb{R}^m$ containing the mean values of the columns of $\boldsymbol{X}$. Using the outer product $\otimes$, we get $\bar{\boldsymbol{X}} = \mathbf{1}_n \otimes \bar{\boldsymbol{x}}$, where $\mathbf{1}_n \in \mathbb{R}^n$ is a vector of ones and $\bar{\boldsymbol{X}} \in \mathbb{R}^{n \times m}$. We then normalise $\boldsymbol{X}$ such that

$$\dot{\boldsymbol{X}} = \frac{\boldsymbol{X} - \bar{\boldsymbol{X}}}{\|\boldsymbol{X} - \bar{\boldsymbol{X}}\|_F}. \tag{7.7}$$

As the Frobenius norm of $\dot{\boldsymbol{X}}$ and $\dot{\boldsymbol{Y}}$ is 1, and $\|\dot{\boldsymbol{Y}}^T \dot{\boldsymbol{X}}\|_*$ is always positive (1 when $\dot{\boldsymbol{X}} = \dot{\boldsymbol{Y}}$, smaller otherwise), and Equation 7.6 lies in $[0, 2]$.

Then, we transform the result to a similarity metric ranging from 0 (not similar) to 1 ($\boldsymbol{X} = \boldsymbol{Y}$),

$$P_s(\boldsymbol{X}, \boldsymbol{Y}) = 1 - \frac{1}{2}\left(\|\dot{\boldsymbol{X}}\|_F^2 + \|\dot{\boldsymbol{Y}}\|_F^2 - 2\|\dot{\boldsymbol{Y}}^T \dot{\boldsymbol{X}}\|_*\right). \tag{7.8}$$

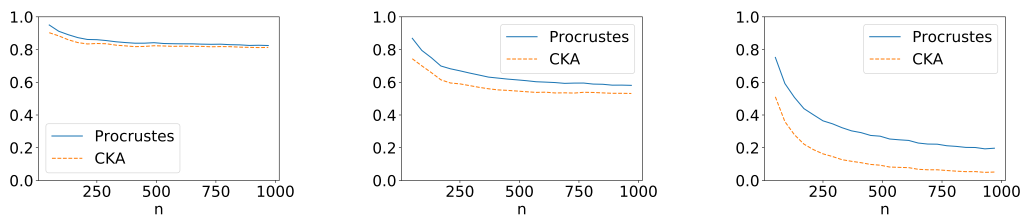
We will refer to Equation 7.8 as Procrustes similarity in the following sections.

7.2.2 *Limitations of CKA and Procrustes similarities*

While CKA and Procrustes lead to accurate results in practice, they suffer from some limitations that need to be taken into account in our study. Before we discuss these limitations, we should clarify that, in the rest of this chapter, $sim(\cdot, \cdot)$ represents a similarity metric in general, while $CKA(\cdot, \cdot)$ and $P_s(\cdot, \cdot)$ specifically refer to CKA and Procrustes similarities.

Sensitivity to architectures Maheswaranathan et al. (2019) have shown that similarity metrics comparing the geometry of representations were overly sensitive to differences in neural architectures. As CKA and Procrustes belong to this metrics family, we can expect them to underestimate the similarity between activations coming from layers of different type (e.g., convolutional and deconvolutional).

Procrustes is sensitive to the number of data examples As we may have representations with high dimensional features (e.g., activations of convolutional layers), we checked the impact of the number of data examples on CKA and Procrustes. To do so, we created four increasingly different matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$, and $\boldsymbol{D}$ with 50 features each: $\boldsymbol{B}$ retains 80% of $\boldsymbol{A}$'s features, $\boldsymbol{C}$ 50%, and $\boldsymbol{D}$ 0%. We then computed the similarity scores given by CKA and Procrustes while varying the number of data examples. As shown in Figures 7.1a and 7.1b, both metrics agree for $sim(\boldsymbol{A}, \boldsymbol{B})$ and $sim(\boldsymbol{A}, \boldsymbol{C})$, giving scores that are close to the fraction of common features between the two matrices. However, we can see in Figure 7.1c that Procrustes highly overestimates $sim(\boldsymbol{A}, \boldsymbol{D})$ while CKA scores rapidly drop.



(a) $sim(\boldsymbol{A}, \boldsymbol{B})$ (b) $sim(\boldsymbol{A}, \boldsymbol{C})$ (c) $sim(\boldsymbol{A}, \boldsymbol{D})$

Figure 7.1: We compute CKA and Procrustes similarity scores with an increasing number of data examples n, and different similarity strength: $\boldsymbol{B}$ retains 80% of $\boldsymbol{A}$'s features, $\boldsymbol{C}$ 50%, and $\boldsymbol{D}$ 0%. Both metrics agree in (a) and (b), but Procrustes overestimates similarity in (c).

CKA ignores small changes in representations When considering a sufficient number of data examples for both Procrustes and CKA, if two representations do not have dramatic differences (i.e., their 10% largest principal components are the same), CKA may overestimate similarity, while Procrustes remains stable, as observed by Ding, Denain and Steinhardt (2021).

Ensuring accurate analysis Given the limitations previously mentioned, we take three remedial actions to guarantee that our analysis is as accurate as possible. Firstly, as both metrics will likely underestimate the similarity between different layer types, we will only discuss the variation of similarity when analysing such cases. For example, we will not compare $sim(\boldsymbol{A}, \boldsymbol{B})$ and $sim(\boldsymbol{A}, \boldsymbol{C})$ if $\boldsymbol{A}$ and $\boldsymbol{B}$ are convolutional layers but $\boldsymbol{C}$ is deconvolutional. We will nevertheless analyse the changes of $sim(\boldsymbol{A}, \boldsymbol{C})$ at different steps of training. Secondly, when both metrics disagree, we know that one of them is likely overestimating the similarity: Procrustes if the number of data examples is not sufficient, CKA if the difference between the two representations is not large enough. Thus, we will always use the smallest of the two results for our interpretations.

7.3 EXPERIMENTAL SETUP

Learning objectives We use the learning objectives presented in Section 4.3, which allow for easy tuning of the ELBO's regularisation strength, namely β-VAE (Higgins et al., 2017), β-TC VAE (Chen et al., 2018), Annealed VAE (Burgess et al., 2018), and DIP-VAE II (Kumar, Sattigeri and Balakrishnan, 2018). To provide fair and complementary insights into the observations from Chapter 6, we will follow the experimental design of Locatello et al. (2019a) regarding the architecture, learning objectives, and regularisation used. Moreover, `disentanglement lib`[1] will be used as a codebase for our experiment. The complete details are available in Appendix C.1.

Datasets For Section 7.4.1, we use three datasets which, based on the results of Locatello et al. (2019a), are increasingly difficult for VAEs in terms of reconstruction loss: dSprites[2] (Higgins et al., 2017), Cars3D (Reed et al., 2015), and SmallNorb (LeCun, Huang and Bottou, 2004).

Training process For Section 7.4.1, we trained five models with different initialisations for 300,000 steps for each (learning objective, regularisation strength, dataset) triplet, and saved intermediate models to compare the similarity within individual models at different epochs. Appendix C.4 explains our epoch selection methodology.

Similarity measurement For every dataset, we sampled 5,000 data examples, and we used them to compute all the similarity measurements. We compute the similarity scores between all pairs of layers of the different models following the different combinations outlined above. As Procrustes similarity takes significantly longer to compute compared to CKA (see below), we only used it to validate CKA results, restricting its usage to one dataset: Cars3D. We will see in Section 7.4.2 that we obtained similar results for the two metrics on Cars3D, as expected.

Computational considerations Overall we trained 300 VAEs using 4 learning objectives, 5 different initialisations, 5 regularisation strengths, and 3 datasets, which took around 6,000 hours on an NVIDIA A100 GPU. We then computed the CKA scores for the 15 layer activations (plus the input) of each model combinations considered above at 5 different epochs, resulting in 470 million similarity scores and approximately 7,000 hours of computation on an Intel Xeon Gold 6136 CPU. As Procrustes is slowed down by the computation of the nuclear norm for high dimensional activations, the same number of similarity scores would have been prohibitively long to compute, requiring 30,000 hours on an NVIDIA A100 GPU. We thus only computed the Procrustes similarity for one dataset, reducing the computation time to 10,000 hours. Based on the estimations of Lacoste et al. (2019), the computations done for this experiment amount to 2,200 Kg of CO_2, which corresponds to the CO_2 produced by one person over 5 months. To mitigate the negative environmental impact of our work, we released all

1 `https://github.com/google-research/disentanglement_lib`
2 Licensed under an Apache 2.0 licence.

our trained models and metric scores at https://data.kent.ac.uk/428/, and https://data.kent.ac.uk/444/, respectively. We hope that this will help to prevent unnecessary recomputation should others wish to reuse our results.

7.4 RESULTS

In this section, we will discuss the similarity between models through the heatmaps obtained with CKA. The two models being compared will be described in the x and y axis, and the results will be averaged over 5 runs of each model. Specifically, given the i^{th} run of two models A_i and B_i and their activations $\boldsymbol{M}_j^{A_i}$ and $\boldsymbol{M}_j^{B_i}$ over n examples at layer j, the value displayed at the cell (j, k) of the heatmap corresponds to the averaged CKA scores between the j^{th} layer of A and the k^{th} layer of B:

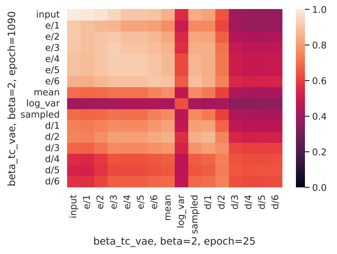
$$CKA_{avg}(\boldsymbol{M}_j^A, \boldsymbol{M}_k^B) = \frac{1}{25}\sum_{i=1}^{5}\sum_{l=1}^{5} CKA(\boldsymbol{M}_j^{A_i}, \boldsymbol{M}_k^{B_l}). \tag{7.9}$$

When describing these figures, we will refer to top-left (resp. bottom-right) quadrants to indicate the similarity scores between all the representations of the encoder (resp. decoder). This includes the off-diagonal CKA scores between layers of the same type. Similarly we will refer to the top-right (resp. bottom-left) quadrants when comparing the representations learned by the encoder and decoder of two models. Note that in this case, we will always discuss the difference of scores between both models, (i.e., both quadrants will be compared). Indeed the layers of the encoder and decoder are of different type and discussing the scores obtained for only one quadrant without contrasting it with the other may be misleading as explained in Section 7.2.

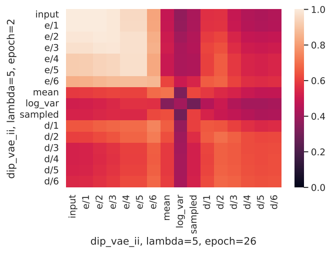
7.4.1 *Assessing the coherence of CKA scores with known facts about VAEs*

The goal of this section is to verify that CKA can provide accurate information about the learning dynamics of VAEs. We thus check that the results observed using representational similarity are consistent with known facts about VAEs. Note that we obtained similar results using Procrustes similarity and fully connected neural network architectures, as reported in Sections 7.4.2 and 7.4.3.

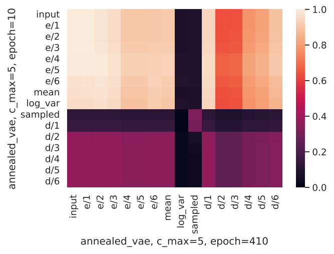
Fact 1: the encoder is learned before the decoder Using the IB theory, (Lee and Jo, 2021) have shown that in VAEs, the encoder is learned before the decoder. Moreover, this behaviour seems to be required for VAEs to learn meaningful representations as decoders which ignore the latent representations (e.g., because of posterior col-



(a) Trained on Cars3D



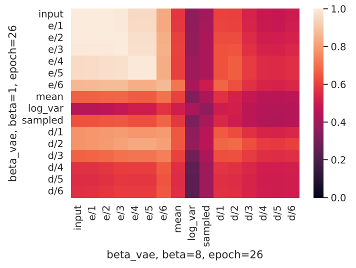
(b) Trained on dSprites



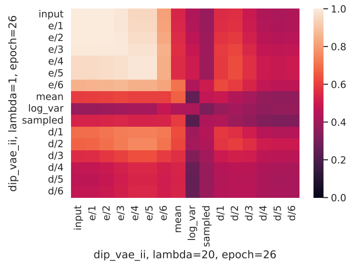
(c) Trained on SmallNorb

Figure 7.2: (a) shows the CKA similarity scores of activations at epochs 25 and 1090 of β-TC VAE trained on Cars3D with $\beta = 2$. (b) shows the CKA similarity scores of activations at epochs 2 and 26 of DIP-VAE II trained on dSprites with $\lambda = 5$. (c) shows the CKA similarity scores of activations at epochs 10 and 410 of Annealed VAE trained on SmallNorb with $c_{max} = 5$.

lapse or lagging inference) provide suboptimal reconstructions (Bowman et al., 2016; He et al., 2019). When comparing the representations learned at different epochs in Figure 7.2, we can see that CKA provides consistent results about this phenomenon: the encoder is learned first, and the representations of its layers become similar to the input after a few epochs (see the bright cells in the top-left quadrants in Figures 7.2a, 7.2b, and 7.2c). The decoder then progressively learns representations that gradually become closer to the input while the mean and variance representations are refined (see the dark cells in the bottom-right quadrant of Figures 7.2a, 7.2b, and 7.2c). Note that our choice of snapshots and snapshot frequency did not influence the results as verified in Appendices C.4 and C.5.



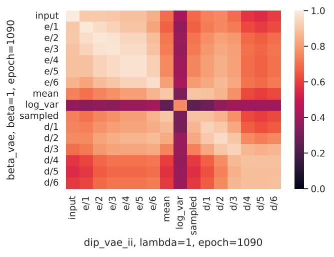
(a) β-VAE



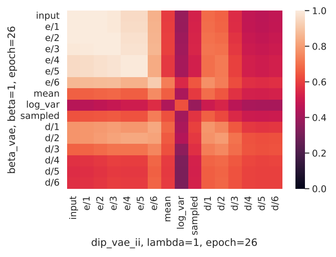
(b) DIP-VAE II

Figure 7.3: (a) shows the CKA similarity scores between the activations of two β-VAEs trained on dSprites with $\beta = 1$, and $\beta = 8$. (b) shows the CKA similarity scores between the activations of two DIP-VAE II trained on dSprites with $\lambda = 1$, and $\lambda = 20$. For both figures, the activations are taken after complete training.

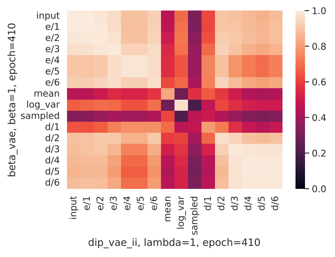
Fact 2: very high regularisation leads to posterior collapse It is well known that an excessively high pressure on the regularisation term of the ELBO in Equation 3.5 leads to posterior collapse (Dai and Wipf, 2018; Lucas et al., 2019b,a; Dai, Wang and Wipf, 2020). When this happens, the sampled representation collapses to the prior — generally $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ — and the decoder has a poor reconstruction quality. This phenomenon is clearly visible with CKA in Figure 7.3. Indeed, the sampled representations of the collapsed model (dark line at the "sampled" column of Figures 7.3a and 7.3b) have a very low similarity with the representations learned by the encoder and decoder of a well-behaved model, in opposition to the sampled representation of a well-behaved model (lighter line at the "sampled" row of Figures 7.3a and 7.3b). This indicates that the collapsed sampled representations do not retain any information about the input, in opposition to any layer of a well-behaved model.



(a) Trained on Cars3D



(b) Trained on dSprites



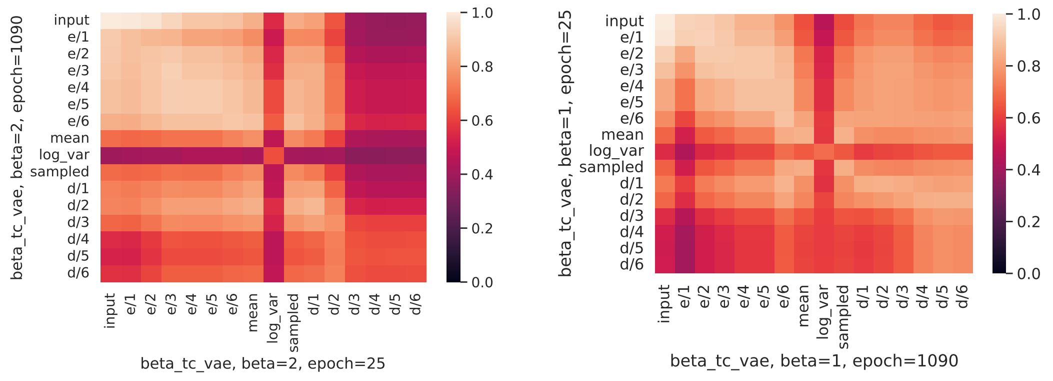
(c) Trained on SmallNorb

Figure 7.4: (a) shows the CKA similarity scores of activations of β-VAE and DIP-VAE II trained on Cars3D with $\beta = 1$, and $\lambda = 1$, respectively. (b) and (c) show the CKA similarity scores of the same learning objectives and regularisation strengths but trained on dSprites and SmallNorb.

Fact 3: encoders learn abstract features It is not uncommon amongst practitioners to apply transfer learning to VAEs by using a pre-trained classifier as an encoder. The last layers (i.e., closest to the output and used for classification) are removed and replaced by the mean and variance layers. The assumption, which underlies transfer learning, is that encoders learn abstract features which are shared with other types of network (Yosinski et al., 2014; Bansal, Nakkiran and Barak, 2021; Csiszárik et al., 2021). The results of CKA are also in line with this fact. Indeed, the bright top-left quadrants of Figures 7.4a, 7.4b, and 7.4c, indicate that encoders of VAEs trained using different learning objectives are highly similar. This observation holds when comparing encoders and classifiers with equivalent architectures (see Appendix C.3).

7.4.2 *Consistency of the results with Procrustes Similarity*

As mentioned in Section 7.3 and illustrated in Figures 7.5 to 7.7, CKA and Procrustes similarity scores for the Cars3D dataset are similar. Figures 7.5 and 7.6 show that Procrustes tends to overestimate the similarity between high-dimensional inputs, as mentioned in Section 7.2.2 (recall the example given in Figure 7.1). In Figure 7.7, we observe a slightly lower similarity with Procrustes than CKA on the 5^{th} and 6^{th} layers of the encoder, indicating that some small changes in the representations may have been underestimated by CKA, as discussed in Section 7.2.2 and by Ding, Denain and Steinhardt (2021). Note that the difference between the CKA and Procrustes similarity scores in Figure 7.7 remains very small (around 0.1) indicating consistent results between both metrics.



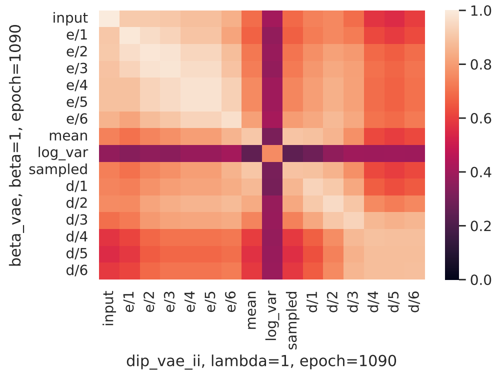
(a) CKA (b) Procrustes

Figure 7.5: (a) shows the CKA similarity scores of activations at epochs 25 and 1090 of β-TC VAE trained on Cars3D with $\beta = 2$. (b) shows the Procrustes similarity scores of the same configuration. We observe the same trend with both metrics with Procrustes slightly overestimating the similarity between high dimensional activations (bottom-right quadrants), which agrees with the properties of the Procrustes similarity reported in Section 7.2.2.

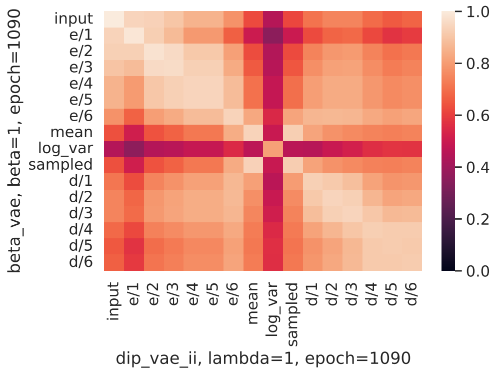
7.4.3 *CKA on fully-connected architectures*

In order to assess the generalisability of our observations, we have repeated our experiment on the fully-connected VAEs that are described in Appendix C.1. We can see in Figures 7.8, 7.9, and 7.10 that the same general trend as for the convolutional architectures can be identified.

Learning in fully-connected VAEs is also bottom-up We can see in Figure 7.8 that, similarly to the convolutional architectures shown in Figure 7.2, the encoder is



(a) CKA



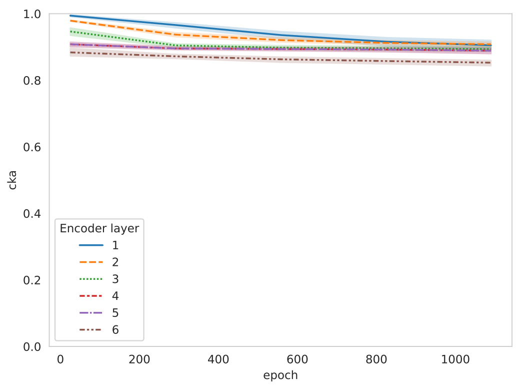
(b) Procrustes

Figure 7.6: (a) shows the CKA similarity scores of activations of β-VAE and DIP-VAE II trained on Cars3D with $\beta = 1$, and $\lambda = 1$, respectively. (b) shows the Procrustes similarity scores using the same configuration. We observe the same trend with both metrics with Procrustes slightly overestimating the similarity between high dimensional activations (bottom-right quadrants) (c.f. Section 7.2.2).

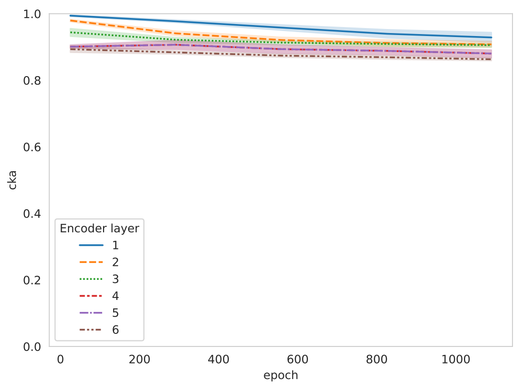
learned early in the training process. Indeed between epochs 1 and 10, the encoder representations become highly similar to the representations of the fully trained model (see Figures 7.8a and 7.8b). The decoder is then learned with its representational similarity with the fully trained decoder raising after epoch 10 (see Figure 7.8c).

Impact of regularisation As in convolutional architectures shown in Figure 7.3, the variance and sampled representations retain little similarity with the encoder representations in the case of posterior collapse, as shown in Figure 7.9. Interestingly, in fully-connected architectures the decoder retains more similarity with its less regularised version than in convolutional architectures, despite suffering from poor reconstruction when heavily regularised. Thus, CKA of the representations of fully-connected decoders may not be a good predictor of reconstruction quality.

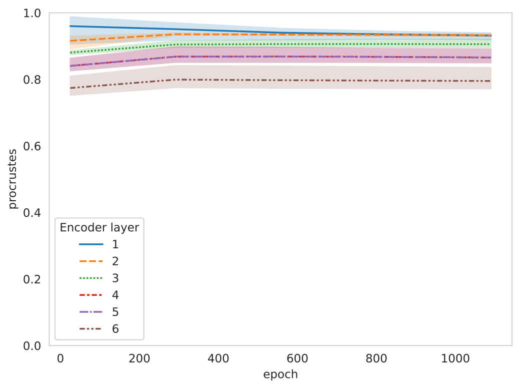
Impact of learning objective Figure 7.10 provides results similar to the convolutional VAEs observed in Figure 7.4, with a very high similarity between encoder layers learned from different learning objectives (see diagonal values of the upper-left quadrant). Here again, the representational similarity of the decoder seems to vary depending on the dataset, even though this is less marked than for convolutional architectures. We can also see that the representational similarity between different layers of the encoder vary depending on the dataset, which was less visible in convolutional architectures. For example, the similarity between the first and subsequent layers of the encoder in SmallNorb is much lower in fully-connected VAEs. Given that SmallNorb is a hard dataset to learn for VAEs (Locatello et al., 2019a), one



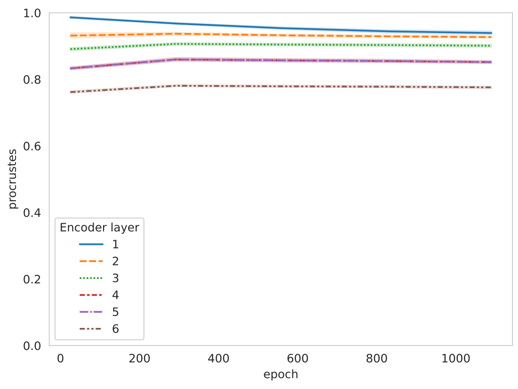
(a) CKA for $\beta = 1$



(b) CKA for $\beta = 8$



(c) Procrustes for $\beta = 1$



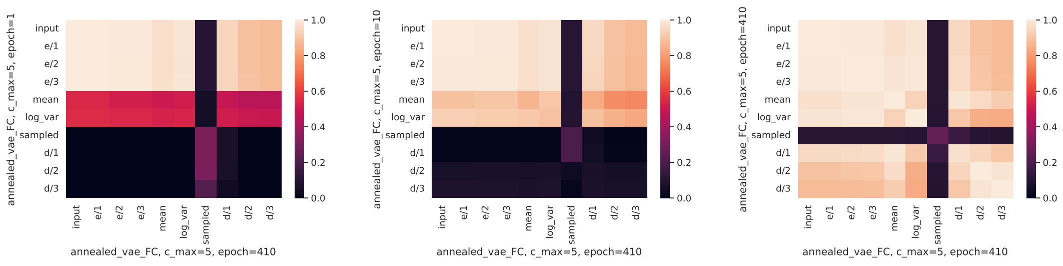
(d) Procrustes for $\beta = 8$

Figure 7.7: (a) shows the CKA scores between the inputs and the activations of the first 6 layers of the encoder of a β-VAE trained on Cars3D with $\beta = 1$. (b) shows the scores between the same representations with $\beta = 8$. (c) and (d) are the Procrustes scores of the same configurations. We observe the same trend for both metrics with more variance in (c) for Procrustes with $\beta = 1$. Procrustes also displays a slightly lower similarity for layers 4 to 6 of the encoder, possibly due to changes in the representation that are underestimated by CKA (c.f. Section 7.2.2).

could hypothesise that the encoder of fully-connected VAE, being less powerful, is unable to retain as much information as its convolutional counterpart, leading to lower similarity scores with the representations of the first encoder layer.

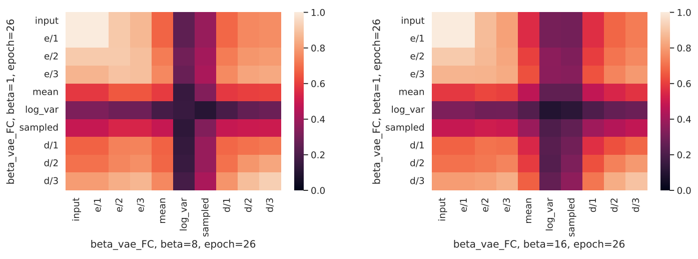
7.4.4 *Impact on the latent variables learned under the polarised regime*

In Section 7.4.1, we have high CKA score between the mean representations of models trained with different seeds and, in some cases, learning objectives. Given that CKA is invariant to rotation and scaling, it means that mean representations with high similarity will have the same number of active and passive variables but the number of active and passive variables may differ when the CKA similarity is lower.



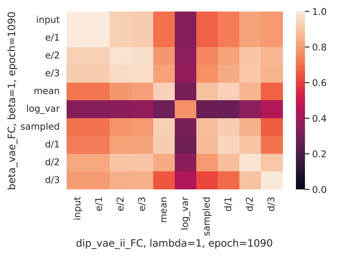
(a) Epoch 1 (b) Epoch 10 (c) Epoch 410

Figure 7.8: (a), (b), and (c) show CKA scores between a fully-trained fully-connected Annealed VAE and a fully-connected Annealed VAE trained for 1, 10 and 410 epochs, respectively. All the models are trained on SmallNorb and the results are averaged over 5 seeds. As in Figure 7.2 of Section 7.4.1, we can see that there is a high similarity between the representations learned by the encoder early in the training and after complete training (see the bright cells in the top-left quadrants in (a), (b), and (c)). The mean and variance representations similarity with a fully trained model increase after a few more epochs (the purple line in the middle disappear between (a) and (b)), and finally the decoder is learned (see bright cells in bottom-right quadrant of (c)).

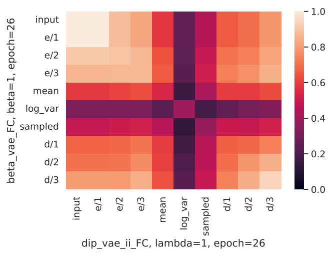


(a) $\beta = 8$ (b) $\beta = 16$

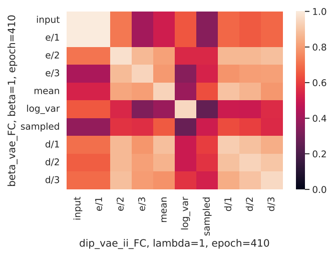
Figure 7.9: (a) and (b) show the representational similarity between fully-connected β-VAEs trained with $\beta = 1$, and fully-connected β-VAEs trained with $\beta = 8$ and $\beta = 16$, respectively. All models are trained on dSprites and the scores are averaged over 5 seeds. In both figures, the encoder representations stay very similar (bright cells in the top-left quadrants), except for the mean, variance and sampled representations. While the variance representation is increasingly different as we increase β, the decoder does not show the dramatic dissimilarity observed in convolutional architectures in Figure 7.3.



(a) Trained on Cars3D



(b) Trained on dSprites



(c) Trained on SmallNorb

Figure 7.10: (a) shows the CKA similarity scores of activations of β-VAE and DIP-VAE II with fully-connected architectures trained on Cars3D with $\beta = 1$, and $\lambda = 1$, respectively. (b) and (c) show the CKA similarity scores of the same learning objectives and regularisation strengths but trained on dSprites and SmallNorb. All the results are averaged over 5 seeds. We can see that the representational similarity of all the layers of the encoder (top-left quadrant) except mean and variance is high (CKA $\geqslant 0.8$). However, as in Figure 7.4, the mean, variance, sampled (center diagonal values), and decoder (bottom-right quadrants) representational similarity of different learning objectives seems to vary depending on the dataset. In (a) and (c) they have high similarity, while in (b) the similarity is lower. Moreover, in (c) the input and first layer of the encoder are quite distinct from the other representations, which was not the case in convolutional architectures.

To see why this is the case, let us first consider the linear case. We will then extend the result to the non-linear case in Theorem 7.4.2

Theorem 7.4.1 (Conservation of variables for linear models)**.** *Given two linear VAEs learned in a polarised regime with variance representations $\mathbf{\Sigma}$ and $\bar{\mathbf{\Sigma}}$, if $CKA(\mathbf{\Sigma}, \bar{\mathbf{\Sigma}}) = 1$, then $\mathbf{\Sigma}$ and $\bar{\mathbf{\Sigma}}$ have the same number of active and passive variables.*

Proof. If $CKA(\mathbf{\Sigma}, \bar{\mathbf{\Sigma}}) = 1$, then $\bar{\mathbf{\Sigma}} = \xi \boldsymbol{R}\mathbf{\Sigma}$ where $\xi \neq 0$ is a scaling factor and $\boldsymbol{R}$ is a rotation matrix.
Recall from Equation 4.15 that given $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{P}\mathbf{\Lambda}\boldsymbol{P}^T$, where $\boldsymbol{P}$ is a permutation matrix, $\mathbf{\Sigma} = \boldsymbol{P}\,\mathrm{diag}\left[\frac{1}{\frac{\mathbf{\Lambda}_{11}^2}{\gamma}+1}, \cdots, \frac{1}{\frac{\mathbf{\Lambda}_{aa}^2}{\gamma}+1}, 1, \cdots, 1\right]\boldsymbol{P}^T$. As $\gamma \to 0$, the first k values, which are active variables become close to 0 and the remaining $k-n$ passive variables are close to 1. Thus, we have

$$\begin{aligned}\bar{\mathbf{\Sigma}} &= \xi \boldsymbol{R}\boldsymbol{P}\,\mathrm{diag}\left[\frac{1}{\frac{\mathbf{\Lambda}_{11}^2}{\gamma}+1}, \cdots, \frac{1}{\frac{\mathbf{\Lambda}_{aa}^2}{\gamma}+1}, 1, \cdots, 1\right]\boldsymbol{P}^T, \\ &= \boldsymbol{V}\,\mathrm{diag}\left[\frac{1}{\frac{\xi\mathbf{\Lambda}_{11}^2}{\gamma}+1}, \cdots, \frac{1}{\frac{\xi\mathbf{\Lambda}_{aa}^2}{\gamma}+1}, \xi, \cdots, \xi\right]\boldsymbol{P}^T.\end{aligned}$$

where $\boldsymbol{V}$ is an orthogonal matrix because both $\boldsymbol{R}$ and $\boldsymbol{P}$ are orthogonal matrices. As $\gamma \to 0$, the first k values, which are active variables become close to 0 and the remaining $k - n$ passive variables are close to ξ. Thus, $\bar{\boldsymbol{R}}$ has the same number of active and passive variables as $\boldsymbol{R}$, as expected. □

Furthermore, it is easy to see that when $CKA(\Sigma, \bar{\Sigma}) < 1$, Σ and $\bar{\Sigma}$ are not guaranteed to have the same number of active and passive variables. For example, let us consider the case where $\boldsymbol{W}^T\boldsymbol{W} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, such that $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = \mathrm{diag}[1, 0]$. Here, the first variable of Σ is passive and the second is active. If $\bar{\Sigma} = \boldsymbol{T}\Sigma$, where $\boldsymbol{T} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$. We will have $CKA(\Sigma, \bar{\Sigma}) < 1$, because such transformation do not preserve angles. The resulting $\bar{\Sigma}$ will be $\bar{\Sigma} = \mathrm{diag}[0, 0]$ and $\bar{\Sigma}$ will have two active variables while Σ only has one.

Theorem 7.4.2 (Conservation of variables for non-linear models)**.** *Given two VAEs learned in a polarised regime with non-linear variance representations $\Sigma_{\boldsymbol{\phi}}(\mathbf{x})$ and $\bar{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})$, if $CKA(\Sigma_{\boldsymbol{\phi}}(\mathbf{x}), \bar{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})) = 1$, then $\Sigma_{\boldsymbol{\phi}}(\mathbf{x})$ and $\bar{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})$ have the same number of active and passive variables.*

Proof. Let us consider the case where $p_{\boldsymbol{\theta}}(\mathbf{z}) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$, $\Sigma_{\boldsymbol{\phi}}(\mathbf{x}))$, and $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z})$, $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}), \gamma\boldsymbol{I})$, such that $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}), \Sigma_{\boldsymbol{\phi}}(\mathbf{x})$ and $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z})$ are non-linear functions. From Equations 3.5 and 2.19, we have

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) =& \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \parallel p_{\boldsymbol{\theta}}(\mathbf{z})\big), \\ =& -\frac{1}{2}\Bigg(m \log\gamma + \frac{1}{\gamma}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\|\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z})\|_2^2\right] + \mathrm{Tr}\left(\Sigma_{\boldsymbol{\phi}}(\mathbf{x})\right) \\ &+ \|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\|_2^2 - \log\det\left(\Sigma_{\boldsymbol{\phi}}(\mathbf{x})\right) - n\Bigg). \end{aligned} \tag{7.10}$$

Following (Dai et al., 2017; Dai and Wipf, 2018; Rolinek, Zietlow and Martius, 2019) we use a Taylor expansion to have a linear approximation of the ELBO at $\mathbf{z} = f_{\boldsymbol{\mu}_z}(\mathbf{x})$. We thus have

$$\begin{aligned} \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}) &= \boldsymbol{\mu}_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big) + \boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big) \cdot \big(\mathbf{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big), \\ &\approx \mathbf{x} + \boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big) \cdot \big(\mathbf{z} - \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big). \end{aligned} \tag{7.11}$$

Plugging Equation 7.11 into Equation 7.10, we obtain

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}) \approx &-\frac{1}{2}\Bigg(m\log\gamma+\frac{1}{\gamma}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\left\|\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\cdot\big(\mathbf{z}-\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\right\|_2^2\right]\\
&+\operatorname{Tr}\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)+\|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\|_2^2-\log\det\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)-n\Bigg),\\
=&-\frac{1}{2}\Big(m\log\gamma+\operatorname{Tr}\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)+\|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\|_2^2-\log\det\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)-n\Big)\\
&-\frac{1}{2\gamma}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Bigg[\operatorname{Tr}\Big(\big(\mathbf{z}-\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\big(\mathbf{z}-\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\\
&\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\Big)\Bigg],\\
=&-\frac{1}{2}\Big(m\log\gamma+\operatorname{Tr}\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)+\|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\|_2^2-\log\det\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)-n\Big)\\
&-\frac{1}{2\gamma}\operatorname{Tr}\Bigg(\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\big(\mathbf{z}-\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\big(\mathbf{z}-\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\Big]\\
&\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\Bigg),\\
=&-\frac{1}{2}\Bigg(\operatorname{Tr}\Big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\Big(\boldsymbol{I}_n+\frac{1}{\gamma}\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\Big)\Big)\\
&-\log\det\big(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})\big)+m\log\gamma+\|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\|_2^2-n\Bigg).\end{aligned}\tag{7.12}$$

Using Equation 7.12, we can now minimise $-\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x})$ w.r.t. $\boldsymbol{\Sigma}$,

$$\begin{aligned}-\frac{\partial\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x})}{\partial\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})}&=\frac{1}{2}\Big(\Big(\boldsymbol{I}_n+\frac{1}{\gamma}\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\Big)-\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})^{-1}\Big)=0,\\
\Leftrightarrow\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})&=\Big(\boldsymbol{I}_n+\frac{1}{\gamma}\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)\Big)^{-1}.\end{aligned}\tag{7.13}$$

Letting the SVD of $\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)^T\boldsymbol{\mu}'_{\boldsymbol{\theta}}\big(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})\big)=\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, we obtain

$$\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})=\boldsymbol{U}\operatorname{diag}\left[\frac{1}{\frac{\boldsymbol{\Lambda}_{11}^2}{\gamma}+1},\cdots,\frac{1}{\frac{\boldsymbol{\Lambda}_{aa}^2}{\gamma}+1},1,\cdots,1\right]\boldsymbol{U}^T.\tag{7.14}$$

If $CKA(\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x}),\bar{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}(\mathbf{x}))=1$, then $\bar{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}(\mathbf{x})=\xi\boldsymbol{R}\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})$ where $\xi\neq0$ is a scaling factor and $\boldsymbol{R}$ is a rotation matrix.

$$\bar{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}(\mathbf{x})=\xi\boldsymbol{V}\operatorname{diag}\left[\frac{1}{\frac{\boldsymbol{\Lambda}_{11}^2}{\gamma}+1},\cdots,\frac{1}{\frac{\boldsymbol{\Lambda}_{aa}^2}{\gamma}+1},1,\cdots,1\right]\boldsymbol{U}^T.\tag{7.15}$$

where $\boldsymbol{V}$ is an orthogonal matrix because both $\boldsymbol{R}$ and $\boldsymbol{U}$ are orthogonal matrices. As $\gamma \to 0$,

$$\bar{\boldsymbol{\Sigma}}_{\phi}(\mathbf{x}) = \xi \boldsymbol{V} \operatorname{diag}\left[0, \cdots, 0, 1, \cdots, 1\right] \boldsymbol{U}^{T}. \tag{7.16}$$

As expected $\bar{\boldsymbol{\Sigma}}_{\phi}(\mathbf{x})$ also has a active variables and $n - k$ passive variables. □

It is also worth noting that because the encoder is learned first, the final number of active and passive variables can be determined very early in the training. We will see in Chapter 8 that both of these points allow us to use the polarised regime as a tool to determine a good dimensionality for the latent space of a VAE in an unsupervised way without fully training any models.

7.5 CONCLUSION

After ensuring that CKA was consistent with known behaviours of VAEs in Section 7.4.1, we used this metric to show that mean representations of different models had the same number of active and passive variables as long as the similarity was high. However, when the similarity is lower, latent representations are not guaranteed to display the same number of active and passive variables across models. Regardless of the similarity between models, because the latent representations of any encoder converge very early in the training, the number of active and passive variables is fixed very quickly.

Impact on component reusability One could hypothesise that two models which equally penalise the KLD but have different number of active and passive variables will have a low CKA similarity. This would also indicate that the angles between the different data samples are different in the two latent spaces. This is an important point to verify before attempting to perform model stitching as methods which rely on the conservation of angles, like relative representations (Moschella et al., 2023) would likely fail in that context. Thus, before attempting relative representation stitching, one should always ensure that the representations to be stitched have a high CKA score.

Impact for the polarised regime The properties of latent representations under the polarised regime are well researched (see Chapter 4), and we have shown that passive and active variables are consistent across seeds and can be detected early during training. Thus, one could use their properties as a tool to monitor or improve VAEs. We will see such an example in Chapter 8 where a measure of the number of active and passive variables is used very early in the training to determine in an unsupervised way how many latent variables are needed to train a VAEs on a specific dataset.

8

AUTOMATING THE SELECTION OF THE NUMBER OF LATENT DIMENSIONS

8.1 INTRODUCTION

"How many latent variables should I use for this model?" is the question that many practitioners using VAEs or AEs have to deal with. When the task has been studied before, this information is available in the literature for the specific architecture and dataset used. However, when it has not, answering this question becomes more complicated. Indeed, the current methods to estimate the latent dimensions require human supervision or at least one fully trained model (Doersch, 2016; Mai Ngoc and Hwang, 2020; Yu and Príncipe, 2019; Boquet et al., 2021). Moreover, most of the proposed solutions are not theoretically grounded.

One could wonder if, instead of looking for an appropriate number of dimensions, it would be sufficient to use a very large number of latent dimensions in all cases. However, beside defeating the purpose of learning compressed representations, this may lead to a range of issues. For example, one would obtain lower accuracy on downstream tasks (Mai Ngoc and Hwang, 2020) and—if the number of dimensions is sufficiently large—very high reconstruction loss (Doersch, 2016). This would also hinder the interpretability of downstream task models such as linear regression, prevent investigating the learned representation with latent traversal (Locatello et al., 2019a), and increase the correlation between the latent variables, as we have shown in Chapter 6.

So far, we have seen in Chapters 4.2 and 6 that well-behaved VAEs learn in a polarised regime, which leads to discrepancies between the mean and sampled representations. Specifically, we concluded in Chapter 6 that mixed and passive variables were responsible for any discrepancies between the mean and sampled representations. This finding will be the base of our analysis in Section 8.3. Thus, one can reasonably assume that a "good" number of latent dimensions corresponds to the maximum number of active variables (i.e., variables used by the decoder for reconstruction) a model can learn before the appearance of passive variables.

In this chapter, we demonstrate that the amount of passive variables present in a latent representation can easily be detected by analysing the discrepancies between the mean and sampled representations. Specifically, we show that one can bound the difference between the traces of the covariance matrices of both representations. Based on this property, we then design a simple yet efficient unsupervised algorithm which fulfills the criteria of the current methods (i.e., low reconstruction loss, high accuracy on downstream tasks, high compression, etc.), without requiring to fully train any models.

Our contributions are as follows:

1. We prove that the discrepancies between the mean and sampled representations of well-behaved VAEs can be used to monitor the type of variables (active or passive) learned by a VAE.
2. Based on this theoretical insight, we propose FONDUE: an algorithm which finds the number of latent dimensions that leads to a low reconstruction loss and good accuracy. In opposition to current methods (Doersch, 2016; Mai Ngoc and Hwang, 2020; Yu and Príncipe, 2019; Boquet et al., 2021), it does not require human supervision or to fully train multiple models.
3. The library created for this experiment is available at https://github.com/bonheml/VAE_learning_dynamics and can be reused with other models or techniques (see paragraph on extending FONDUE of Section 8.3) for further research in the domain. A notebook illustrating the usage of FONDUE can also be found at https://github.com/bonheml/fondue-demo.

Notational considerations As in the previous chapters, we will use the superscript $^{(i)}$ to denote the values obtained for the ith sample $\mathrm{x}^{(i)}$ of the random variable x, and represent the jth dimension of a vector representation using the subscript $_j$. Similarly, given a subset of indexes $\boldsymbol{a} = 1, \cdots, k$, the superscript $^{(\boldsymbol{a})}$ will denote the values obtained for the samples $\mathrm{x}^{(1,\dots,k)}$ of the random variable x, and the subscript $_{\boldsymbol{a}}$ will represent the dimensions $1, ..., k$ of a vector representation. We will also use a shortened version of the mean, variance and sampled representations, such that $\boldsymbol{\mu} \triangleq \boldsymbol{\mu}(\mathrm{x};\boldsymbol{\phi})$, $\boldsymbol{\sigma} \triangleq diag[\boldsymbol{\Sigma}(\mathrm{x};\boldsymbol{\phi})]$, and $\mathrm{z} \triangleq \boldsymbol{\mu} + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}^{1/2}$, respectively. Similarly, for a specific data example $\mathrm{x}^{(i)}$ and noise sample $\boldsymbol{\epsilon}^{(i)}$, $\boldsymbol{\mu}^{(i)} \triangleq \boldsymbol{\mu}(\mathrm{x}^{(i)};\phi)$, $\boldsymbol{\sigma}^{(i)} \triangleq diag[\boldsymbol{\Sigma}(\mathrm{x}^{(i)};\phi)]$ and $\mathrm{z}^{(i)} \triangleq \boldsymbol{\mu}^{(i)} + \boldsymbol{\epsilon}^{(i)} \odot (\boldsymbol{\sigma}^{(i)})^{1/2}$. For multiple data examples $\boldsymbol{X} = \{\mathrm{x}^{(i)}\}_{i=0}^{h}$, $\boldsymbol{M} \triangleq [\boldsymbol{\mu}^{(0)} \cdots \boldsymbol{\mu}^{(h)}]^T$, $\boldsymbol{S} \triangleq [\boldsymbol{\sigma}^{(0)} \cdots \boldsymbol{\sigma}^{(h)}]^T$, $\boldsymbol{E} \triangleq [\boldsymbol{\epsilon}^{(0)} \cdots \boldsymbol{\epsilon}^{(h)}]^T$, and $\boldsymbol{Z} \triangleq [\mathrm{z}^{(0)} \cdots \mathrm{z}^{(h)}]^T$.

8.2 BACKGROUND

8.2.1 *Intrinsic dimension estimation*

It is generally assumed that a dataset $\boldsymbol{X} = \{\mathbf{x}^{(i)}\}_{i=0}^{h}$ of h i.i.d. data examples $\mathbf{x}^{(i)} \in \mathbb{R}^m$ is a locally smooth non-linear transformation g of a lower-dimensional dataset $\boldsymbol{Y} = \{\mathbf{y}^{(i)}\}_{i=0}^{h}$ of h i.i.d. samples $\mathbf{y}^{(i)} \in \mathbb{R}^d$, where $d \leqslant m$ (Campadelli et al., 2015; Chollet, 2021). The goal of ID estimation is to recover d given $\boldsymbol{X}$. In recent years, these techniques have successfully been applied to deep learning to empirically show that the intrinsic dimension of images was much lower than their extrinsic dimension (i.e., the number of pixels) (Gong, Boddeti and Jain, 2019; Ansuini et al., 2019; Pope et al., 2021). Based on these findings, we will use IDEs as the initial number of dimensions n for FONDUE in Section 8.3.2. In practice, we compute the IDEs using two ID estimation techniques: MLE (Levina and Bickel, 2004) and Two Nearest Neighbours (TwoNN) (Facco et al., 2017).

8.2.2 *Maximum Likelihood Estimation*

Levina and Bickel (2004) modelled the neighbourhood of a given point $\mathbf{x}^{(i)}$ as a Poisson process in a d-dimensional sphere $S_{\mathbf{x}^{(i)}}(R)$ of radius R around $\mathbf{x}^{(i)}$. This Poisson process is denoted $\{N(t, \mathbf{x}^{(i)}), 0 \leqslant t \leqslant R\}$, where $N(t, \mathbf{x}^{(i)})$ is a random variable representing the number of neighbours of $\mathbf{x}^{(i)}$ within a radius t, and is distributed according to a Poisson distribution[1]. This Poisson process, $\{N(t, \mathbf{x}^{(i)}), 0 \leqslant t \leqslant R\}$, will count the total number of points falling into the successive d-dimensional spheres of radius $0 \leqslant t \leqslant R$.

Intuitively, when $d = 3$ this can be thought of as an onion to which we add an outer peel for each increasing radius value t, until we reach the maximum radius R. Thus, each $N(t, \mathbf{x}^{(i)})$ will give us a snapshot of the number of points contained in all the peels stacked so far in the onion of radius t. As $N(t, \mathbf{x}^{(i)})$ is a function of the surface area of the sphere, its rate is a function of d and one can estimate d using MLE. However, we generally cannot access all the existing neighbours of $\mathbf{x}^{(i)}$ in a given radius without infinite data, so we approximate the process using a fixed number of neighbours.

More formally, each point $\mathbf{x}^{(j)} \in S_{\mathbf{x}^{(i)}}(R)$ is thus considered as an event, its arrival time $t = T(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ being the Euclidean distance from $\mathbf{x}^{(i)}$ to its j^{th} neighbour $\mathbf{x}^{(j)}$.

1 Note that this does not imply any distributional assumption about the dataset

By expressing the rate $\lambda(t, \mathbf{x}^{(i)})$ of the process $N(t, \mathbf{x}^{(i)})$ as a function of the surface area of the sphere—and thus relating $\lambda(t, \mathbf{x}^{(i)})$ to d—they obtain a MLE of the ID d:

$$\bar{d}_R(\mathbf{x}^{(i)}) = \left[\frac{1}{N(R, \mathbf{x}^{(i)})} \sum_{j=1}^{N(R,\mathbf{x}^{(i)})} \log \frac{R}{T(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})} \right]^{-1} . \tag{8.1}$$

Equation 8.1 is then simplified by fixing the number of neighbours, k, instead of the radius R of the sphere, such that

$$\bar{d}_k(\mathbf{x}^{(i)}) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{T(\mathbf{x}^{(i)}, \mathbf{x}^{(k)})}{T(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})} \right]^{-1} , \tag{8.2}$$

where the last summand is omitted, as it is zero for $j = k$. The final IDE $\bar{d}_k$ is the averaged score over n data examples (Levina and Bickel, 2004)

$$\bar{d}_k = \frac{1}{n} \sum_{i=1}^{n} \bar{d}_k(\mathbf{x}^{(i)}). \tag{8.3}$$

To obtain an accurate estimation of the ID with MLE, it is very important to choose a sufficient number of neighbours k to form a dense small sphere (Levina and Bickel, 2004). On one hand, if k is too small, MLE will generally underestimate the ID, and suffer from high variance (Levina and Bickel, 2004; Campadelli et al., 2015; Pope et al., 2021). On the other hand, if k is too large, the ID will be overestimated (Levina and Bickel, 2004; Pope et al., 2021).

A worked example Now, let us consider the point $\mathbf{x}^{(i)} = (0, 0, 0)$ and 3 closest neighbours $\boldsymbol{Y} = \{(0, 1, 0)(1, 0, 0), (2, 0, 0)\}$. We have $N(t = 1, \mathbf{x}^{(j)}) = 2$ and $N(t = 2, \mathbf{x}^{(j)}) = 3$ because $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}$ are within a radius $t = 1$ of $\mathbf{x}^{(i)}$, and all $\mathbf{y}^{(j)}$ are within a radius $t = 2$.

Using the distances between $\mathbf{x}^{(i)}$ and its neighbours, $T(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})$, the dimensionality can be estimated by Equation 8.2 as follows

$$\begin{aligned} \bar{d}_3(\mathbf{x}^{(i)}) &= \left[\frac{1}{2} \sum_{j=1}^{2} \log \frac{T(\mathbf{x}^{(i)}, \mathbf{y}^{(3)})}{T(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})} \right]^{-1} , \\ &= \left[\frac{1}{2} \sum_{j=1}^{2} \log \frac{2}{T(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})} \right]^{-1} , \\ &= [\log 2]^{-1} , \\ &\approx 3.3, \end{aligned} \tag{8.4}$$

which is reasonably close to the true data ID.

To make sure that the estimate is stable, we repeat this estimation over multiple data points and average the results as per Equation 8.3.

8.2.3 *TwoNN*

Facco et al. (2017) proposed an estimation of the ID based on the ratio of the two nearest neighbours of $\mathbf{x}^{(i)}$, $r_{\mathbf{x}^{(i)}} = \frac{T(\mathbf{x}^{(i)},\mathbf{x}^{(l)})}{T(\mathbf{x}^{(i)},\mathbf{x}^{(j)})}$, where $\mathbf{x}^{(j)}$ and $\mathbf{x}^{(l)}$ are the first and second closest neighbours of $\mathbf{x}^{(i)}$, respectively. r follows a Pareto distribution with scale $s = 1$ and shape d, and its density function $f(r)$ is

$$f(r) = \frac{ds^d}{r^{d+1}} = dr^{-(d+1)}. \tag{8.5}$$

Its cumulative distribution function is thus

$$F(r) = 1 - \frac{s^d}{r^d} = 1 - r^{-d}, \tag{8.6}$$

and, using Equation 8.6, one can readily obtain $d = \frac{-\log(1-F(r))}{\log r}$. From this, we can see that d is the slope of the straight line passing through the origin, which is given by the set of coordinates $\mathbb{S} = \{(\log r_{\mathbf{x}^{(i)}}, -\log(1 - F(r_{\mathbf{x}^{(i)}}))) \mid i = 1, \cdots, m\}$, and can be recovered by linear regression.

As TwoNN uses only two neighbours, it can be sensitive to outliers (Facco et al., 2017) and does not perform well on high ID (Pope et al., 2021), overestimating the ID in both cases.

8.2.4 *Ensuring an accurate analysis*

Given the limitations previously mentioned, we take two remedial actions to guarantee that our analysis is as accurate as possible. To provide an IDE which is as accurate as possible with MLE, we will measure the MLE with an increasing number of neighbours and, similar to Karbauskaitė, Dzemyda and Mazėtis (2011), retain the IDE which is stable for the largest number of k values. TwoNN will be used as a complementary metric to validate our choice of k for MLE. In case of significant discrepancies with a significantly higher TwoNN IDE, we will rely on the results provided by MLE.

Assumptions In the rest of this chapter, we assume that the considered models are learning under the polarised regime, which naturally happens for VAEs whose prior is Gaussian with diagonal covariance (Rolinek, Zietlow and Martius, 2019; Bonheme and Grzes, 2023) and reasonable values of β (i.e., when β is not large enough to lead to posterior collapse). As discussed above, Dai and Wipf (2018) have also shown that active and passive variables can be observed very early in the training of such models (i.e., after the first few epochs) and we will see in Section 8.3 that this early convergence assumption plays an important role on the computational time of FONDUE. These requirements can be made without loss of generality as we have seen in Chapters 4 and 7 that the polarised regime was necessary for VAEs to learn properly and happens very early in training (Dai and Wipf, 2018; He et al., 2019).

8.2.5 *Related work*

To the best of our knowledge, the literature on finding an appropriate number of latent dimensions for VAEs is limited, and the existing techniques—mostly designed for deterministic autoencoders—generally rely on approaches requiring human supervision. A great majority of these techniques are based on the Elbow (a.k.a. scree plot) method (James et al., 2013) which visually finds the point where a curve "bends" before diminishing returns occur.

Elbow method using the reconstruction error Doersch (2016) trained multiple models with different numbers of latent dimensions and selected the ones with the lowest reconstruction error. They noted that models' performances were noticeably worse when using extreme numbers of latent dimensions n. In their experiment, this happened for $n < 4$ and $n > 10,000$ on MNIST.

Elbow method using the accuracy on downstream tasks Mai Ngoc and Hwang (2020) suggested to train multiple models with different numbers of latent dimensions, and then compare the accuracy of the latent representations on a downstream task. They observed that while a higher number of latent dimensions could lead to a lower reconstruction error, it generally caused instability on downstream tasks. They thus concluded that the best number of latent dimensions for VAEs should be the one with the smallest classification variance and the highest accuracy, and they obtained similar results for AEs.

Automated evaluation based on information theory Yu and Príncipe (2019) observed that, from the data preprocessing inequality, $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \geqslant \mathrm{H}(\mathbf{z})$. From this, Boquet et al. (2021) proposed to automatically find the number of latent dimensions of deterministic autoencoders by comparing the values of $\mathrm{I}(\boldsymbol{X}, \hat{\boldsymbol{X}})$ and $\mathrm{H}(\boldsymbol{Z})$. While no formal proof was provided, they hypothesised that given an optimal number of dimensions n^*, any number of latents $n < n^*$ would result in $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) > \mathrm{H}(\mathbf{z})$, and any $n \geqslant n^*$ would lead to $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \approx \mathrm{H}(\mathbf{z})$. From this, they proposed to do a binary search over a sorted array of latent dimensions $\boldsymbol{n} = [1, ..., n]$ defined by the user. At each iteration, they trained a new autoencoder with a bottleneck size $\boldsymbol{n}_i$, estimated $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}})$ and $\mathrm{H}(\mathbf{z})$ for each batch, and averaged the results over each epoch. Each model was fully trained unless $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \approx \mathrm{H}(\mathbf{z})$ (i.e., the algorithm requires at least one fully trained model). As Yu and Príncipe (2019)[2], they used a kernel estimator of the Rényi's α-order entropy with an Radial Basis Function (RBF) kernel but with the order of $\alpha = 2$. We will refer to their algorithm as the IB algorithm in the rest of the chapter.

Differences with our contribution The work the most closely related to ours is the IB algorithm of Boquet et al. (2021) which is the only one proposing some level of automation. However, our contribution differs from theirs in several aspects: 1) our approach is based on the polarised regime, not on information theory; 2) we provide a theoretical justification of our algorithm and a proof of convergence while this was left for future work in Boquet et al. (2021); 3) our solution is generally faster than the original implementation of the IB algorithm as we do not require to fully train any model; 4) the IB algorithm requires human supervision to select a range of likely latent dimensions which is not needed by FONDUE. We will further see in Section 8.5 that the number of dimensions obtained with FONDUE provides a better trade-off between reconstruction quality and accuracy on downstream task than IB. In other words, we will see that FONDUE provides results closer to Elbow methods than IB.

8.3 MEAN AND SAMPLED REPRESENTATIONS UNDER THE POLARISED REGIME

As we have seen in Chapter 6, the polarised regime leads to discrepancies between mean and sampled representations which can be indicative of the type of variables learned by a VAE.

2 More details about Yu and Príncipe (2019) can be found in Appendix D.2

The aim of this section is to provide some theoretical insight into how the different types of variables impact the difference between the two representations.

Specifically, we will study the difference between the traces of the covariance matrices of both representations when the latent variables are composed of a combination of different types of variables. This will allow us to use bounded quantities to detect when passive variables start appearing. From this, we will show that the bounds of this difference are specific to the types of variables present in the latent representations.

Finally, to illustrate a possible usage of these bounds, we will propose an algorithm for Facilely Obtaining the Number of latent Dimensions by Unsupervised Estimation (FONDUE) which can quickly provide an estimate of the maximum number of latent dimensions that a VAE can have without containing any (unused) passive variables. The impact of FONDUE on the reconstruction and downstream task performance will further be studied in the next section.

8.3.1 *Identifying the types of variables learned*

In this section we will derive the difference between the traces of the covariance matrices of $\boldsymbol{M}$ and $\boldsymbol{Z}$ when the latent representation is composed of: (1) only active variables, (2) active and passive variables, and (3) active and mixed variables. From these results we will then present the bounds of the difference between the traces of the covariance matrices of $\boldsymbol{M}$ and $\boldsymbol{Z}$ at $n + 1$ latent variables when (1) n is the maximal number of active variables, and (2) n is the maximal number of non-passive variables. We will see in Section 8.3.2 that these two cases will inform the choice of the threshold value for FONDUE. To make the distinction between the cross-covariance matrices and variance-covariance matrices clear, we will use $\mathrm{Cov}[\boldsymbol{A}, \boldsymbol{B}]$ for the former and $\mathrm{Var}[\boldsymbol{A}] \triangleq \mathrm{Cov}[\boldsymbol{A}, \boldsymbol{A}]$ for the latter.

First, let us consider the case where all variables are active. As mentioned in Theorem 6.3.1, in this case $\lim_{\gamma \to 0} \boldsymbol{Z} = \boldsymbol{M}$. Thus, Proposition 8.3.1 trivially follows:

Proposition 8.3.1. *If all the n latent variables of a VAE are active, then*

$$\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) = 0.$$

Proof. We know from Theorem 6.3.1 that when a latent representation is only composed of active variables, $\lim_{\gamma \to 0} \boldsymbol{Z} = \boldsymbol{M}$. Thus, $\lim_{\gamma \to 0} \mathrm{Var}[\boldsymbol{Z}] = \mathrm{Var}[\boldsymbol{M}]$ and $\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) = \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$. □

When the latent representation is composed of active and passive variables only, one can decompose the variances into active and passive parts. Then, recalling that passive variables have a variance of 1 in sampled representations and close to 0 in mean representations, and using Proposition 8.3.1 again for the active part, it follows that:

Proposition 8.3.2. *If s of the n latent variables of a VAE are passive and the remaining $n - s$ variables are active, then*

$$\lim_{\gamma \to 0} \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z}]) - \operatorname{Tr}(\operatorname{Var}[\boldsymbol{M}]) = s.$$

Proof. Let us consider the partitioned matrices

$$\operatorname{Var}[\boldsymbol{Z}] = \left[\begin{array}{c|c} \operatorname{Var}[\boldsymbol{Z_a}] & \operatorname{Cov}[\boldsymbol{Z_a}, \boldsymbol{Z_p}] \\ \hline \operatorname{Cov}[\boldsymbol{Z_p}, \boldsymbol{Z_a}] & \operatorname{Var}[\boldsymbol{Z_p}] \end{array}\right], \tag{8.7}$$

and

$$\operatorname{Var}[\boldsymbol{M}] = \left[\begin{array}{c|c} \operatorname{Var}[\boldsymbol{M_a}] & \operatorname{Cov}[\boldsymbol{M_a}, \boldsymbol{M_p}] \\ \hline \operatorname{Cov}[\boldsymbol{M_p}, \boldsymbol{M_a}] & \operatorname{Var}[\boldsymbol{M_p}] \end{array}\right], \tag{8.8}$$

where $\cdot_{\boldsymbol{a}}$ and $\cdot_{\boldsymbol{p}}$ denote the subsets of active and passive variables, respectively. We know from Theorem 6.3.1 that $\lim_{\gamma \to 0} \operatorname{Var}[\boldsymbol{Z_p}] = \boldsymbol{I}_{s \times s}$ and $\lim_{\gamma \to 0} \operatorname{Var}[\boldsymbol{M_p}] = \boldsymbol{0}_{s \times s}$. Thus,

$$\begin{aligned} \lim_{\gamma \to 0} \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z}]) &= \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}]) + \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_p}]), && (8.9) \\ &= \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}]) + \operatorname{Tr}(\boldsymbol{I}_{s \times s}), && (8.10) \\ &= \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}]) + s, && (8.11) \end{aligned}$$

and

$$\begin{aligned} \lim_{\gamma \to 0} \operatorname{Tr}(\operatorname{Var}[\boldsymbol{M}]) &= \operatorname{Tr}(\operatorname{Var}[\boldsymbol{M_a}]) + \operatorname{Tr}(\operatorname{Var}[\boldsymbol{M_p}]), && (8.12) \\ &= \operatorname{Tr}(\operatorname{Var}[\boldsymbol{M_a}]) + \operatorname{Tr}(\boldsymbol{0}_{s \times s}), && (8.13) \\ &= \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}]). && (8.14) \end{aligned}$$

Combining equations 8.11 and 8.14 and recalling from Proposition 8.3.1 that

$$\lim_{\gamma \to 0} \operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}]) = \operatorname{Tr}(\operatorname{Var}[\boldsymbol{M_a}]),$$

we obtain

$$\lim_{\gamma\to 0}\Big(\operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z}])-\operatorname{Tr}(\operatorname{Var}[\boldsymbol{M}])\Big)=\operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}])+s-\operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z_a}])=s, \quad (8.15)$$

as expected. □

While mixed variables (i.e., variables that are passive for some input and active for others) are less trivial to analyse, using the same techniques as Proposition 8.3.2, and recalling that mixed variables come from a mixture distribution, we have:

Proposition 8.3.3. *If s of the n latent variables of a VAE are mixed and the remaining $n-s$ variables are active, then* $0<\lim_{\gamma\to 0}\operatorname{Tr}(\operatorname{Var}[\boldsymbol{Z}])-\operatorname{Tr}(\operatorname{Var}[\boldsymbol{M}])<s$.

Proof. Let us consider the partitioned matrices

$$\operatorname{Var}[\boldsymbol{Z}]=\left[\begin{array}{c|c}\operatorname{Var}[\boldsymbol{Z_a}] & \operatorname{Cov}[\boldsymbol{Z_a},\boldsymbol{Z_m}]\\ \hline \operatorname{Cov}[\boldsymbol{Z_m},\boldsymbol{Z_a}] & \operatorname{Var}[\boldsymbol{Z_m}]\end{array}\right], \quad (8.16)$$

and

$$\operatorname{Var}[\boldsymbol{\mu}]=\left[\begin{array}{c|c}\operatorname{Var}[\boldsymbol{M_a}] & \operatorname{Cov}[\boldsymbol{M_a},\boldsymbol{M_m}]\\ \hline \operatorname{Cov}[\boldsymbol{M_m},\boldsymbol{M_a}] & \operatorname{Var}[\boldsymbol{M_m}]\end{array}\right], \quad (8.17)$$

where $\cdot_{\boldsymbol{a}}$ and $\cdot_{\boldsymbol{m}}$ denote the subsets of active and mixed variables, respectively. We know from Theorem 6.3.1 that up to some permutations a mixed variable $\boldsymbol{Z}_i=\left[\boldsymbol{Z}_i^{(\boldsymbol{a})},\boldsymbol{Z}_i^{(\boldsymbol{p})}\right]$, where $\boldsymbol{a}$ is the subset of data examples indexes for which the variable is active and $\boldsymbol{p}$ the subset of data example indexes for which the variable is passive. Given h data examples, we thus have $h=\operatorname{card}(\boldsymbol{a})+\operatorname{card}(\boldsymbol{b})$ where $\operatorname{card}(\cdot)$ denotes the cardinality of a set. Let us define $w_i\triangleq\frac{\operatorname{card}(\boldsymbol{a})}{h}$ with $0<w_i<1$. We have $(1-w_i)=\frac{h-\operatorname{card}(\boldsymbol{a})}{h}=\operatorname{card}(\boldsymbol{b})$. Given the mean of the mixed variable i of the sampled representation $\bar{\boldsymbol{Z}}_i$, we thus obtain:

$$\lim_{\gamma\to 0}\bar{\boldsymbol{Z}}_i=w_i\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}+(1-w_i)\bar{\boldsymbol{Z}}_i^{(\boldsymbol{p})}=w_i\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}. \quad (8.18)$$

Now, let us calculate the variance of $\boldsymbol{Z}_i$, $\operatorname{Var}[\boldsymbol{Z}_i]$:

$$\lim_{\gamma\to 0}\operatorname{Var}[\boldsymbol{Z}_i]=w_i\Big(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}-\bar{\boldsymbol{Z}}_i\Big)^2+(1-w_i)w_i\Big(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{p})}-\bar{\boldsymbol{Z}}_i\Big)^2 \quad (8.19)$$

$$+w_i\operatorname{Var}\left[\boldsymbol{Z}_i^{(\boldsymbol{a})}\right]+(1-w_i)\operatorname{Var}\left[\boldsymbol{Z}_i^{(\boldsymbol{p})}\right], \quad (8.20)$$

$$=w_i(1-w_i)^2\Big(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}\Big)^2+w_i^2(1-w_i)\Big(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}\Big)^2 \quad (8.21)$$

$$+ w_i \mathrm{Var}\left[\boldsymbol{Z}_i^{(\boldsymbol{a})}\right] + 1 - w_i, \tag{8.22}$$

$$= w_i(1-w_i)\left(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}\right)^2 + w_i \mathrm{Var}\left[\boldsymbol{Z}_i^{(\boldsymbol{a})}\right] + 1 - w_i, \tag{8.23}$$

$$= \kappa_i + 1 - w_i, \tag{8.24}$$

where $\kappa_i = w_i(1-w_i)\left(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}\right)^2 + w_i \mathrm{Var}\left[\boldsymbol{Z}_i^{(\boldsymbol{a})}\right]$. Writing $\boldsymbol{M}_i = \left[\boldsymbol{M}_i^{(\boldsymbol{a})}, \boldsymbol{M}_i^{(\boldsymbol{p})}\right]$, we obtain in the same way

$$\lim_{\gamma \to 0} \bar{\boldsymbol{M}}_i = w_i \bar{\boldsymbol{M}}_i^{(\boldsymbol{a})} + (1-w_i)\bar{\boldsymbol{M}}_i^{(\boldsymbol{p})} = w_i \bar{\boldsymbol{M}}_i^{(\boldsymbol{a})} = w_i \bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}, \tag{8.25}$$

and

$$\lim_{\gamma \to 0} \mathrm{Var}[\boldsymbol{M}_i] = w_i\left(\bar{\boldsymbol{M}}_i^{(\boldsymbol{a})} - \bar{\boldsymbol{M}}_i\right)^2 + (1-w_i)w_i\left(\bar{\boldsymbol{M}}_i^{(\boldsymbol{p})} - \bar{\boldsymbol{M}}_i\right)^2 \tag{8.26}$$

$$+ w_i \mathrm{Var}\left[\boldsymbol{M}_i^{(\boldsymbol{a})}\right] + (1-w_i)\mathrm{Var}\left[\boldsymbol{M}_i^{(\boldsymbol{p})}\right], \tag{8.27}$$

$$= w_i(1-w_i)\left(\bar{\boldsymbol{M}}_i^{(\boldsymbol{a})}\right)^2 + w_i \mathrm{Var}\left[\boldsymbol{M}_i^{(\boldsymbol{a})}\right], \tag{8.28}$$

$$= w_i(1-w_i)\left(\bar{\boldsymbol{Z}}_i^{(\boldsymbol{a})}\right)^2 + w_i \mathrm{Var}\left[\boldsymbol{Z}_i^{(\boldsymbol{a})}\right], \tag{8.29}$$

$$= \kappa_i. \tag{8.30}$$

Using Equation 8.24, we have

$$\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) = \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}_{\boldsymbol{a}}]) + \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}_{\boldsymbol{m}}]), \tag{8.31}$$

$$= \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}_{\boldsymbol{a}}]) + \sum_{i=1}^{s} (\kappa_i + 1 - w_i). \tag{8.32}$$

In the same way, from Equation 8.30 we obtain

$$\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) = \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}_{\boldsymbol{a}}]) + \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}_{\boldsymbol{m}}]), \tag{8.33}$$

$$= \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}_{\boldsymbol{a}}]) + \sum_{i=1}^{s} \kappa_i. \tag{8.34}$$

Thus, $\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) = \sum_{i=1}^{s}(1-w_i)$. Given that for all i, $0 < w_i < 1$, for the s mixed variables $0 < \sum_{i=1}^{s}(1-w_i) < s$, and $0 < \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) < s$, as required. $\square$

We are now interested in how the difference of the traces evolves when we increase the number of latent dimensions. When we have n active variables, we know form Proposition 8.3.1 that the difference of trace will be close to 0. Using Propositions 8.3.2 and 8.3.3 with $s = 1$, we know that if the next variable is not active, the difference will increase by at most 1.

Theorem 8.3.1. *n is the maximal number of dimensions for which a VAE contains only active variables if*

- *for n latent variables,* $\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) = 0$,
- *and for $n+1$ latent variables,* $0 < \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant 1$.

Proof. We know from Proposition 8.3.1 that if all the variables are active, $\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) = 0$. Once their maximum number n is reached, the next variable learned will be either passive or mixed. Recall from Proposition 8.3.2 that for $s = 1$ passive variables, $\lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) = 1$. Moreover, using Proposition 8.3.3 with $s = 1$ mixed variables, $0 < \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) < 1$. As a result, if n is the maximum number of active variables, at $n+1$, $0 < \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant 1$, where the upper bound is tight when the $(n+1)^{th}$ variable is passive. □

When we have n non-passive variables, using Proposition 8.3.3 with $s \leqslant n$, the difference of the trace will be between higher than 0 but lower than n. Using Proposition 8.3.2 with $s = 1$, we know that if the next variable is passive, the difference will increase by 1.

Theorem 8.3.2. *n is the maximal number of dimensions for which a VAE contains only non-passive variables if*

- *for n latent variables,* $0 \leqslant \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) < n$,
- *and for $n+1$ latent variables,* $1 \leqslant \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) < n+1$.

Proof. Here we consider the case where n is the maximum number of mixed and active variables that can be reached. Using Proposition 8.3.3 with $0 \leqslant s \leqslant n$, at n we have:

$$0 \leqslant \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) < n, \tag{8.35}$$

where the lower bound is tight when $s = 0$. As the $(n+1)^{th}$ variable will always be passive, using Proposition 8.3.2 with $s = 1$, we obtain

$$1 \leqslant \lim_{\gamma \to 0} \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) < n+1, \tag{8.36}$$

as expected. □

A useful property of Theorems 8.3.1 and 8.3.2 is that they hold very early in training. Indeed, the variance of the decoder very quickly approaches 0 (Dai and Wipf, 2018; Dai et al., 2018) leading to observations of the polarised regime after a few epochs, as we have observed in Chapter 7. This allows Theorems 8.3.1 and 8.3.2 to provide stable estimates using mean and sampled representations which reflect accurately the number of active and passive variables present in the final model. While these theorems can be useful by themselves to study the type of representations learned by an already trained VAE, we believe that they can also help to design new tools to improve models' quality. For example, by providing an estimation of the maximum number of non-passive latent dimensions that a VAE can reach for a given dataset. We will see in Section 8.3.2 that this can be done in an unsupervised way, without fully training any models or manually applying the Elbow method.

8.3.2 *Finding the number of dimensions by unsupervised estimation*

As discussed in Section 8.3.1, the traces of the covariance matrices of the mean and sampled representations start to diverge when non-active variables appear. We can thus use Theorems 8.3.1 and 8.3.2 to find the number of latent dimensions retaining the most information while remaining highly compressed (i.e., no passive variables).

For example, let us consider a threshold $t = 1$. We know from Theorem 8.3.1 that if the difference between traces is lower than or equal to t for n latent dimensions but higher than t for $n + 1$ latent dimensions, then n is the largest number of dimensions for which the latent representation contains only active variables. If, on the other hand we allow t to be higher, the model can encode additional non-active variables, as per Theorem 8.3.2. After selecting a suitable value for t depending on the desired level of compression, we can thus iteratively check the difference between traces for different values of n after training each model for a few steps until we obtain the largest n for which the difference is lower than the threshold t that we defined. To this aim, we propose an algorithm to FONDUE.

Theorem 8.3.3. *Any execution of FONDUE (Algorithm 1) returns the largest number of dimensions n for which $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$:*

- *If $t < 1$, FONDUE returns the maximal number of latent dimensions for which a VAE contains only active variables.*
- *If $t \geqslant 1$, FONDUE returns the maximal number of latent dimensions for which a VAE contains only non-passive variables.*

In Algorithm 1, we define a lower and upper bound of n, l and u, and update the predicted number of latent variables n until, after i iterations, $n_i = l_i$. Using the loop invariant $l_i \leqslant n_i \leqslant u_i$, we can show that the algorithm terminates when $l_i = n_i = \text{floor}\left(\frac{l_i+u_i}{2}\right)$, which can only be reached when $u_i = n_i + 1$, that is, when n_i is the maximum number of latent dimensions for which we have $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$. The threshold t is determined according to Theorems 8.3.1 and 8.3.2, as discussed above. Indeed, recall from Theorems 8.3.1 and 8.3.2 that the mean and sampled representations start to diverge only after the number of latent dimensions has become large enough for non-active variables to appear.

Remark. Given that l and u only take values of latent dimensions for which $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$ and $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) > t$, respectively, Theorems 8.3.1 and 8.3.2 imply that for all iterations i, $l_i < u_i$.

Using the loop invariant $l_i \leqslant n_i \leqslant u_i$ for each iteration i, we will now show that Algorithm 1 terminates when $l_i = n_i = \text{floor}\left(\frac{l_i+u_i}{2}\right)$, which can only be reached when $u_i = n_i + 1$, that is when n_i is the maximum number of latent dimensions for which we have $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$.

Proof.

Initialisation: $l_0 = 0, n_0 = IDE_{data}, u_0 = \infty$, thus $l_0 < n_0 < u_0$.

Maintenance: We will consider both branches of the if statement separately:

- For $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$ (lines 9-11), $u_i = u_{i-1}, n_i = min(n_{i-1} \times 2, u_i)$, and $l_i = n_{i-1}$. We directly see that $n_i \leqslant u_i$. We know from Remark 8.3.2 that $l_i < u_i$ and we also have $l_i < n_{i-1} \times 2$, it follows that $l_i < n_i$. Grouping both inequalities, we get $l_i < n_i \leqslant u_i$.

- For $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) > t$ (lines 12-14), $u_i = p_{i-1}, l_i = l_{i-1}$, and $n_i = \text{floor}\left(\frac{l_i+u_i}{2}\right)$. Using Remark 8.3.2 we can directly see that $l_i \leqslant \text{floor}\left(\frac{l_i+u_i}{2}\right) < u_i$ and we obtain $l_i \leqslant n_i < u_i$.

Termination: The loop terminates when $l_i = n_i$. Given that $l_i < n_i$ when $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$, this is only possible when $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) > t$, which is when $n_i = \text{floor}\left(\frac{l_i+u_i}{2}\right)$. We know from Remark 8.3.2 that $l_i < u_i$, so we must have $(l_i + u_i) \bmod 2 > 0$. As $a \bmod 2 \in \{0, 1\}$, the only possible value for u_i to satisfy $l_i = n_i = \text{floor}\left(\frac{l_i+u_i}{2}\right)$ is $u_i = n_i + 1$. Thus, n_i is the largest number of latent dimensions for which $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$. $\square$

How does FONDUE work? FONDUE will seek to reach the maximum number of dimensions for which the difference between $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}])$ and $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$ is lower than or equal to the threshold t, as illustrated in Figure 8.1. The number of dimensions is first initialised to the IDE of the dataset IDE_{data} to start from a reasonable number of latents. Note that a random initialisation will not impact the number of dimensions predicted by FONDUE but may slow the algorithm down if the value is very far from the predicted number of dimensions. Indeed, FONDUE will need more iterations to converge in that case. However, as shown in Section 8.6, even with extreme initialisation values, FONDUE remains faster than fully training one model. After initialisation, at each iteration, FONDUE will train a VAE for a few epochs (generally just two) and retrieve the mean and sampled representations corresponding to 10,000 data examples. We then compute the traces of the covariance matrices of the mean and sampled representations (i.e., $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}])$ and $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$) and the difference between them. If this difference is lower than or equal to the threshold, we set our lower bound to the current number of latent dimensions and train a VAE again with twice the number of latents, as illustrated in Figure 8.2. If the difference is higher than the threshold, we set the current number of latent dimensions to our upper bound and train a VAE again with half of the sum of the lower and upper bound, as illustrated in Figure 8.3. We iterate these two steps until our current number of latent dimensions is the largest possible dimensionality for which the difference is smaller than or equal to the threshold.

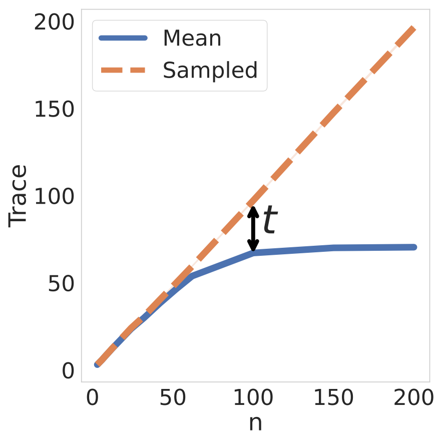


Figure 8.1: Traces of the covariance matrices of the mean and sampled representations of VAEs trained on Celeba with an increasing number of latent dimensions n. FONDUE retrieves the largest n for which the difference is smaller or equal to a given threshold t.

Extending FONDUE Note that one could use different flavours of FONDUE by replacing the difference $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$ by any metric as long as its score is 1) consistently lower than some threshold t before a "good" n and higher or equal afterwards, 2) stable early in the training. One such example is $IDE_{\mathbf{z}} - IDE_{\boldsymbol{\mu}}$ which generally provides consistent results with the original algorithm as shown in Section 8.8. We will see in Section 8.4.2 that the information theoretic metric of Boquet et al. (2021) can also be readily integrated into FONDUE.

Algorithm 1 FONDUE

```
 1: procedure FONDUE($t, IDE_{data}, e$)
 2:     $l \leftarrow 0$                                        ▷ Lower bound
 3:     $u \leftarrow \infty$                                        ▷ Upper bound
 4:     $n \leftarrow IDE_{data}$                     ▷ Current number of latent dimensions
 5:     $m \leftarrow \{\}$
 6:     while $n \neq l$ do
 7:         $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]), \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leftarrow$ GET-MEM($m, n, e$)
 8:         if $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$ then          ▷ Figure 8.2
 9:             $l \leftarrow n$
10:             $n \leftarrow \min(n \times 2, u)$
11:         else                                                ▷ Figure 8.3
12:             $u \leftarrow n$
13:             $n \leftarrow \mathrm{floor}\left(\frac{l+u}{2}\right)$
14:         end if
15:     end while
16:     return $n$
17: end procedure
```

Algorithm 2 GET-MEM

```
1: procedure GET-MEM($m, n, e$)
2:     if $m[n] = \emptyset$ then
3:         $vae \leftarrow$ TRAIN-VAE($dim = n, n\_epochs = e$)
4:         $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]), \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leftarrow Traces(vae)$
5:         $m[n] \leftarrow \mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]), \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$
6:     end if
7:     return $m[n]$
8: end procedure
```

8.4 EXPERIMENTAL SETUP

After computing the IDEs of the datasets on which the models will be trained, we will use them to initialise FONDUE. We will then assess the performance of FONDUE by

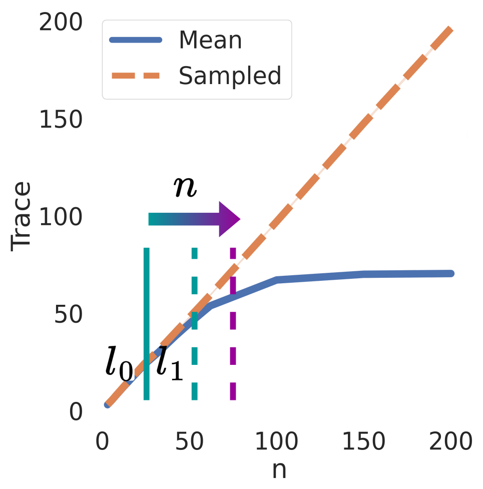


Figure 8.2: Update l and increase n until $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) > t$.

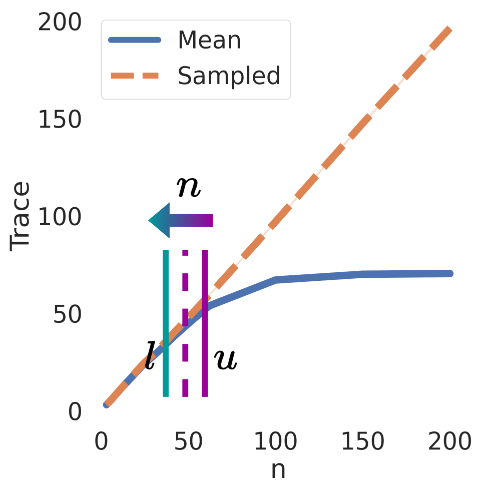


Figure 8.3: Update u and decrease n until $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}]) \leqslant t$.

comparing it with the existing techniques of dimension selection discussed in Section 8.2.5. This will be done by ensuring that the number of dimensions selected by the Elbow method for reconstruction and downstream tasks is consistent with the values proposed by FONDUE, and by comparing the results obtained with FONDUE and the IB algorithm of Boquet et al. (2021).

8.4.1 *General setup*

Datasets We use three datasets with an increasing number of intrinsic dimensions: Symmetric solids (Murphy et al., 2021), dSprites (Higgins et al., 2017), and Celeba (Liu et al., 2015). The numbers of generative factors of the first two datasets are 2 and 5, respectively, and the IDE of these two datasets should be close to these values. While we do not know the generative factors of Celeba, Pope et al. (2021) reported an IDE of 26 when using $k = 20$ neighbours with MLE, indicating a higher number of intrinsic dimensions than dSprites and Symsol.

Data preprocessing Each image is resized to $64 \times 64 \times c$, where $c = 1$ for Symmetric solids and dSprites, and $c = 3$ for Celeba. We also removed duplicate images (i.e., cases where different rotations resulted in the same image) and labels from Symmetric solids and created a reduced version: `symsol_reduced` which is available at `https://data.kent.ac.uk/436`.

VAE training We use the β-VAE architecture detailed in Higgins et al. (2017) for all the datasets, together with the standard learning objective of VAEs, as presented in Equation 3.5. Each VAE is trained 5 times with a number of latent dimensions $n = 3, 6, 8, 10, 12, 18, 24, 32, 42$ on every dataset. For Celeba, which has the highest IDE, we additionally train VAEs with latent dimensions $n = 52, 62, 100, 150, 200$.

Estimations of the ID For all the datasets, we estimate the ID using 10,000 data examples. As in Pope et al. (2021), the MLE scores are computed with an increasing number of neighbours $k = 3, 5, 10, 20$. Moreover, we repeat the MLE computations 3 times with different seeds to detect any variance in estimates.

Downstream tasks To monitor how good the learned latent representations are on downstream tasks, we train gradient boosted trees to classify the labels of each dataset based on the mean representations. Each label is learned as a separate classification task, and we evaluate the results on a dataset based on the averaged test accuracy over all these tasks, similarly to (Locatello et al., 2019b,a).

8.4.2 *Comparing FONDUE with the IB algorithm*

As the IB algorithm is the most closely related to FONDUE, we will compare the results of both algorithms. However, we will see that a few modifications to the IB algorithm are required to ensure a fair comparison in the unsupervised setting. For completeness, we also combine the results of Theorems 8.3.1 and 8.3.2 with binary search to compare with the IB algorithm in the supervised setting in Appendix D.3.

Why the IB algorithm cannot be directly compared with FONDUE Because the IB algorithm performs binary search over a user-defined array of likely dimensions, its execution time is greatly dependent on the size of this array. Indeed, each time the dimension selected in the IB algorithm is lower than n^*, the corresponding model is trained until convergence. Thus, if we select a range of values from 1 to n such that $floor(\frac{n-1}{2})$ is lower than n^* we will always have at least one full model training, which, for the tested dataset is slower than FONDUE. For example, if $n^* = 30$, any $n < 61$ would trigger at least one full model training. Comparing the execution time of FONDUE and the original IB algorithm will thus mostly be based on whether the binary search encounters a situation where $n \geqslant n^*$ and require to fully train one or more models or not. Moreover, the original IB algorithm will give a value that depends on the range selected. For example, if $n^* = 30$ but the selected range is from 1 to 20, the original IB algorithm will return 20. Thus, manually selecting the results could also impact the quality of the predicted number of dimensions. For example one could force the IB algorithm to provide good predictions by selecting a very small range of values consistent with the Elbow methods while a larger range of values would provide worse predictions. To summarise, the human supervision plays an important role in the quality of the IB algorithm predictions, making it unpractical to compare with a fully unsupervised algorithm such as FONDUE. We thus propose to use a unified algorithm which does not require to define a range of values to circumvent this issue. As mentioned above, we also compare the original IB algorithm where all models are trained for a fixed number of epochs and a binary search for a fixed range of the number of dimensions using the difference between traces obtained in Theorem 8.3.3 in Appendix D.3. Appendix D.3 is thus a comparison between the IB algorithm and Theorem 8.3.3 in a supervised context, where the range of dimensions to search is manually defined.

Using FONDUE$_{IB}$ for a fair comparison The IB algorithm relies on the assumption that if the current number of dimensions n is lower than the target number of dimensions n^*, $\mathrm{H}(\mathbf{z}) - \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) < 0$ and $\mathrm{H}(\mathbf{z}) - \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \approx 0$ otherwise. One can thus directly use this inequality to provide an alternative version of FONDUE, FONDUE$_{IB}$, which will also be completely unsupervised and will not require to fully train one or more models. This is done by replacing $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}])$ and $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$ by $\mathrm{H}(\mathbf{z})$ and $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}})$ in the original algorithm, as illustrated in Algorithms 3 and 4. In Boquet et al. (2021), the scores of $\mathrm{H}(\mathbf{z})$ and $\mathrm{I}(\mathbf{x}, \hat{\mathbf{x}})$ are truncated to 2 decimals, making $\mathrm{H}(\mathbf{z}) - \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) < 0$ equivalent to $\mathrm{H}(\mathbf{z}) - \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \leqslant -0.01$. We thus set the threshold to $t = -0.01$ for FONDUE$_{IB}$ and keep the original truncation. As in Boquet et al. (2021), we use an RBF kernel and set $\alpha = 2$.

Additional details on our implementation can be found in Appendix D.1 and our code is available at https://github.com/bonheml/VAE_learning_dynamics, and a demonstration of FONDUE is available in a notebook at https://github.com/bonheml/fondue-demo.

8.5 RESULTS

In this section, we will analyse the results of the experiments detailed in Section 8.4. First, we will review the IDE of the different datasets in Section 8.5.1 obtained from MLE and TwoNN, the two ID estimators discussed in Section 8.2.1. Then, in Section 8.5.2, we will evaluate the results of FONDUE by comparing the selected number of dimensions with the reconstruction loss and downstream task accuracy.

Algorithm 3 FONDUE$_{IB}$

```
 1: procedure FONDUE$_{IB}(t, IDE_{data}, e)$
 2:     $l \leftarrow 0$
 3:     $u \leftarrow \infty$
 4:     $n \leftarrow IDE_{data}$
 5:     $m \leftarrow \{\}$
 6:     while $n \neq l$ do
 7:         $\mathrm{H}(\mathbf{z}),\ \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \leftarrow \text{GET-MEM}_{\text{IB}}(m, n, e)$
 8:         if $\mathrm{H}(\mathbf{z}) - \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \leqslant t$ then
 9:             $l \leftarrow n$
10:             $n \leftarrow \min(n \times 2, u)$
11:         else
12:             $u \leftarrow n$
13:             $n \leftarrow \text{floor}\left(\frac{l+u}{2}\right)$
14:         end if
15:     end while
16:     return $n$
17: end procedure
```

Algorithm 4 GET-MEM$_{IB}$

```
1: procedure GET-MEM$_{IB}(m, n, e)$
2:     if $m[n] = \emptyset$ then
3:         $vae \leftarrow \text{TRAIN-VAE}(dim = n, n\_epochs = e)$
4:         $\mathrm{H}(\mathbf{z}),\ \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \leftarrow IB(vae)$
5:         $m[n] \leftarrow \mathrm{H}(\mathbf{z}),\ \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}})$
6:     end if
7:     return $m[n]$
8: end procedure
```

8.5.1 *Estimating the intrinsic dimensions of the datasets*

As mentioned in Section 8.4, we have selected 3 datasets of increasing intrinsic dimensionality: Symsol (Murphy et al., 2021), dSprites (Higgins et al., 2017), and Celeba (Liu et al., 2015). Following Karbauskaitė, Dzemyda and Mazėtis (2011), we will retain for our analysis the MLE estimates which are stable for the largest number of k values. We can see in Figure 8.4 that the MLE estimations become stable when k is between 10 and 20, similar to what was reported by Levina and Bickel (2004). These IDEs are also generally close to TwoNN estimations, except for Celeba, where TwoNN seems to overestimate the ID, as reported by Pope et al. (2021).

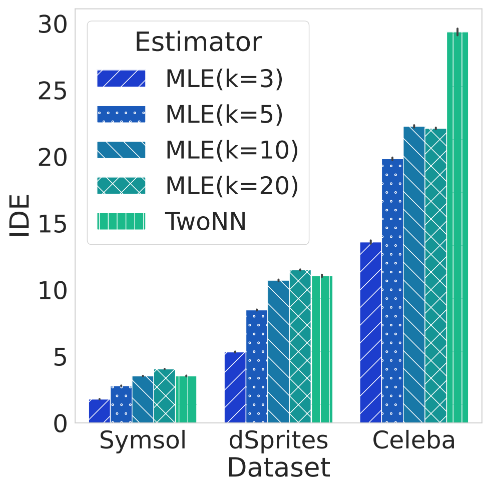


Figure 8.4: IDEs of dSprites, Celeba, and Symsol using different ID estimation methods.

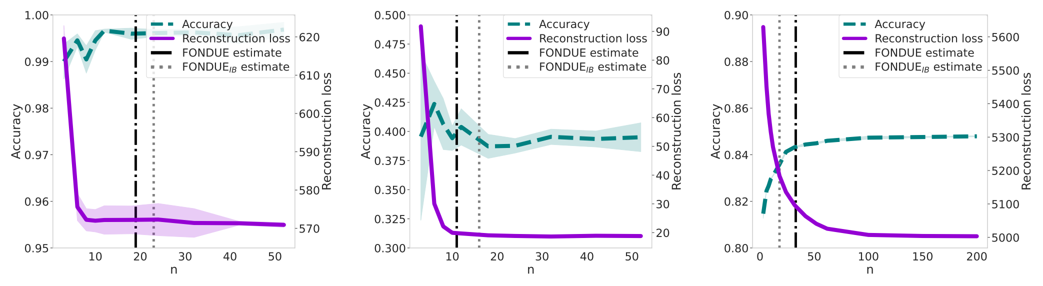
Celeba's IDE was previously estimated to be 26 for MLE with $k = 20$ (Pope et al., 2021) neighbours, and we know that Symsol and dSprites have 2 and 5 generative factors, respectively. We thus expect their IDEs to be close to these values. We can see in Figure 8.4 that MLE and TwoNN overestimate the IDs of Symsol and dSprites, with IDEs of 4 and 11 instead of the expected 2 and 5. Our result for Celeba is close to Pope et al. (2021) with an estimate of 22; the slight difference may be attributed to the difference in our averaging process (Pope et al. (2021) used the averaging described by MacKay and Ghahramani (2005) instead of the original averaging of Levina and Bickel (2004)).

Because stability is more important than overestimation for our purposes, in the rest of this chapter, we will consider the IDEs obtained from MLE with $k = 20$. Specifically, we will initialise FONDUE with $n = 4$ for Symsol, $n = 11$ for dSprites and $n = 22$ for Celeba.

8.5.2 *Evaluating FONDUE*

Using the IDEs reported in the previous section to initialise FONDUE, we will now report the results of FONDUE and FONDUE$_{IB}$ for the considered datasets.

Obtaining stable estimates To ensure stable estimates, we computed FONDUE multiple times, gradually increasing the number of epochs e until the predicted dimensionality n stopped changing (i.e., until γ is sufficently small). As reported



(a) Symsol (b) dSprites (c) Celeba

Figure 8.5: Reconstruction loss and accuracy obtained for generation and downstream tasks of **VAEs** for Symsol, dSprites, and Celeba with an increasing number of latent variables. The black and grey vertical lines indicate the number of dimensions found by FONDUE and FONDUE$_{IB}$.

in Table 8.1, the results were already stable after two epochs[3]. We set a fixed threshold of $t = 1$ in all our experiments and used memoisation (see Algorithm 2) to avoid unnecessary retraining and speed up Algorithm 1. For FONDUE$_{IB}$ we followed the same process, with a fixed threshold of $t = -0.01$.

Analysing the results of FONDUE As shown in Table 8.1, the execution time for finding the number of dimensions for one dataset is much shorter than for fully training one model, which is approximately 2h using the same GPUs. Moreover, on dSprites and Celeba, FONDUE finds the number of dimensions corresponding to low reconstruction loss and good accuracy, which is consistent with the results obtained manually from the Elbow methods (Doersch, 2016; Mai Ngoc and Hwang, 2020). It is worth noting that the number of dimensions provided by FONDUE for dSprites is also close to the dimensionality of 10 generally used in the literature (Higgins et al., 2017; Burgess et al., 2018; Kim and Mnih, 2018; Locatello et al., 2019a). When compared to FONDUE, FONDUE$_{IB}$ is always slower as it requires more epochs to provide stable estimates. Furthermore, we can see in Figure 8.5 that it displays an inconsistent behaviour, overestimating the number of dimensions for dSprites and underestimating them for Celeba. Both FONDUE and FONDUE$_{IB}$ overestimate the number of dimensions for Symsol, with a stronger overestimation of FONDUE$_{IB}$. Indeed, it is apparent in Figure 8.5a that 15 dimensions would be enough to ensure a low reconstruction error and a high accuracy, but FONDUE overestimates it by 4 dimensions and this overestimation is doubled by FONDUE$_{IB}$. We hypothesise that the performance of both FONDUE versions may be impacted

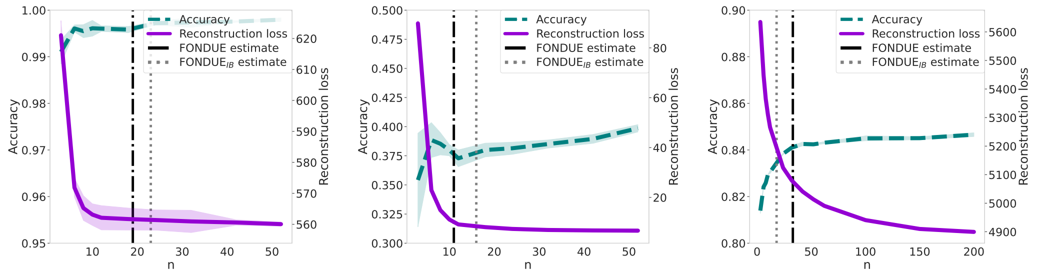
3 Note that the numbers of epochs given in Table 8.1 correspond to the minimum number of epochs needed for FONDUE to be stable. For example, if we obtain the same score after 1 and 2 epochs, the number of epochs given in Table 8.1 is 1.

Table 8.1: Number of latent variables n obtained with FONDUE and FONDUE$_{IB}$. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score.

	Dataset	n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training
FONDUE	Symsol	19.1 ± 0.7	6 min	8	1
FONDUE	dSprites	10.9 ± 0.7	42 min	4	2
FONDUE	Celeba	32.6 ± 0.7	17 min	6	2
FONDUE$_{IB}$	Symsol	23.3 ± 0.5	40 min	8	6
FONDUE$_{IB}$	dSprites	15.7 ± 0.5	52 min	5	2
FONDUE$_{IB}$	Celeba	18.4 ± 0.7	21 min	5	3

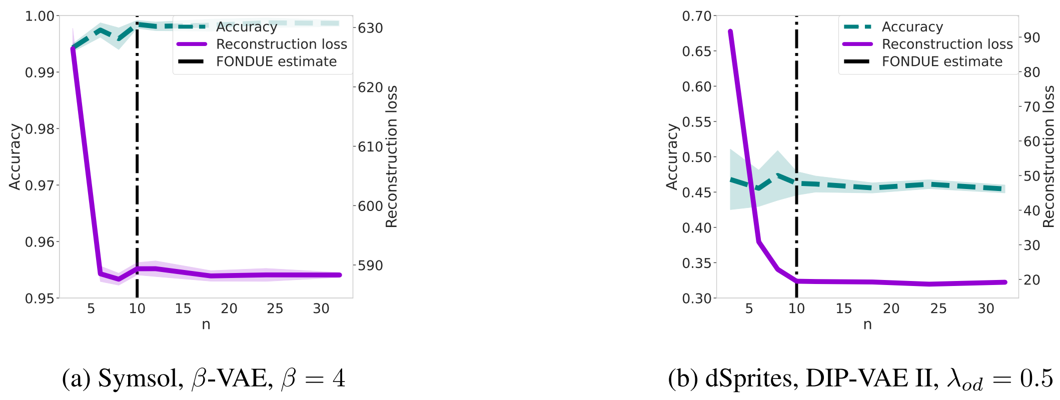
by noisy environments as the VAEs trained on Symsol present the largest variance of reconstruction loss. Despite this, Figure 8.5 shows that FONDUE estimates generally locate the Elbow point correctly, ignoring diminishing returns for more complex datasets like Figure 8.5c. It thus consistently provides a number of dimensions corresponding to a good trade-off between low reconstruction loss and high accuracy, which is not always the case with FONDUE$_{IB}$.

Are the results obtained with FONDUE on VAEs applicable to deterministic AEs? We can see in Figure 8.6 that FONDUE estimates obtained on VAEs also agree with the Elbow method for deterministic AEs with equivalent architectures. Overall, these results indicate that the dimensionality selected by FONDUE on VAEs can be reused for AEs trained on the same dataset with an identical architecture.



(a) Symsol (b) dSprites (c) Celeba

Figure 8.6: Reconstruction loss and accuracy obtained for generation and downstream tasks of **deterministic AEs** for Symsol, dSprites, and Celeba with an increasing number of latent variables. The black and grey vertical lines indicate the number of dimensions found by FONDUE and FONDUE$_{IB}$ for VAEs with the same architectures.



(a) Symsol, β-VAE, $\beta = 4$

(b) dSprites, DIP-VAE II, $\lambda_{od} = 0.5$

Figure 8.7: Reconstruction loss and accuracy obtained for generation and downstream tasks with an increasing number of latent variables. (a) shows the results for β-VAE with $\beta = 4$ on Symsol, and (b) shows the results for DIP-VAE II with $\lambda_{od} = 0.5$ on dSprites. The vertical lines indicate the number of dimensions found by FONDUE.

Can FONDUE be applied to other architectures, hyperparameters and learning objectives? To assess the generalisability of FONDUE to other hyperparameters and learning objectives, we compared the results obtained by FONDUE with the Elbow methods for β-VAEs with $\beta = [0.5, 2, 4]$ and for DIP-VAE II (Kumar, Sattigeri and Balakrishnan, 2018), with hyperparameter values $\lambda_{od} = [0.5, 1, 2, 4]$. We can see in Figure 8.7 that FONDUE generalises well to different hyperparameter values, and learning objectives, with results on par with the Elbow method, as before. As shown in Tables 8.2 and 8.3, the number of dimensions predicted are consistent with the pressure applied on the bottleneck: models with higher β (or λ_{od}) apply a more agressive pruning and have a lower number of active variables than models with lower β (or λ_{od}). Overall, the results obtained are still faster to compute than fully training one model. Additional results for different learning objectives, hyperparameters and datasets can be found in Appendix D.4. FONDUE also seems to be robust to architectural changes as reported in Appendix 8.7.

8.6 IMPACT OF THE INITIALISATION

In Section 8.5.2, we used the IDE of the dataset as the initial number of dimensions. To show the impact of this choice on FONDUE, we run the algorithm with 10 randomly selected initial numbers of dimensions chosen between 1 and 200 on 10 different seeds. We can see in Tables 8.4, 8.5 and 8.6 that the initialisation does not change the number of dimensions predicted with IDE initialisation in Table 8.1, but the execution time can be longer when the initial number of dimensions is further away from the

Table 8.2: Number of latent variables n obtained with FONDUE with different β values for β-VAE. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score.

β	Dataset	n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training
0.5	Symsol	26 ± 0.7	18 min	8	3
0.5	dSprites	13.6 ± 1.2	80 min	6	3
0.5	Celeba	49.9 ± 1.5	46 min	8	4
2	Symsol	13.6 ± 0.8	4 min	6	1
2	dSprites	7.4 ± 0.5	26 min	5	1
2	Celeba	21.9 ± 0.3	8 min	6	1
4	Symsol	10 ± 0.5	4 min	6	1
4	dSprites	6.3 ± 0.7	24 min	5	1
4	Celeba	15.9 ± 1.0	23 min	6	3

Table 8.3: Number of latent variables n obtained with FONDUE with different λ_{od} values for DIP-VAE II. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score.

λ_{od}	Dataset	n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training
0.5	Symsol	17.5 ± 1.1	7 min	8	1
0.5	dSprites	10.2 ± 0.6	19 min	4	1
0.5	Celeba	27.3 ± 1.7	8 min	6	1
1	Symsol	15.9 ± 2.0	7 min	7	1
1	dSprites	9.5 ± 1.0	25 min	5	1
1	Celeba	22.7 ± 2.3	8 min	6	1
2	Symsol	12.4 ± 1.1	7 min	6	1
2	dSprites	7.9 ± 0.7	55 min	6	2
2	Celeba	20.9 ± 1.8	16 min	6	2
4	Symsol	11.7 ± 0.9	13 min	6	2
4	dSprites	8.1 ± 0.7	28 min	6	1
4	Celeba	18.7 ± 1.4	23 min	6	3

predicted number of dimensions. Despite this, all these results remain below the average training time needed for one model on the same GPU (around 2 hours). To conclude, while using the data IDE as the initial number of dimensions does not change the results of FONDUE, it allows the algorithm to start closer to the predicted number of dimensions and thus can shorten its running time.

Table 8.4: Number of latent variables n obtained with FONDUE with random initial numbers of dimensions n_{init} on Symsol. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score. The last line is the average over all initial numbers of dimensions.

n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training	n_{init}
19.3 $\pm$ 0.5	6 min	7	1	5
18.2 $\pm$ 0.4	4 min	4	1	19
18.7 $\pm$ 0.8	4 min	5	1	22
19.0 $\pm$ 0.8	5 min	6	1	39
18.9 $\pm$ 0.9	7 min	8	1	74
19.0 $\pm$ 0.8	8 min	9	1	94
18.7 $\pm$ 0.5	8 min	9	1	116
19.3 $\pm$ 0.7	7 min	8	1	145
19.1 $\pm$ 0.6	7 min	8	1	180
18.8 $\pm$ 0.9	8 min	9	1	182
18.9 $\pm$ 0.7	6 min	8	1	

8.7 FONDUE ON FULLY-CONNECTED ARCHITECTURES

We report the results obtained by FONDUE for fully-connected (FC) architectures in Table 8.7 and Figure 8.8. As shown in Table 8.7, the execution time for estimating the number of dimensions for one dataset is much shorter than for training one model (this is approximately 2h on the same GPUs), consistently with convolutional VAEs. As in Section 8.5.2, FONDUE correctly finds the number of latent dimensions that would be selected by Elbow methods, as shown in Figure 8.8. FONDUE$_{IB}$ predictions are more consistent with those of FONDUE for FC architectures, except for dSprites where the number of dimensions is largely overestimated.

As in Section 8.5.2, we gradually increase the number of epochs until FONDUE reaches a stable estimation of the latent dimensions. As these models have fewer parameters than the convolutional architecture used in Section 8.5.2, they converge

Table 8.5: Number of latent variables n obtained with FONDUE with random initial numbers of dimensions n_{init} on Celeba. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score. The last line is the average over all initial numbers of dimensions.

n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training	n_{init}
32.4 $\pm$ 1.9	23 min	8	2	5
32.3 $\pm$ 0.8	20 min	7	2	19
32.4 $\pm$ 1.2	17 min	6	2	22
31.6 $\pm$ 1.8	23 min	8	2	39
31.7 $\pm$ 0.8	23 min	8	2	74
31.9 $\pm$ 1.1	26 min	9	2	94
32.2 $\pm$ 0.4	23 min	8	2	116
32.1 $\pm$ 1.1	23 min	8	2	145
32.2 $\pm$ 1.3	26 min	9	2	180
31.9 $\pm$ 1.3	28 min	10	2	182
32.1 $\pm$ 1.2	28 min	8	2	

Table 8.6: Number of latent variables n obtained with FONDUE with random initial numbers of dimensions n_{init} on dSprites. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score. The last line is the average over all initial numbers of dimensions.

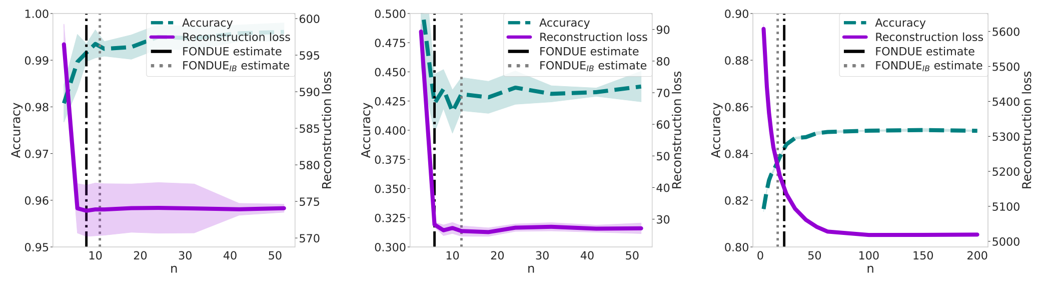
n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training	n_{init}
10.6 $\pm$ 1.1	63 min	6	2	5
10.6 $\pm$ 0.8	63 min	6	2	19
10.7 $\pm$ 0.5	53 min	5	2	22
11.2 $\pm$ 0.6	74 min	7	2	39
11.2 $\pm$ 0.8	74 min	7	2	74
11.1 $\pm$ 1.0	84 min	8	2	94
10.5 $\pm$ 1.1	84 min	8	2	116
11.0 $\pm$ 0.7	84 min	8	2	145
10.9 $\pm$ 0.6	84 min	8	2	180
11.1 $\pm$ 0.9	84 min	8	2	182
10.9 $\pm$ 0.8	75 min	7	2	

Table 8.7: Number of latent variables n obtained with FONDUE and FONDUE$_{IB}$ for **fully-connected VAEs**. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score.

	Dataset	n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training
FONDUE	Symsol	7.6 ± 0.7	9 min	5	3
FONDUE	dSprites	5.2 ± 0.6	16 min	5	1
FONDUE	Celeba	$22.0 \pm 0.$	23 min	6	4
FONDUE$_{IB}$	Symsol	11 ± 0.5	14 min	6	4
FONDUE$_{IB}$	dSprites	12.3 ± 0.5	28 min	5	2
FONDUE$_{IB}$	Celeba	16.4 ± 0.7	5 min	5	1

more slowly and need to be trained for more epochs on Celeba and Symsol before reaching a stable estimation (Arora, Cohen and Hazan, 2018; Sankararaman et al., 2020). As before, FONDUE$_{IB}$ generally requires more epochs than FONDUE to converge, except for Celeba.

Overall, we can see that FONDUE also provides good results on the FC architectures, despite a slower convergence, showing robustness to architectural changes.



(a) Symsol (b) dSprites (c) Celeba

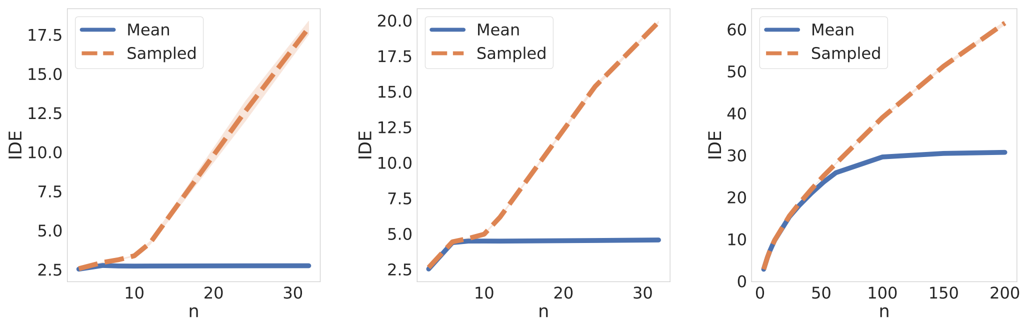
Figure 8.8: Reconstruction loss and accuracy obtained for generation and downstream tasks of **fully-connected VAEs** for test data from Symsol, dSprites, and Celeba with an increasing number of latent variables. The plain and dashed vertical lines indicate the number of dimensions found by FONDUE and FONDUE$_{IB}$.

8.8 FONDUE BASED ON INTRINSIC DIMENSION ESTIMATION

In this section, we are interested in replacing $\mathrm{Tr}(\mathrm{Var}[\mathbf{z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{\mu}])$ by $IDE_{\mathbf{z}} - IDE_{\boldsymbol{\mu}}$ in Algorithm 1. While we will not provide a formal proof of the relationship between the polarised regime and ID estimation, we will show that ID estimation match the constraints discussed in Section 8.3.2 as it is 1) sensitive to the different types of variables in mean and sampled representations 2) stable early during the training. We generally obtain similar results as in Section 8.5, and any discrepancies may be attributed to the choice of t value as, in opposition to Section 8.3.2, we lack a principled way of choosing it.

8.8.1 *IDEs of the mean and sampled representations of VAEs*

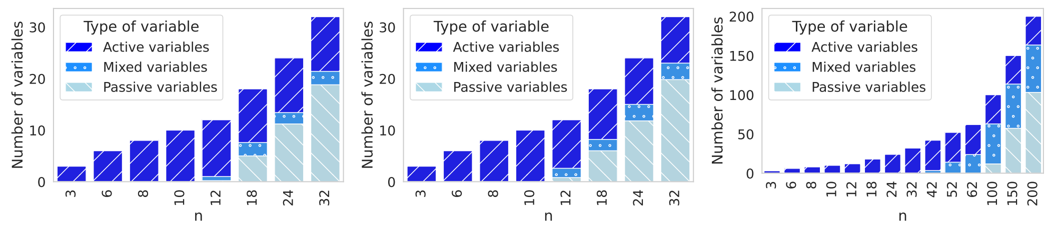
Mean and sampled representations have different IDEs Looking into the IDEs of mean and sampled representations in Figure 8.9, we see a clear pattern emerges: when increasing the number of latent variables, the IDEs remain similar up to a point, then abruptly diverge. As discussed in Section 4.2, once a VAE has enough latent variables to encode the information needed by the decoder, the remaining variables will become passive to minimise the KLD in Equation 3.5. This phenomenon will naturally occur when we increase the number of latent variables. We can thus hypothesise that the difference between the mean and sampled IDEs grows with the number of mixed and passive variables. This is verified by computing the number of active, mixed, and passive variables using the method proposed in Section 6.4.1, as shown in Figure 8.10.



(a) Symsol (b) dSprites (c) Celeba

Figure 8.9: IDE of the mean and sampled representations of VAEs trained with an increasing number of latent dimensions n. (a), (b), and (c) shows the results on Symsol, dSprites, and Celeba, respectively.

The IDEs of the model's representations do not change much after the first epoch The IDEs of the different layers do not change much after the first epoch for well-performing models (see Figure 8.11). However, for Celeba, whose number of latent dimensions is lower than the data IDE and thus cannot reconstruct the data



(a) Symsol (b) dSprites (c) Celeba

Figure 8.10: Quantity of active, mixed, and passive variables of VAEs trained with an increasing number of latent dimensions n. (a), (b), and (c) show the results on Symsol, dSprites, and Celeba.

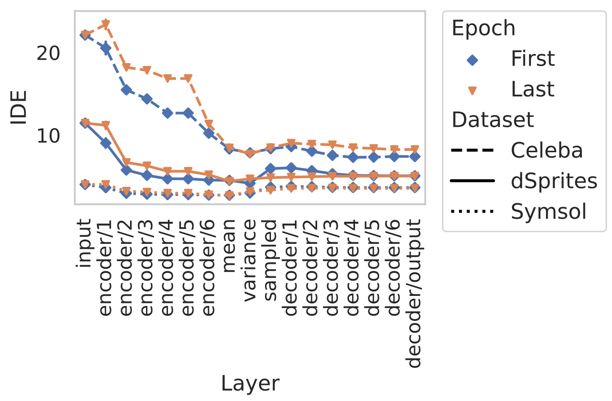


Figure 8.11: The evolution over multiple epochs of the IDE of the representations learned by VAEs using 10 latent variables on Symsol, dSprites, and Celeba.

well, the IDEs tend to change more in the early layers of the encoder, displaying a higher variance.

8.8.2 *FONDUE with IDE*

As discussed above, the IDEs of the mean and sampled representations start to diverge when (unused) passive variables appear, and this is already visible after the first epochs of training. The difference of IDEs between the mean and sampled representations thus meet the two criteria for extension listed in Section 8.3.2 and can be used as a new flavour of FONDUE, FONDUE$_{IDE}$. As for FONDUE$_{IB}$, we simply replace the difference between traces by the difference between IDEs as shown in Algorithms 5 and 6.

How to select a suitable value of t? As we do not have a theoretical relationship between IDE and the polarised regime, it is more complicated to provide a principled way to select the threshold of FONDUE$_{IDE}$. While t was set to a fixed value of 1

Algorithm 5 FONDUE$_{IDE}$

```
1: procedure FONDUE$_{IDE}(t, IDE_{data}, e)$
2:     $l \leftarrow 0$
3:     $u \leftarrow \infty$
4:     $n \leftarrow IDE_{data}$
5:     $m \leftarrow \{\}$
6:     while $n \neq l$ do
7:         $IDE_z, IDE_\mu \leftarrow$ GET-MEM$(m, n, e)$
8:         if $IDE_z - IDE_\mu) \leqslant t$ then
9:             $l \leftarrow n$
10:            $n \leftarrow \min(n \times 2, u)$
11:        else
12:            $u \leftarrow n$
13:            $n \leftarrow$ floor $\left(\frac{l+u}{2}\right)$
14:        end if
15:    end while
16:    return $n$
17: end procedure
```

Algorithm 6 GET-MEM$_{IDE}$

```
1: procedure GET-MEM$(m, n, e)$
2:     if $m[n] = \emptyset$ then
3:         $vae \leftarrow$ TRAIN-VAE$(dim = n, n\_epochs = e)$
4:         $IDE_z, IDE_\mu \leftarrow IDEs(vae)$
5:         $m[n] \leftarrow IDE_z, IDE_\mu$
6:     end if
7:     return $m[n]$
8: end procedure
```

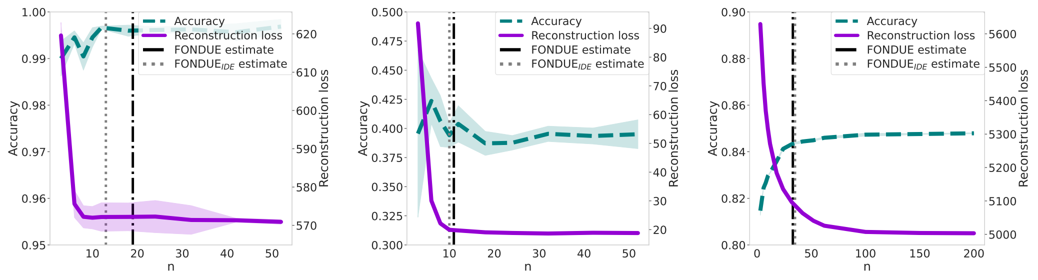
Table 8.8: Number of latent variables n obtained with FONDUE and FONDUE$_{IDE}$. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score.

	Dataset	n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training
FONDUE	Symsol	19.1 ± 0.7	6 min	8	1
FONDUE	dSprites	10.9 ± 0.7	42 min	4	2
FONDUE	Celeba	32.6 ± 0.7	17 min	6	2
FONDUE$_{IDE}$	Symsol	12.6 ± 0.5	10 min	6	2
FONDUE$_{IDE}$	dSprites	10.1 ± 0.7	42 min	4	2
FONDUE$_{IDE}$	Celeba	34.7 ± 1.0	32 min	6	4

as in Section 8.4, one could wonder if this would be a good fit for their particular use case. By looking at Figures 8.9 and 6.1, one can see that the difference between the IDEs of the mean and sampled representations is generally close to the number of additional mixed and passive variables. Thus, t represents this number of "extra variables" (mixed and passive) that we want to allow the model to use, indicating that the threshold obtained for FONDUE is readily applicable to FONDUE$_{IDE}$.

Obtaining stable estimates As in Section 8.5, to ensure stable estimates, we computed FONDUE multiple times, gradually increasing the number of epochs e until the predicted p stopped changing. As reported in Table 8.8, the results were generally stable after two epochs, except for Celeba which needed four.

Analysing the results of FONDUE As shown in Table 8.8, the execution time of FONDUE$_{IDE}$ for finding the number of dimensions for one dataset is much shorter than for fully training one model (approximately 2h using the same GPUs) but generally longer than the original FONDUE algorithm. Moreover, one can see in Figure 8.12 that the number of latent dimensions predicted by FONDUE and FONDUE$_{IDE}$ are very close for dSprites and Celeba. FONDUE$_{IDE}$ also performs better on Symsol with a number of dimensions closer to what would be chosen with the Elbow method. We hypothesise that FONDUE$_{IDE}$ may cope better with more noisy setups where multiple runs of a VAE reach very different reconstruction loss. It could thus be interesting to investigate ID estimation through the lens of the polarised regime to provide a more robust alternative to FONDUE with theoretical guarantees.



(a) Symsol (b) dSprites (c) Celeba

Figure 8.12: Reconstruction loss and accuracy obtained for generation and downstream tasks of **VAEs** for Symsol, dSprites, and Celeba with an increasing number of latent variables. The plain and dashed vertical lines indicate the number of dimensions found by FONDUE and FONDUE$_{IDE}$.

8.9 CONCLUSION

By studying the effect of the polarised regime on the mean and sampled representations, we have shown that one can detect the types of variables (active, passive or mixed) learned by VAEs. These observations lead to FONDUE: an algorithm which can find the number of latent dimensions after which the mean and sampled representations start to strongly diverge. Increasing the number of dimensions beyond this number will result in adding passive or mixed variables which will generally not contribute much to the reconstruction quality and downstream task accuracy. Hence, FONDUE will locate the Elbow point of the performance curves. After proving the correctness of our algorithm, we have shown that it is a faster, automated and unsupervised alternative to existing methods which does not require to fully train any model, is not impacted by architectural changes, and can be used for deterministic AEs.

While FONDUE has been demonstrated to be an efficient algorithm, it could be improved and extended in several ways: 1) we have shown that the dimensions given by FONDUE could also be used for deterministic AEs, but it would be interesting to see if this applies to a larger range of unsupervised models (e.g., Generative Adversarial Networks (GANs), clustering methods, etc.); 2) FONDUE can be extended in a number of ways by replacing the difference of Trace in Algorithm 2 by any function that reliably provides different results in mean and sampled representations early in the training process, as illustrated with IDEs in Section 8.8. These extensions could be beneficial both in terms of execution time (if the function is faster or convergence is reached earlier) and complementary theoretical insights (if the function is also theoretically grounded). 3) One could also extend FONDUE to find a good hyperparameter value for β-VAE by replacing n by β in Algorithms 1 and 2.

Part III

CONCLUSION AND FUTURE WORK

9 CONCLUSION

In this thesis we have shown that the polarised regime is a signature behaviour of "classical" VAEs which arises from the ELBO optimisation. While the analysis of this phenomenon has been used to explain posterior collapse and to draw links with PCA, it has been restricted to one data example and a standard Gaussian prior. After extending these results to other priors and multiple data examples, we have shown that the polarised regime can be used as a tool to analyse and improve the latent representations learned by VAEs.

The polarised regime with other priors One could wonder if VAEs with different priors would also learn in a polarised regime. Such knowledge is important as this mechanism is responsible for the sparsity of the latent representation and other priors could introduce an unsuspected (and undesired) redundancy of the latent space. This is especially important in the context of disentangled representations, if the end users assume that there is no correlation between the latents and select their downstream model accordingly. In Chapter 5, we show that iVAEs, whose prior is Gaussian but where the mean and variance are learned still behave in a polarised regime. Thus, such models, which are guaranteed to provide disentangled representations, still benefit from a sparse latent space induced by the polarised regime. This also revealed that such study (even in the simple linear case) can highlight specific behaviours of the models which differs from what is observed in "classical" VAEs. For example, the prior variance of the active variables of iVAEs are often very large because they are based on the squared mean value which themselves can be high when the reconstruction becomes very precise. While this study is restricted to VAEs with Gaussian prior, we believe that other VAEs with non-Gaussian priors should be subject to a similar analysis. Indeed, some results may be wrongly imputed to specific hyperparameters while, in fact, this stems from the absence of polarised regime.

The polarised regime with multiple data examples Most of the metrics used to evaluate the quality of the latent representations (e.g., disentanglement metrics, representational similarity metrics) are based on multiple data examples. Thus, one cannot directly study the impact of the polarised regime on these metrics using the original results from Dai et al. (2017); Rolinek, Zietlow and Martius (2019) as they are restricted to one data example. Therefore, in Chapter 6 we further extend the polarised regime to multiple data examples and show that active variables are similar in the mean and sampled representations while passive variables differ. We also observe the emergence of “mixed variables”: variables which are active for only a subset of data examples. Based on this extension, we propose a simple rule to distinguish active, mixed and passive variables in the latent space.

Explaining unexpected behaviours of VAEs Based on the extension of the polarised regime to multiple data examples, in Chapter 6, we study the intriguing result reported by Locatello et al. (2019a). Specifically, Locatello et al. (2019a) observed empirically that the mean representation was less disentangled than its sampled counterpart. Given that the mean (and not the sampled) representations are used as inputs for downstream tasks, this challenged the assumption that disentangled sampled representations guaranteed disentangled embeddings on downstream tasks. However, the extension of the polarised regime to multiple data examples show that active variables are similar in the mean and sampled representation. Thus, the discrepancy between mean and sampled representations can only come from the passive (and mixed) variables. We empirically demonstrate that passive variables are indeed the source of the discrepancy as representations truncated of their passive variables display similar disentanglement scores. Furthermore, a fine-grained analysis of the passive variables of the mean representations reveal systematic correlation between passive and active variables which are then erased in the sampled representations (because the passive variables values are replaced by random samples from the standard Gaussian). We thus conclude that mean representations can safely be used for downstream tasks where disentanglement matters as long as they are truncated of their passive variables. This truncation should also be considered whenever the downstream model is sensitive to correlated values as passive variables can reach very high correlation score with some active variables.

Consistency of the polarised regime In Chapter 7 we study the representations learned by VAEs with different initialisations and learning objectives. We observe that the angle between the latent representations of different data samples is similar for different initialisations and learning objectives. Using the invariance properties of

CKA, we demonstrate that VAEs with high representational similarity are guaranteed to have the same number of active and passive variables. Specifically, any models with high representational similarity will have the same active variables up to rotation and scaling. Moreover, we empirically observe that the number of active and passive variables is determined very early in the training.

Improving VAEs with the polarised regime Using the extension of the polarised regime to multiple data examples and its early convergence property, in Chapter 9, we show that it can also be used as a tool to improve VAEs. Given the properties of active and passive variable, we demonstrate that the difference between the trace of the covariance of the mean and sampled representations can be used as a proxy to monitor the appearance of passive variables. Taking advantage of the early convergence of the number of active and passive variables, we then design FONDUE, an algorithm looking for the maximum number of latent dimensions after which passive variables start to appear. We demonstrate that this algorithm gives similar results as the Elbow method without human supervision and do not require to fully train any models. FONDUE thus improves existing algorithms to determine the dimensionality of the latent representations which require some level of human supervision and to fully train at least one model. We also show that FONDUE can be extended in a number of ways by replacing the trace difference by any another metric which can indicate the appearance of passive variables early in the training. We further observed that the number of dimensions obtained for VAEs could also be used for deterministic AEs with similar results.

10

FUTURE WORK

As discussed in Chapter 9, we have shown that the polarised regime and its extensions could be used to understand and improve the representations learned by VAEs. However, our results are only the first steps into this understudied domain, and our findings can be extended in a number of ways.

Which priors lead to a polarised regime? We have seen in Chapter 5 that any VAEs with Gaussian prior and learned mean and variance learn in a polarised regime. However, many extensions of the original VAEs model propose to use different prior distribution adapted to their specific tasks. For example, Mercatali and Freitas (2021) proposed to use a Gumbel-Softmax prior distribution for Natural Language Processing (NLP) models, Mathieu et al. (2019) proposed multiple prior distributions to enforce disentangled representation learning, and Floto, Kremer and Nica (2023) demonstrated that a titled Gaussian prior was beneficial for Out Of Distribution (OOD) detection. However, we do not know whether these models learn in a polarised regime, which leads to a number of questions. Do these models still benefit from the sparsity inducing behaviour of "classical" VAEs? Does one need to prune passive variables from the latent representations before using them on downstream tasks? Crucially, can this impact the analysis of the latent space and the resulting interpretations[1]? Thus, determining which priors induce (or not) a polarised regime is an important topic to pursue.

Which models learn in a polarised regime? In this thesis we restricted our study to a sub-family of VAEs aiming to provide disentangled representations. However, one could wonder whether these results can be extended to other types of VAEs such as hierarchical VAEs (Ranganath, Tran and Blei, 2016; Sø nderby et al., 2016; Klushyn et al., 2019; Chien and Wang, 2019; Vahdat and Kautz, 2020; Li et al., 2020),

1 An example of such a case is the unexplained discrepancies between the disentanglement scores of mean and sampled representations observed by Locatello et al. (2019a)

or f-divergence VAEs (Wan, Li and Hovakimyan, 2020). Furthermore, does this phenomenon extend to different types of generative models such as Variational Gradient Origin Networks (VGONs) (Bond-Taylor and Willcocks, 2021) or GANs (Goodfellow et al., 2014; Chu, Blanchet and Glynn, 2019)? Clarifying this could lead to improved understanding of the representations learned by these models.

Creating new tools based on the polarised regime Variational inference in deep learning is not always used to directly optimise the model, as in VAEs. For example, Variational dropout is used to sparsify the activations of various layers (Kingma, Salimans and Welling, 2015). If the polarised regime can also be applied in this context, one could benefit from this thesis' research and, for example, extend FONDUE to automatically determine the size of any layers regularised with variational dropout.

Part IV

APPENDIX

A

APPENDIX OF CHAPTER 5

A.1 EXPERIMENTAL SETUP

To facilitate the reproducibility of our experiment, the source code is available at https://github.com/bonheml/VAE_learning_dynamics, the hyper-parameters and architecture used are described in Tables A.1 and A.2.

Table A.1: Model hyperparameters

Parameter	Value
Batch size	64
Latent space dimension	30
Optimizer	Adam
Adam: β_1	0.9
Adam: β_2	0.999
Adam: ϵ	1e-8
Adam: learning rate	0.0001
Reconstruction loss	Bernoulli
Training steps	300,000
Train/test split	90/10

Table A.2: Model architecture

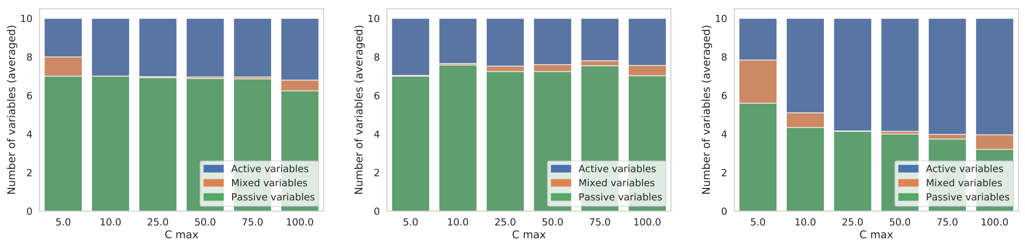
Encoder
Input: $(\mathbb{R}^{64\times 64\times channels}, \mathbb{R}^5)$
Conv, kernel=4×4, filters=32, activation=ReLU, strides=2
Conv, kernel=4×4, filters=32, activation=ReLU, strides=2
Conv, kernel=4×4, filters=64, activation=ReLU, strides=2
Conv, kernel=4×4, filters=64, activation=ReLU, strides=2
FC, output shape=261, activation=ReLU
FC, output shape=2x30
Decoder
Input: $\mathbb{R}^{30}$
FC, output shape=256, activation=ReLU
Deconv, kernel=4×4, filters=64, activation=ReLU, strides=2
Deconv, kernel=4×4, filters=32, activation=ReLU, strides=2
Deconv, kernel=4×4, filters=32, activation=ReLU, strides=2
Deconv, kernel=4×4, filters=channels, activation=ReLU, strides=2
Prior variance
Input: $\mathbb{R}^5$
FC, output shape=50, activation=Leaky ReLU
FC, output shape=50, activation=Leaky ReLU
FC, output shape=50, activation=Leaky ReLU
FC, output shape=30

B

APPENDIX OF CHAPTER 6

B.1 THE CURIOUS CASE OF ANNEALED VAE

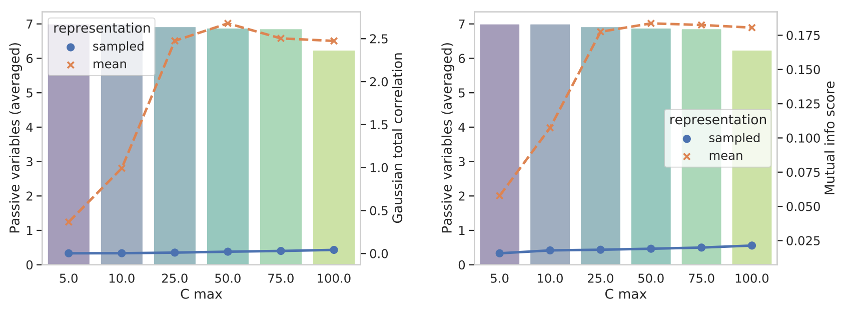
In opposition to the other models that we studied in Section 7.4, annealed VAE surprisingly exhibits a high number of passive variables regardless of the regularisation strength on most datasets, which can be seen in Figure B.1. Note that in contrast to the remaining architectures that we study, a higher value of the hyperparameter C means that the regularisation strength decreases, whereas higher β in β-VAEs implies stronger regularisation. As Burgess et al. (2018) originally argued that a higher channel capacity, C, should help the model to learn more latent factors as the training progresses, one would assume that the number of active variables should increase with a higher value of C, but it is generally not the case in Figure B.1.



(a) DSprites (b) Scream dSprites (c) Cars3D

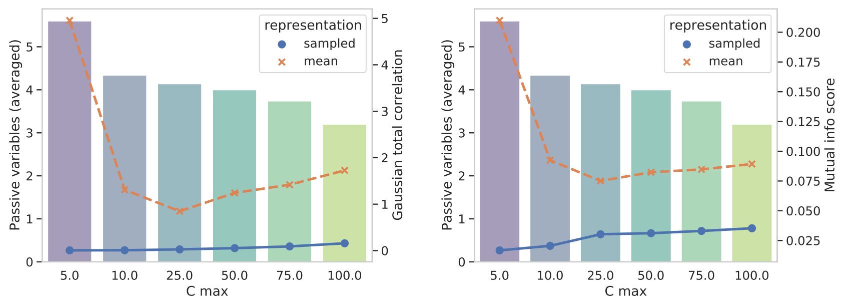
Figure B.1: Number of passive, mixed and active variables of annealed VAE trained on dSprites, scream dSprites and cars3D with decreased regularisation strength. The results are averaged over 50 runs for each regularisation value.

Given the near constant number of passive variables observed across all the bars in Figure B.1, one could expect similarly constant TC and averaged MI scores. However, we can see in Figures B.2 and B.3 that the TC and averaged MI generally decrease with higher regularisation strengths. Moreover, Figure B.5 shows that removing the passive variables effectively reduces the overall TC and averaged MI scores, suggesting that a higher channel capacity, C, (i.e., a lower regularisation strength at the end of training) may encourage the passive variables of mean representations to be more correlated. This is further confirmed in Figure B.4, where we can see that the effective rank obtained with a higher channel capacity and less passive variables is close to the one obtained with a lower channel capacity and more passive variables. For example, in cars3D, the effective rank of the mean representation for a channel capacity of 25 and 4 passive variables is the same as the one obtained for a channel capacity of 100 and 3 passive variables. In conclusion, despite their near-constant number of passive variables, annealed VAE's results are consistent with our other findings: the passive variables of the mean representations are still responsible for the higher TC and averaged MI scores, but their correlation seems to increase with the channel capacity, C.



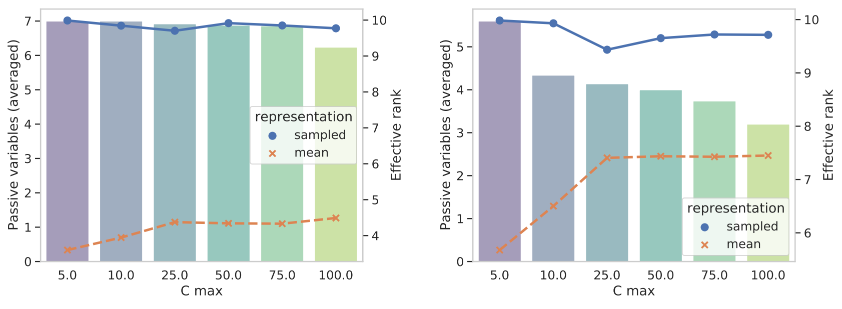
(a) (b)

Figure B.2: Comparison of the total correlation and averaged mutual information with the number of passive variables of mean and sampled representations of annealed VAE trained on dSprites. Figure (a) is the total correlation and Figure (b) the averaged mutual information. The lines indicate the metric scores of the two representations, and the bars the average number of passive variables.



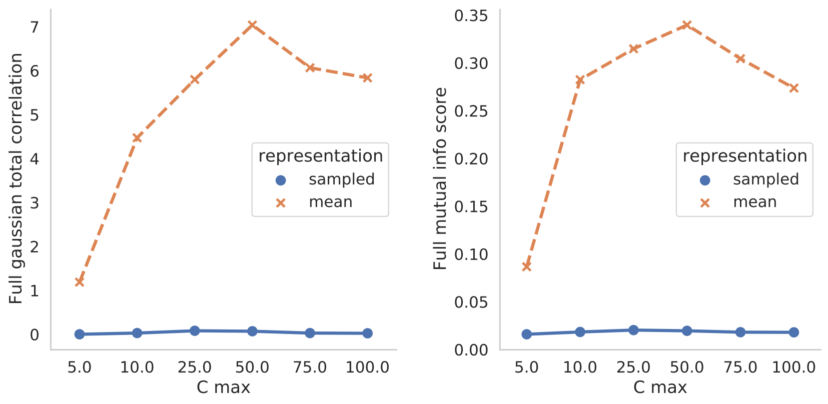
(a) (b)

Figure B.3: Comparison of the total correlation and averaged mutual information with the number of passive variables of mean and sampled representations of annealed VAE trained on Cars3D. Figure (a) is the total correlation and Figure (b) the averaged mutual information. The lines indicate the metric scores of the two representations, and the bars the average number of passive variables.

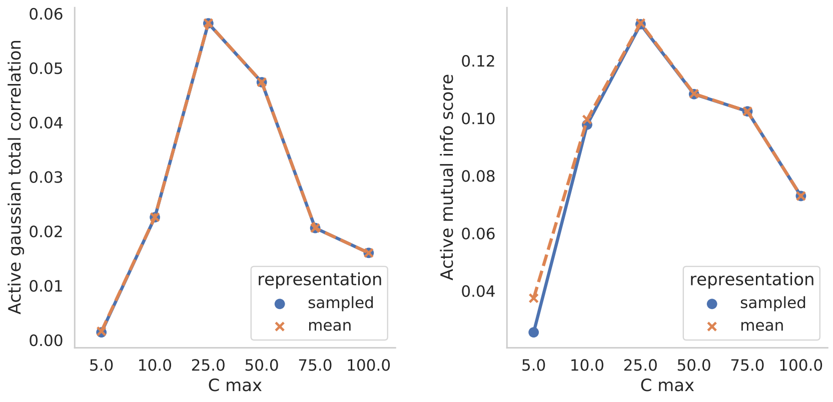


(a) (b)

Figure B.4: Comparison of the effective rank with the number of passive variables of mean and sampled representations of annealed VAE trained on dSprites and Cars3D. The lines indicate the metric scores of the two representations, and the bars the average number of passive variables.



(a) (b)



(c) (d)

Figure B.5: Comparison of the total correlation and averaged mutual information scores of the mean representation of annealed VAE trained on scream dSprites. Figures (a) and (b) are the results of the total correlation and averaged mutual information score using the full representation, and figures (c) and (d) are the results using active variables only.

C

APPENDIX OF CHAPTER 7

C.1 EXPERIMENTAL SETUP

To facilitate the reproducibility of our experiment, we detail below the configuration used for model training.

VAE training

Our implementation uses the same hyperparameters as Locatello et al. (2019a), and the details are listed in Tables C.1 and C.2. We reimplemented Locatello et al. (2019a) code base, designed for Tensorflow 1, in Tensorflow 2 using Keras. The model architecture used is also identical, as described in Table C.3. Each model is trained 5 times, on seeded runs with seed values from 0 to 4. Intermediate models are saved every 1,000 steps for SmallNorb, 6,000 steps for Cars3D and 11,520 steps for dSprites. Every image input is normalised to have pixel values between 0 and 1.

For the fully-connected models presented in Appendix 8.7, we used the same architecture and hyperparameters as those implemented in `disentanglement lib` of Locatello et al. (2019a), and the details are presented in Tables C.4 and C.5.

C.2 RESOURCES

As mentioned in Sections 7.1 and 7.3, we released the code of our experiment, the pre-trained models and similarity scores of Chapter 7:

- The similarity scores can be downloaded at `https://data.kent.ac.uk/444/`
- The pre-trained models can be downloaded at `https://data.kent.ac.uk/428/`

Table C.1: Shared hyperparameters

Parameter	Value
Batch size	64
Latent space dimension	10
Optimizer	Adam
Adam: β_1	0.9
Adam: β_2	0.999
Adam: ϵ	1e-8
Adam: learning rate	0.0001
Reconstruction loss	Bernoulli
Training steps	300,000
Intermediate model saving	every 6K steps
Train/test split	90/10

Table C.2: Model-specific hyperparameters

Model	Parameter	Value
β-VAE	β	[1, 2, 4, 6, 8]
β-TC VAE	β	[1, 2, 4, 6, 8]
DIP-VAE II	λ_{od}	[1, 2, 5, 10, 20]
	λ_d	λ_{od}
Annealed VAE	C_{max}	[5, 10, 25, 50, 75]
	γ	1,000
	iteration threshold	100,000

- The source code is available at https://github.com/bonheml/VAE_learning_dynamics

Note that the 300 VAE models released correspond to models trained with:

- 4 different learning objectives,
- 5 initialisations,
- 3 datasets,
- 5 regularisation strengths.

Table C.3: Shared architecture

Encoder	Decoder
Input: $\mathbb{R}^{64\times63\times channels}$	$\mathbb{R}^{10}$
Conv, kernel=4×4, filters=32, activation=ReLU, strides=2	FC, output shape=256, activation=ReLU
Conv, kernel=4×4, filters=32, activation=ReLU, strides=2	FC, output shape=4x4x64, activation=ReLU
Conv, kernel=4×4, filters=64, activation=ReLU, strides=2	Deconv, kernel=4×4, filters=64, activation=ReLU, strides=2
Conv, kernel=4×4, filters=64, activation=ReLU, strides=2	Deconv, kernel=4×4, filters=32, activation=ReLU, strides=2
FC, output shape=256, activation=ReLU, strides=2	Deconv, kernel=4×4, filters=32, activation=ReLU, strides=2
FC, output shape=2x10	Deconv, kernel=4×4, filters=channels, activation=ReLU, strides=2

Table C.4: Fully-connected architecture

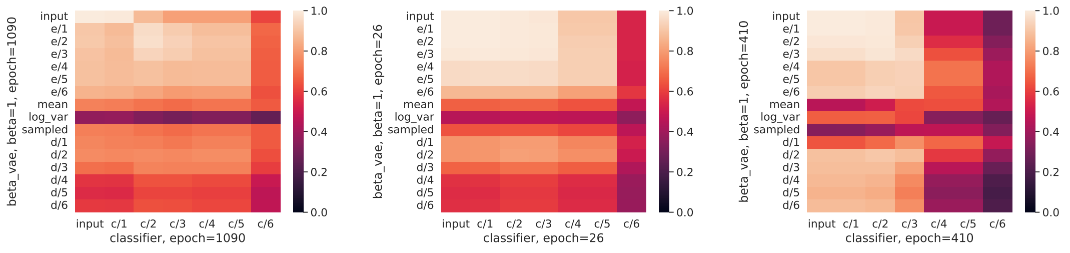
Encoder	Decoder
Input: $\mathbb{R}^{64\times63\times channels}$	$\mathbb{R}^{10}$
FC, output shape=1200, activation=ReLU	FC, output shape=256, activation=tanh
FC, output shape=1200, activation=ReLU	FC, output shape=1200, activation=tanh
FC, output shape=2x10	FC, output shape=1200, activation=tanh

Table C.5: Hyperparameters of fully-connected models

Model	Parameter	Value
β-VAE	β	[1, 8, 16]
β-TC VAE	β	[2]
DIP-VAE II	λ_{od}	[1, 20, 50]
	λ_d	λ_{od}
Annealed VAE	C_{max}	[5]
	γ	1,000
	iteration threshold	100,000

C.3 HOW SIMILAR ARE THE REPRESENTATIONS LEARNED BY ENCODERS AND CLASSIFIERS?

To compare VAEs with classifiers, we used the convolutional architecture of an encoder for classification, replacing the mean and variance layers by the final classifier layers. As shown in Figure C.1, we obtain a high representational similarity when comparing VAEs and classifiers indicating, consistently with the observations of Yosinski et al. (2015), that classifiers seem to learn generative features. This explains why encoders based on pre-trained classifier architectures such as VGG have empirically demonstrated good performances (Liu, Siu and Wang, 2021) and also suggests that the weights of the pre-trained architecture could be used as-is without further updates.



(a) Trained on Cars3D (b) Trained on dSprites (c) Trained on SmallNorb

Figure C.1: (a) shows the CKA similarity scores of activations of a classifier and a β-VAE trained on Cars3D with $\beta = 1$. (b) and (c) show the CKA similarity scores of the same learning objectives and regularisation strengths but trained on dSprites and SmallNorb. All the results are averaged over 5 seeds. We can see that the representational similarity between the layers of the classifiers and of the encoder (top-left quadrant) except mean and variance is very high (CKA stays close to 1). However, the mean, variance, and sampled representations (central diagonal values) are different from the representations learned by the classifier.

C.4 REPRESENTATIONAL SIMILARITY OF VAES AT DIFFERENT EPOCHS

The results obtained in Section 7.4.1 have shown a high similarity between the encoders at an early stage of training and fully trained. One can wonder whether these results are influenced by the choice of epochs used in Figure 7.2. After explaining our epoch selection process, we show below that it does not influence our results, which are consistent over snapshots taken at different stages of training.

Epoch selection

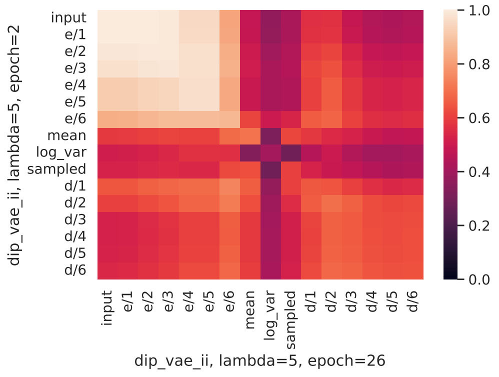
For dSprites, we took snapshots of the models at each epoch, but for Cars3D and SmallNorb, which both train for a higher number of epochs, it was not feasible computationally to calculate the CKA between every epoch. We thus saved models trained on SmallNorb every 10 epochs, and models trained on Cars3D every 25 epochs. Consequently, the epochs chosen to represent the early training stage in Section 7.4.1 is always the first snapshot taken for each model. Below, we preform additional experiments with a broader range of epoch numbers to show that the results are consistent with our findings in Chapter 6, and they do not depend on specific epochs.

Similarity changes over multiple epochs

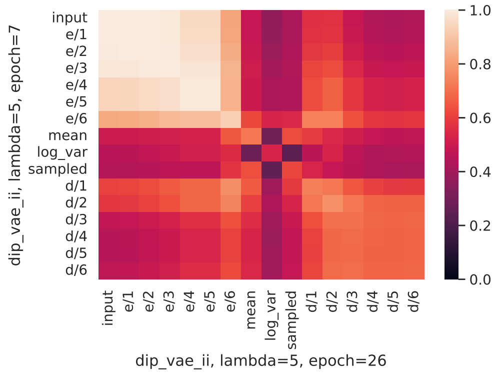
In Figures C.2, C.3, and C.4, we can observe the same trend of learning phases as in Section 7.4.1. First, the encoder is learned, as shown by the high representational similarity of the upper-left quadrant of Figures C.2a, C.3a, and C.4a. Then, the decoder is learned, as shown by the increased representational similarity of the bottom-right quadrant of Figures C.2b, C.3b, and C.4b. Finally, further small refinements of the encoder and decoder representations take place in the remaining training time, as shown by the slight increase of representational similarity in Figures C.2c, C.3c, and C.4c, and Figures C.2d, C.3d, and C.4d.

C.5 CONVERGENCE RATE OF DIFFERENT VAES

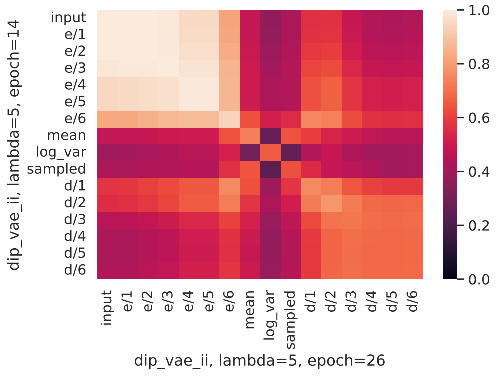
We can see in Figure C.5 that all the models converge at the same epoch, with less regularised models reaching lower losses. While annealed VAEs start converging together with the other models, they then take longer to plateau, due to the annealing process. We can see them distinctly in the upper part of Figure C.5. Overall, the epochs at which the models start to converge are consistent with our choice of epoch for early training in Section 7.4.1.



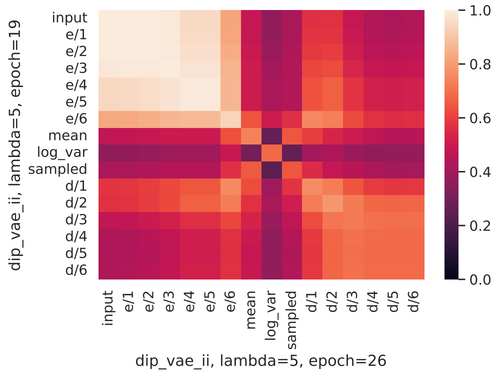
(a) Epoch 2



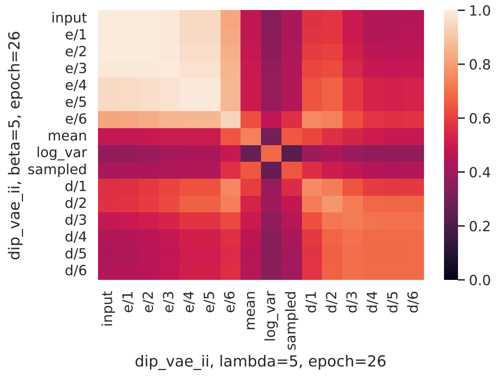
(b) Epoch 7



(c) Epoch 14

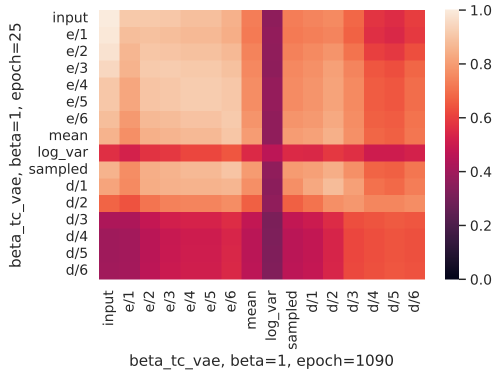


(d) Epoch 19

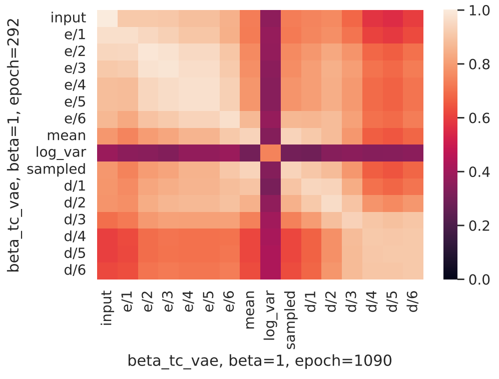


(e) Epoch 26

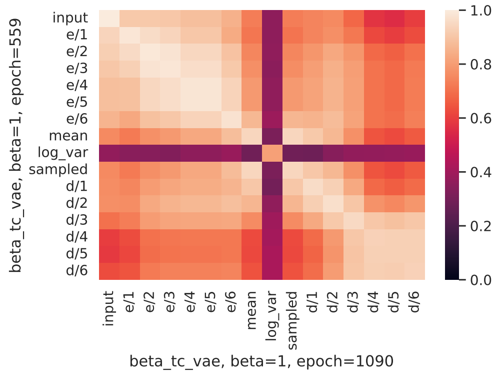
Figure C.2: (a), (b), (c), (d), and (e) show the representational similarity between DIP-VAE II after full training, and at epochs 2, 7, 14, 19, and 26, respectively. All models are trained on dSprites and the results are averaged over 5 runs.



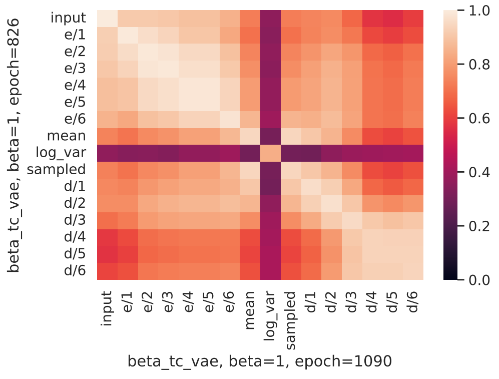
(a) Epoch 25



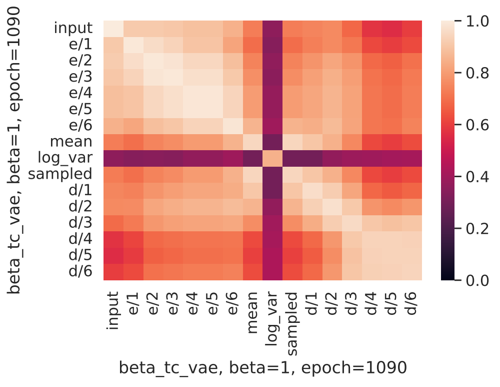
(b) Epoch 292



(c) Epoch 559

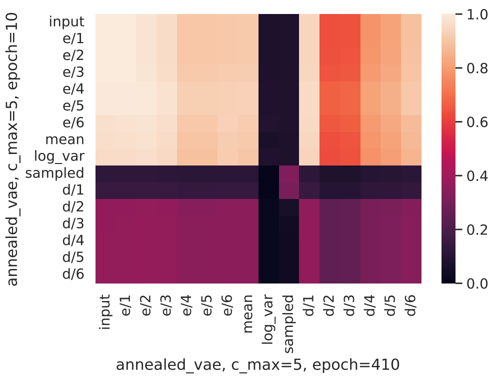


(d) Epoch 826

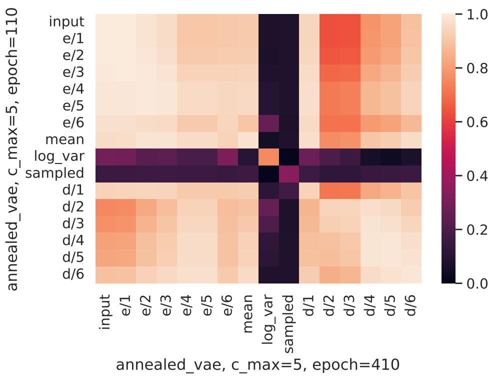


(e) Epoch 1090

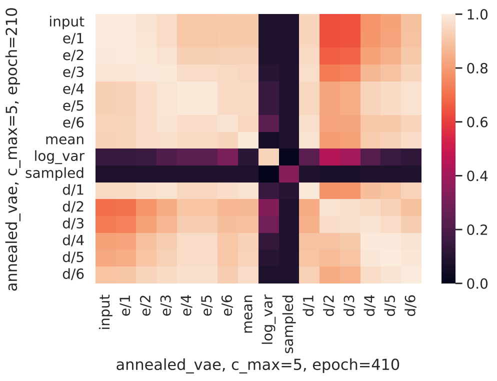
Figure C.3: (a), (b), (c), (d) and (e) show the representational similarity between β-TC VAE after full training, and at epochs 25, 292, 559, 826, and 1090 respectively. All models are trained on Cars3D and the results are averaged over 5 runs.



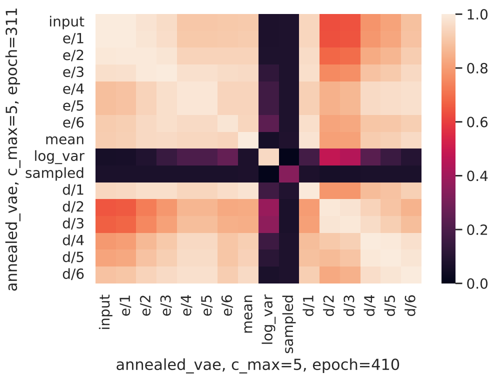
(a) Epoch 10



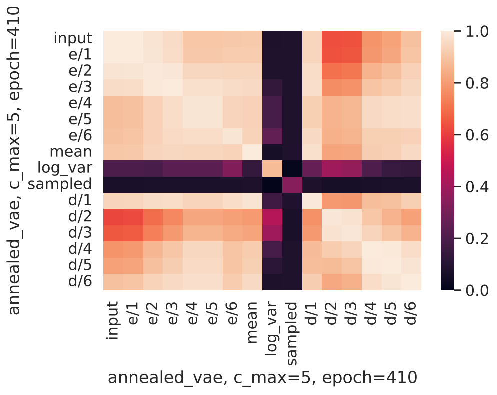
(b) Epoch 110



(c) Epoch 210

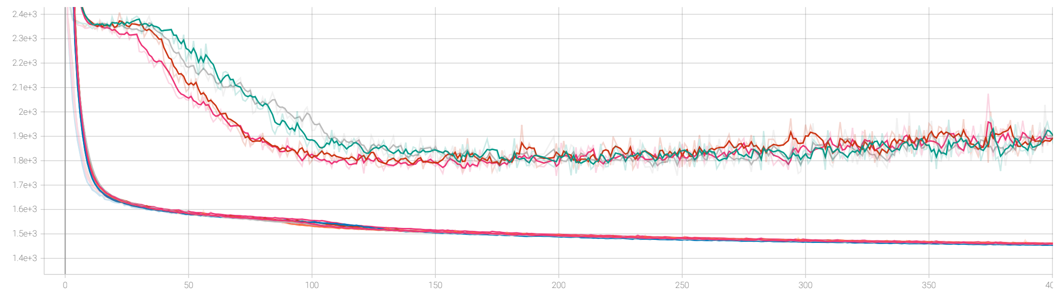


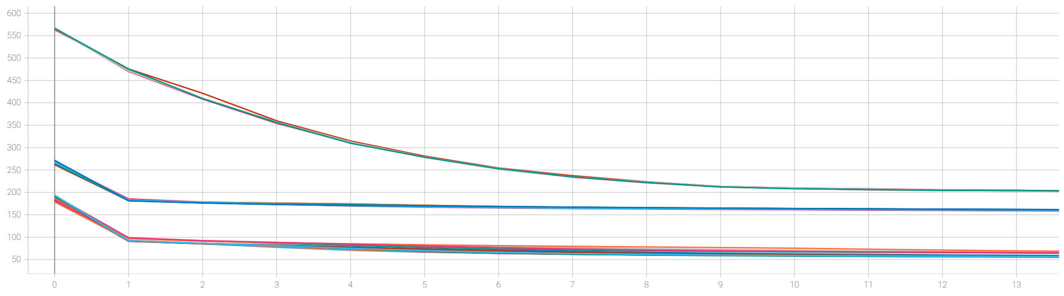
(d) Epoch 311



(e) Epoch 311

Figure C.4: (a), (b), (c), (d) and (e) show the representational similarity between Annealed VAE after full training, and at epochs 10, 110, 210, 311, and 410 respectively. All models are trained on SmallNorb and the results are averaged over 5 runs.


(a) Convergence on Cars3D


(b) Convergence on dSprites

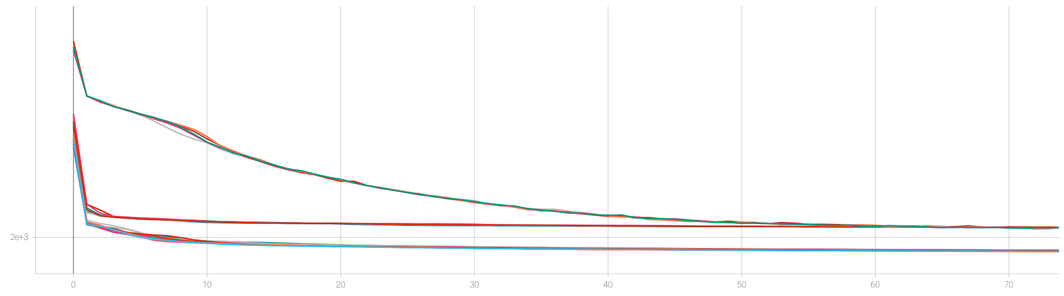

(c) Convergence on SmallNorb

Figure C.5: In (a), (b), and (c), we show the model loss of each model that converged when trained on Cars3D, dSprites, and SmallNorb, respectively. For each learning objective, we display 5 runs of the least and most regularised versions.

D

APPENDIX OF CHAPTER 8

D.1 EXPERIMENTAL SETUP

Our implementation uses the same hyperparameters as Locatello et al. (2019a), as listed in Table D.1. We reimplemented the Locatello et al. (2019a) code base, designed for Tensorflow 1, in Tensorflow 2 using Keras. The model architectures used are also similar, as described in Tables D.2 and D.3. We used the convolutional architecture in Chapter 8 and the fully-connected architecture in Section 8.7. Each model is trained 5 times with seed values from 0 to 4. Every image input is normalised to have pixel values between 0 and 1. TwoNN is used with an anchor of 0.9, and the hyperparameters for MLE can be found in Table D.4.

Table D.1: VAEs hyperparameters

Parameter	Value
Batch size	64
Latent space dimension	3, 6, 8, 10, 12, 18, 24, 32.
	For Celeba only: 42, 52, 62, 100, 150, 200
Optimizer	Adam
Adam: β_1	0.9
Adam: β_2	0.999
Adam: ϵ	1e-8
Adam: learning rate	0.0001
Reconstruction loss	Bernoulli
Training steps	300,000
Train/test split	90/10
β	1

Table D.2: Architecture

Encoder	Decoder
Input: $\mathbb{R}^{64\times63\times channels}$	$\mathbb{R}^{10}$
Conv, kernel=4×4, filters=32, activation=ReLU, strides=2	FC, output shape=256, activation=ReLU
Conv, kernel=4×4, filters=32, activation=ReLU, strides=2	FC, output shape=4x4x64, activation=ReLU
Conv, kernel=4×4, filters=64, activation=ReLU, strides=2	Deconv, kernel=4×4, filters=64, activation=ReLU, strides=2
Conv, kernel=4×4, filters=64, activation=ReLU, strides=2	Deconv, kernel=4×4, filters=32, activation=ReLU, strides=2
FC, output shape=256, activation=ReLU, strides=2	Deconv, kernel=4×4, filters=32, activation=ReLU, strides=2
FC, output shape=2x10	Deconv, kernel=4×4, filters=channels, activation=ReLU, strides=2

Table D.3: Fully-connected architecture

Encoder	Decoder
Input: $\mathbb{R}^{64\times63\times channels}$	$\mathbb{R}^{10}$
FC, output shape=1200, activation=ReLU	FC, output shape=256, activation=tanh
FC, output shape=1200, activation=ReLU	FC, output shape=1200, activation=tanh
FC, output shape=2x10	FC, output shape=1200, activation=tanh

Table D.4: MLE hyperparameters

Parameter	Value
k	3, 5, 10, 20
anchor	0.8
seed	0
runs	5

D.2 ADDITIONAL RELATED WORK

Elbow method based on the Structure Preservation Index In the context of deterministic AEs trained on textual data, Gupta, Banchs and Rosso (2016) proposed to apply the Elbow method to the Structure Preservation Index (SPI) instead of the reconstruction loss. The idea of SPI is to capture structural distortions between the input documents and their reconstruction. It is defined as follows:

$$SPI = \frac{1}{h} \sum_{i,j} \| \boldsymbol{D}_{ij} - \hat{\boldsymbol{D}}_{ij} \|, \tag{D.1}$$

where $\boldsymbol{D}_{ij}$ is the cosine similarity between the documents $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$, and $\hat{\boldsymbol{D}}_{ij}$ is calculated the same way with the reconstructed documents $\hat{\mathbf{x}}^{(i)}$ and $\hat{\mathbf{x}}^{(j)}$.

Human supervision based on the information plane Yu and Príncipe (2019) studied the information of the input preserved by the bottleneck layer $\mathbf{z}$ of stacked AEs (Vincent et al., 2010) using the information plane spanned by $\mathrm{I}(\mathbf{x}, \mathbf{z})$ and $\mathrm{I}(\hat{\mathbf{x}}, \mathbf{z})$, where $\mathrm{I}(\cdot, \cdot)$ denotes mutual information. In order to approximate the true entropy, they used a kernel estimator of the Rényi's α-order entropy. Specifically, given the samples $\boldsymbol{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{h}$ of a random variable $\mathbf{x}$, a positive definite kernel κ, the resulting Gram matrix $\boldsymbol{K}$ where $\boldsymbol{K}_{i,j} \triangleq \kappa(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$, and its normalised version $\boldsymbol{A}_{i,j} \triangleq \frac{\boldsymbol{K}_{i,j}}{h\sqrt{\boldsymbol{K}_{i,i}\boldsymbol{K}_{j,j}}}$, the entropy estimator is

$$S_\alpha(\boldsymbol{A}) \triangleq \frac{1}{1-\alpha} \log_2 \left(\sum_{i=1}^{h} \lambda_i(\boldsymbol{A})^\alpha \right), \tag{D.2}$$

where $\lambda_i(A)$ denotes the ith eigenvalue of $\boldsymbol{A}$.
Similarly, the joint entropy is estimated by

$$S_\alpha(\boldsymbol{A}, \boldsymbol{B}) \triangleq S_\alpha \left(\frac{\boldsymbol{A} \odot \boldsymbol{B}}{\mathrm{Tr}(\boldsymbol{A} \odot \boldsymbol{B})} \right), \tag{D.3}$$

where $\odot$ denotes the Hadamard product,‘ and A and B are normalised Gram matrices as before. From Equations D.2 and D.3, one can thus obtain the MI,

$$\mathrm{I}_\alpha(\boldsymbol{A}, \boldsymbol{B}) \triangleq S_\alpha(\boldsymbol{A}) + S_\alpha(\boldsymbol{B}) - S_\alpha(\boldsymbol{A}, \boldsymbol{B}). \tag{D.4}$$

In their experiment, Yu and Príncipe (2019) set $\alpha = 1.01$ and chose a RBF kernel, such that, given $\boldsymbol{X} \in \mathbb{R}^{h \times m}$

$$\kappa(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp \left(- \frac{\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|^2}{2s^2} \right), \tag{D.5}$$

where $\|\cdot\|^2$ is the squared Euclidean distance. s is estimated based on Silverman's rule of thumb for Gaussian density estimation (Silverman, 1998), such that $s \triangleq \sigma(\boldsymbol{X}) h^{-1/(4+m)}$ where σ denotes the standard deviation. The authors observed that for a large enough n, the information plane started to display curved patterns and concluded that a good number of latent dimensions n corresponded to the information plane just before the appearance of this change of pattern, which can be seen as an application of the Elbow method. The proposed technique requires to fully train multiple models and visually inspect the information planes obtained for different n.

D.3 COMPARISON IN THE SUPERVISED SETTING

In this section, we compare the results obtained by the IB algorithm with the results obtained with $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$ in the supervised setting. To avoid unfair comparison due to IB fully training one or more models depending on the chosen range, we restrict both implementations to the number of epochs after which they provide stable estimate for each dataset, as per Table 8.1. The implementations of the IB algorithm and the supervised version of $\mathrm{Tr}(\mathrm{Var}[\boldsymbol{Z}]) - \mathrm{Tr}(\mathrm{Var}[\boldsymbol{M}])$, BS_{FONDUE}, are shown in Algorithms 7 and 8. We further define $l = 1$ and $u = 200$ for all the runs. Note that the GET-MEM functions are the same as Algorithms 2 and 4 and the thresholds t remain unchanged. For IB, Algorithm 7 is thus equivalent to the original implementation of (Boquet et al., 2021) except that any model training is fixed to a given number of epochs. We can see in Table D.5 that for the chosen range, IB and BS_{FONDUE} take longer to compute than FONDUE and FONDUE_{IB} for all datasets except Symsol where the execution time is similar. Indeed, the number of models to partially train is higher with binary search than in the unsupervised setting for dSprites and Celeba but similar for Symsol. The predicted number of dimensions are consistent with Table 8.1, BS_{FONDUE} being closer to the results obtained with Elbow methods than IB, as before.

Algorithm 7 IB

```
1: procedure IB($t, l, u, e$)
2:     $m \leftarrow \{\}$
3:     while $l < u$ do
4:         $n \leftarrow \mathrm{floor}\left(\frac{l+u}{2}\right)$
5:         $\mathrm{H}(\mathbf{z}),\ \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \leftarrow \text{GET-MEM}_{\mathrm{IB}}(m, n, e)$
6:         if $\mathrm{H}(\mathbf{z}) - \mathrm{I}(\mathbf{x}, \hat{\mathbf{x}}) \leqslant t$ then
7:             $l \leftarrow n + 1$
8:         else
9:             $u \leftarrow n$
10:        end if
11:    end while
12:    return $l - 1$
13: end procedure
```

Table D.5: Number of latent variables n obtained with BS$_{FONDUE}$ and IB. The results are averaged over 10 seeds, and computation times are reported for NVIDIA A100 GPUs. The computation time is given for one run of the algorithm over the minimum number of epochs needed to obtain a stable score.

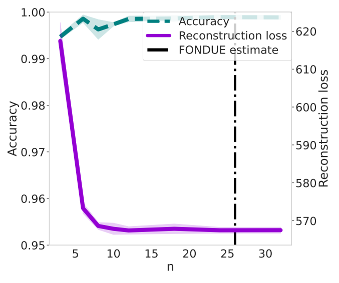
	Dataset	n (avg $\pm$ SD)	Time/run	Models trained	Epochs/training
BS$_{FONDUE}$	Symsol	19.2 $\pm$ 0.6	6 min	8	1
BS$_{FONDUE}$	dSprites	11.3 $\pm$ 0.8	75 min	8	2
BS$_{FONDUE}$	Celeba	32.2 $\pm$ 1.0	21 min	8	2
IB	Symsol	23.6 $\pm$ 0.8	40 min	8	6
IB	dSprites	15.6 $\pm$ 0.5	74 min	8	2
IB	Celeba	18.5 $\pm$ 0.5	31 min	8	3

Algorithm 8 BS$_{FONDUE}$

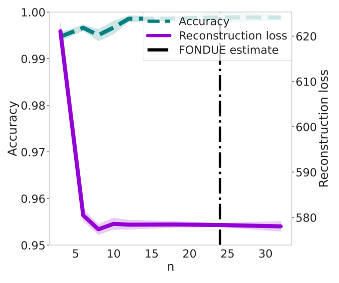
```
1:  procedure BS_{FONDUE}($t, l, u, e$)
2:      $m \leftarrow \{\}$
3:      while $l < u$ do
4:          $n \leftarrow \text{floor}\left(\frac{l+u}{2}\right)$
5:          $\text{Tr}(\text{Var}[\boldsymbol{Z}]), \text{Tr}(\text{Var}[\boldsymbol{M}]) \leftarrow$ GET-MEM$(m, n, e)$
6:          if $\text{Tr}(\text{Var}[\boldsymbol{Z}]) - \text{Tr}(\text{Var}[\boldsymbol{M}]) \leqslant t$ then
7:              $l \leftarrow n + 1$
8:          else
9:              $u \leftarrow n$
10:         end if
11:     end while
12:     return $l - 1$
13: end procedure
```

D.4 GENERALISATION TO DIFFERENT LEARNING OBJECTIVES AND HYPERPARAMETER VALUES

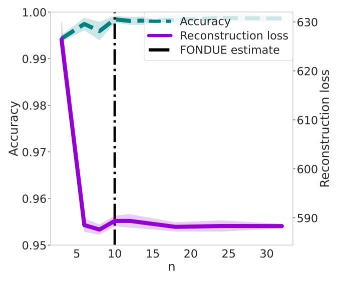
This section provides additional figures comparing β-VAEs with different β values and DIP-VAEs II with different λ_{od} values with Elbow methods. Overall, we can see in Figures D.1 to D.4 that FONDUE provides consistent results across hyperparameter values and learning objectives, with some overestimation for Symsol as observed in Section 8.5.2.



(a) Symsol, β-VAE, $\beta = 0.5$

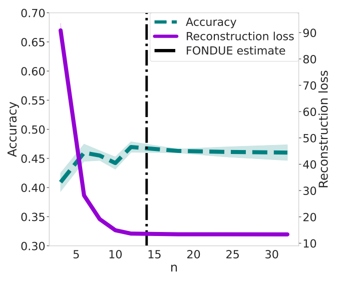


(b) Symsol, β-VAE, $\beta = 2$

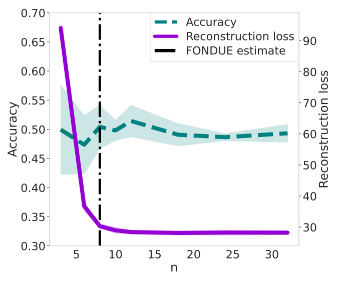


(c) Symsol, β-VAE, $\beta = 4$

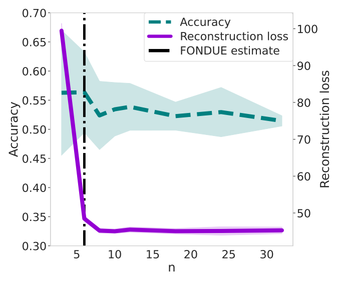
Figure D.1: Reconstruction loss and accuracy obtained for generation and downstream tasks with an increasing number of latent variables on Symsol with β-VAE. (a) shows the results for $\beta = 0.5$, (b) for $\beta = 2$, and (c) for $\beta = 4$



(a) dSprites, β-VAE, $\beta = 0.5$

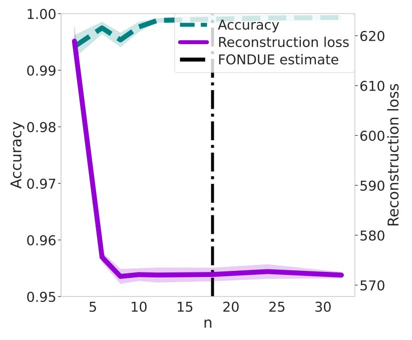


(b) dSprites, β-VAE, $\beta = 2$

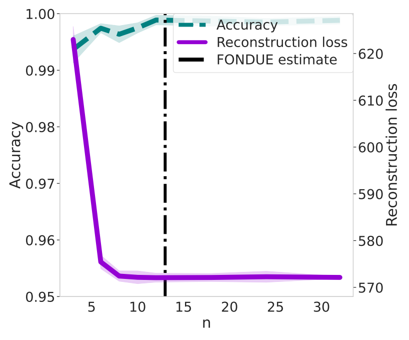


(c) dSprites, β-VAE, $\beta = 4$

Figure D.2: Reconstruction loss and accuracy obtained for generation and downstream tasks with an increasing number of latent variables on dSprites with β-VAE. (a) shows the results for $\beta = 0.5$, (b) for $\beta = 2$, and (c) for $\beta = 4$

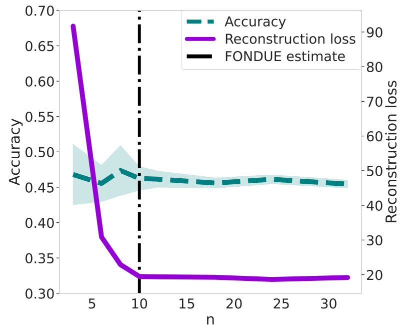


(a) Symsol, DIP-VAE II, $\lambda_{od} = 0.5$

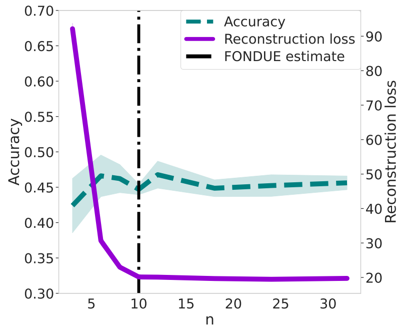


(b) Symsol, DIP-VAE II, $\lambda_{od} = 2$

Figure D.3: Reconstruction loss and accuracy obtained for generation and downstream tasks with an increasing number of latent variables on Symsol with DIP-VAE II. (a) shows the results for $\lambda_{od} = 0.5$ and (b) for $\lambda_{od} = 2$.



(a) dSprites, DIP-VAE II, $\lambda_{od} = 0.5$



(b) dSprites, DIP-VAE II, $\lambda_{od} = 2$

Figure D.4: Reconstruction loss and accuracy obtained for generation and downstream tasks with an increasing number of latent variables on dSprites with DIP-VAE II. (a) shows the results for $\lambda_{od} = 0.5$ and (b) for $\lambda_{od} = 2$.

BIBLIOGRAPHY

Adel, T., Ghahramani, Z. and Weller, A. (2018). Discovering interpretable representations for both deep generative and discriminative models. In *Proceedings of the 35th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 80, pp. 50–59.

Alemi, A. A., Fischer, I., Dillon, J. V. and Murphy, K. (2017). Deep Variational Information Bottleneck. In *International Conference on Learning Representations*, vol. 5.

Alemi, A. A., Poole, B., Fischer, I., Dillon, J. V., Saurous, R. A. and Murphy, K. (2018). Fixing a Broken ELBO. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, *Proceedings of Machine Learning Research*, vol. 80.

Ali, S. M. and Silvey, S. D. (1966). A General Class of Coefficients of Divergence of One Distribution from Another. *Journal of the Royal Statistical Society Series B (Methodological)*, 28(1), pp. 131–142.

Alin, A. (2010). Multicollinearity. *WIREs Computational Statistics*, 2(3), pp. 370–374.

Ansuini, A., Laio, A., Macke, J. H. and Zoccolan, D. (2019). Intrinsic dimension of data representations in deep neural networks. In *Advances in Neural Information Processing Systems*, vol. 32.

Arora, S., Cohen, N. and Hazan, E. (2018). On the optimization of deep networks: Implicit acceleration by overparameterization. In J. Dy and A. Krause, eds., *Proceedings of the 35th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 80, PMLR, pp. 244–253.

Bansal, Y., Nakkiran, P. and Barak, B. (2021). Revisiting Model Stitching to Compare Neural Representations. In *Advances in Neural Information Processing Systems*.

Barocas, S., Hardt, M. and Narayanan, A. (2019). *Fairness and Machine Learning: Limitations and Opportunities*. fairmlbook.org, http://www.fairmlbook.org.

Bengio, Y., Courville, A. and Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8).

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V. and Kalai, A. T. (2016). Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *Advances in Neural Information Processing Systems*, vol. 29.

Bond-Taylor, S. and Willcocks, C. G. (2021). Gradient origin networks. In *International Conference on Learning Representations*.

Bonheme, L. and Grzes, M. (2023). The polarised regime of identifiable variational autoencoders. *ICLR TinyPapers*.

Boquet, G., Macias, E., Morell, A., Serrano, J. and Vicario, J. L. (2021). Theoretical tuning of the autoencoder bottleneck layer dimension: A mutual information-based algorithm. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pp. 1512–1516.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R. and Bengio, S. (2016). Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*.

Brualdi, R. A. and Schneider, H. (1983). Determinantal identities: Gauss, Schur, Cauchy, Sylvester, Kronecker, Jacobi, Binet, Laplace, Muir, and Cayley. *Linear Algebra and its Applications*, 52-53.

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G. and Lerchner, A. (2018). Understanding Disentangling in β-VAE. *arXiv e-prints*, 1804.03599.

Caliskan, A., Bryson, J. J. and Narayanan, A. (2017). Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334).

Campadelli, P., Casiraghi, E., Ceruti, C. and Rozza, A. (2015). Intrinsic dimension estimation: Relevant techniques and a benchmark framework. *Mathematical Problems in Engineering*, 2015.

Campbell, L. L. (1960). Minimum coefficient rate for stationary random processes. *Information and Control*, 3(4).

Chan, J. Y.-L., Leow, S. M. H., Bea, K. T., Cheng, W. K., Phoong, S. W., Hong, Z.-W. and Chen, Y.-L. (2022). Mitigating the multicollinearity problem and its machine learning approach: A review. *Mathematics*, 10(8).

Chen, R. T. Q., Li, X., Grosse, R. B. and Duvenaud, D. K. (2018). Isolating Sources of Disentanglement in Variational Autoencoders. In *Advances in Neural Information Processing Systems*, vol. 31.

Cheng, Z., Li, J., Wang, C., Gu, J., Xu, H., Li, X. and Metze, F. (2021). Revisiting factorizing aggregated posterior in learning disentangled representations. *arXiv e-prints*, 2009.05739.

Chien, J.-T. and Wang, C.-W. (2019). Variational and Hierarchical Recurrent Autoencoder. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019*.

Chollet, F. (2021). *Deep Learning with Python, Second Edition*. Manning.

Chu, C., Blanchet, J. and Glynn, P. (2019). Probability functional descent: A unifying perspective on GANs, variational inference, and reinforcement learning. In K. Chaudhuri and R. Salakhutdinov, eds., *Proceedings of the 36th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 97, PMLR, pp. 1213–1222.

Cortes, C., Mohri, M. and Rostamizadeh, A. (2012). Algorithms for Learning Kernels Based on Centered Alignment. *Journal of Machine Learning Research*, 13(1).

Cover, T. M. (1999). *Elements of information theory*. John Wiley & Sons.

Creager, E., Madras, D., Jacobsen, J.-H., Weis, M., Swersky, K., Pitassi, T. and Zemel, R. (2019). Flexibly fair representation learning by disentanglement. In *Proceedings of the 36th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 97, pp. 1436–1445.

Csiszár, I. (1967). Information-type measures of difference of probability distributions and indirect observation. *studia scientiarum Mathematicarum Hungarica*, 2, pp. 229–318.

Csiszárik, A., Kőrösi-Szabó, P., Matszangosz, Á. K., Papp, G. and Varga, D. (2021). Similarity and matching of neural network representations. In *Advances in Neural Information Processing Systems*.

Dai, B., Wang, Z. and Wipf, D. (2020). The Usual Suspects? Reassessing Blame for VAE Posterior Collapse. In *Proceedings of the 37th International Conference on Machine Learning*.

Dai, B. and Wipf, D. (2018). Diagnosing and Enhancing VAE Models. In *International Conference on Learning Representations*, vol. 6.

Dai, B., Wang, Y., Aston, J., Hua, G. and Wipf, D. (2017). Hidden Talents of the Variational Autoencoder. *arXiv e-prints*, 1706.05148.

Dai, B., Wang, Y., Aston, J., Hua, G. and Wipf, D. (2018). Connections with Robust PCA and the Role of Emergent Sparsity in Variational Autoencoder Models. *Journal of Machine Learning Research*, 19(41), pp. 1–42.

Deisenroth, M. P., Faisal, A. A. and Ong, C. S. (2020). *Mathematics for machine learning*. Cambridge University Press.

Ding, F., Denain, J.-S. and Steinhardt, J. (2021). Grounding Representation Similarity Through Statistical Testing. In *Advances in Neural Information Processing Systems*.

Doersch, C. (2016). Tutorial on Variational Autoencoders. *arXiv e-prints*, 1606.05908.

Escoufier, Y. (1973). Le traitement des variables vectorielles. *Biometrics*, 29(4), pp. 751–760.

Facco, E., dÉrrico, M., Rodriguez, A. and Laio, A. (2017). Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific Reports*, 7(1).

Floto, G., Kremer, S. and Nica, M. (2023). The tilted variational autoencoder: Improving out-of-distribution detection. In *The Eleventh International Conference on Learning Representations*.

Garg, P., Villasenor, J. and Foggo, V. (2020). Fairness metrics: A comparative analysis. In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 3662–3666.

Gentle, J. E. (2009). *Computational Statistics*. Springer New York.

Golub, G. H. and Van Loan, C. F. (2013). *Matrix computations*. The Johns Hopkins University Press, fourth edition edn.

Gonen, H. and Goldberg, Y. (2019). Lipstick on a Pig: Debiasing Methods Cover up Systematic Gender Biases in Word Embeddings But do not Remove Them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Gong, S., Boddeti, V. N. and Jain, A. K. (2019). On the intrinsic dimensionality of image representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3987–3996.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence and K. Weinberger, eds., *Advances in Neural Information Processing Systems*, vol. 27, Curran Associates, Inc.

Gresele, L., Kügelgen, J. V., Stimper, V., Schölkopf, B. and Besserve, M. (2021). Independent mechanism analysis, a new concept? In A. Beygelzimer, Y. Dauphin, P. Liang and J. W. Vaughan, eds., *Advances in Neural Information Processing Systems*.

Gupta, P., Banchs, R. E. and Rosso, P. (2016). Squeezing bottlenecks: Exploring the limits of autoencoder semantic representation capabilities. *Neurocomputing*, 175, pp. 1001–1008.

Hadamard, J. (1893). Résolution d'une question relative aux déterminants. *Bulletin des Sciences Mathématiques*, 2, pp. 240–246.

He, J., Spokoyny, D., Neubig, G. and Berg-Kirkpatrick, T. (2019). Lagging Inference Networks and Posterior Collapse in Variational Autoencoders. In *International Conference on Learning Representations*, vol. 7.

Hellinger, E. (1909). Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136), pp. 210–271.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Shakir, M. and Lerchner, A. (2017). β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations*, vol. 5.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An introduction to statistical learning*, vol. 112. Springer.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London Series A Mathematical and Physical Sciences*, 186(1007), pp. 453–461.

Josse, J. and Holmes, S. (2016). Measuring multivariate association and beyond. *Statistics Surveys*, 10, pp. 132 – 167.

Kailath, T. (1967). The divergence and bhattacharyya distance measures in signal selection. *IEEE Transactions on Communication Technology*, 15(1), pp. 52–60.

Karbauskaitė, R., Dzemyda, G. and Mazėtis, E. (2011). Geodesic distances in the maximum likelihood estimator of intrinsic dimensionality. *Nonlinear Analysis*, 16(4), pp. 387–402.

Khemakhem, I., Kingma, D., Monti, R. and Hyvarinen, A. (2020). Variational Autoencoders and Nonlinear ICA: A Unifying Framework. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, *Proceedings of Machine Learning Research*, vol. 108.

Kim, H. and Mnih, A. (2018). Disentangling by Factorising. In *Proceedings of the 35th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 80.

Kingma, D. P., Salimans, T. and Welling, M. (2015). Variational dropout and the local reparameterization trick. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama and R. Garnett, eds., *Advances in Neural Information Processing Systems*, vol. 28, Curran Associates, Inc.

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, vol. 2.

Klushyn, A., Chen, N., Kurle, R., Cseke, B. and van der Smagt, P. (2019). Learning hierarchical priors in vaes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, eds., *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc.

Kornblith, S., Norouzi, M., Lee, H. and Hinton, G. (2019). Similarity of Neural Network Representations Revisited. In *Proceedings of the 36th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 97.

Kudugunta, S., Bapna, A., Caswell, I. and Firat, O. (2019). Investigating Multilingual NMT Representations at Scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1), pp. 79–86.

Kumar, A., Sattigeri, P. and Balakrishnan, A. (2018). Variational Inference of Disentangled Latent Concepts from Unlabeled Observations. In *International Conference on Learning Representations*, vol. 6.

Lacoste, A., Luccioni, A., Schmidt, V. and Dandres, T. (2019). Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:191009700*.

LeCun, Y., Huang, F. J. and Bottou, L. (2004). Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2.

Lee, S. and Jo, J. (2021). Information Flows of Diverse Autoencoders. *Entropy*, (7).

Levina, E. and Bickel, P. J. (2004). Maximum Likelihood Estimation of Intrinsic Dimension. In *Advances in Neural Information Processing Systems*, vol. 16.

Li, Z., Murkute, J. V., Gyawali, P. K. and Wang, L. (2020). Progressive Learning and Disentanglement of Hierarchical Representations. In *International Conference on Learning Representations*.

Lin, J. (1991). Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1), pp. 145–151.

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

Liu, Z.-S., Siu, W.-C. and Wang, L.-W. (2021). Variational autoencoder for reference based image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 516–525.

Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B. and Bachem, O. (2019a). Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *Proceedings of the 36th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 97.

Locatello, F., Abbati, G., Rainforth, T., Bauer, S., Schölkopf, B. and Bachem, O. (2019b). On the Fairness of Disentangled Representations. In *Advances in Neural Information Processing Systems*, vol. 32.

Locatello, F., Tschannen, M., Bauer, S., Rätsch, G., Schölkopf, B. and Bachem, O. (2020). Disentangling Factors of Variations Using Few Labels. In *International Conference on Learning Representations*, vol. 8.

Lucas, J., Tucker, G., Grosse, R. B. and Norouzi, M. (2019a). Don't Blame the ELBO! A linear VAE Perspective on Posterior Collapse. In *Advances in Neural Information Processing Systems*, vol. 32.

Lucas, J., Tucker, G., Grosse, R. B. and Norouzi, M. (2019b). Understanding Posterior Collapse in Generative Latent Variable Models. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop*.

MacKay, D. J. and Ghahramani, Z. (2005). Comments on 'Maximum Likelihood Estimation of Intrinsic Dimension' by E. Levina and P. Bickel (2004).

Maheswaranathan, N., Williams, A., Golub, M., Ganguli, S. and Sussillo, D. (2019). Universality and Individuality in Neural Dynamics Across Large Populations of Recurrent Networks. In *Advances in Neural Information Processing Systems*, vol. 32.

Mai Ngoc, K. and Hwang, M. (2020). Finding the best k for the dimension of the latent space in autoencoders. In *Computational Collective Intelligence*, Springer International Publishing, pp. 453–464.

Martin, K. (2022). *Ethics of data and analytics: Concepts and cases*. CRC Press.

Mathieu, E., Rainforth, T., Siddharth, N. and Teh, Y. W. (2019). Disentangling Disentanglement in Variational Autoencoders. In *Proceedings of the 36th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 97.

Mercatali, G. and Freitas, A. (2021). Disentangling generative factors in natural language with discrete variational autoencoders. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 3547–3556.

Mita, G., Filippone, M. and Michiardi, P. (2021). An Identifiable Double VAE For Disentangled Representations. In *Proceedings of the 38th International Conference* *on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 139, pp. 7769–7779.

Mohamed, S., Rosca, M., Figurnov, M. and Mnih, A. (2020). Monte Carlo Gradient Estimation in Machine Learning. *Journal of Machine Learning Research*, 21(1).

Morcos, A., Raghu, M. and Bengio, S. (2018). Insights on Representational Similarity in Neural Networks with Canonical Correlation. In *Advances in Neural Information Processing Systems*, vol. 31.

Moschella, L., Maiorca, V., Fumero, M., Norelli, A., Locatello, F. and Rodolà, E. (2023). Relative representations enable zero-shot latent space communication. In *The Eleventh International Conference on Learning Representations*.

Murphy, K., Esteves, C., Jampani, V., Ramalingam, S. and Makadia, A. (2021). Implicit representation of probability distributions on the rotation manifold. In *International Conference on Machine Learning*.

Murphy, K. P. (2022). *Probabilistic Machine Learning: An introduction*. MIT Press.

Neyshabur, B., Sedghi, H. and Zhang, C. (2020). What is Being Transferred in Transfer Learning? In *Advances in Neural Information Processing Systems*, vol. 33.

Paisley, J., Blei, D. M. and Jordan, M. I. (2012). Variational Bayesian Inference with Stochastic Search. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, Madison, WI, USA: Omnipress, ICML'12, pp. 1363–1370.

Petersen, K. B., Pedersen, M. S. et al. (2008). The matrix cookbook. *Technical University of Denmark*, 7(15), p. 510.

Pope, P., Zhu, C., Abdelkader, A., Goldblum, M. and Goldstein, T. (2021). The Intrinsic Dimension of Images and Its Impact on Learning. In *International Conference on Learning Representations*, vol. 9.

Pozrikidis, C. (2014). *An introduction to grids, graphs, and networks*. Oxford University Press.

Purdom, E. (2006). *Multivariate kernel methods in the analysis of graphical structures*. Ph.D. thesis, Ann Arbor.

Raghu, M., Gilmer, J., Yosinski, J. and Sohl-Dickstein, J. (2017). SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and Interpretability. In *Advances in Neural Information Processing Systems*, vol. 30.

Raghu, M., Zhang, C., Kleinberg, J. and Bengio, S. (2019). Transfusion: Understanding Transfer Learning for Medical Imaging. In *Advances in Neural Information Processing Systems*, vol. 32.

Ranganath, R., Tran, D. and Blei, D. (2016). Hierarchical variational models. In M. F. Balcan and K. Q. Weinberger, eds., *Proceedings of The 33rd International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 48, New York, New York, USA: PMLR, pp. 324–333.

Reed, S., Zhang, Y., Zhang, Y. and Lee, H. (2015). Deep Visual Analogy-Making. In *Advances in Neural Information Processing Systems*, vol. 28.

Reizinger, P., Gresele, L., Brady, J., Kügelgen, J. V., Zietlow, D., Schölkopf, B., Martius, G., Brendel, W. and Besserve, M. (2022). Embrace the gap: VAEs perform independent mechanism analysis. In *Advances in Neural Information Processing Systems*, vol. 36.

Rényi, A. (1961). On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, vol. 4, University of California Press, pp. 547–562.

Rezende, D. J., Mohamed, S. and Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 32.

Robert, P. and Escoufier, Y. (1976). A Unifying Tool for Linear Multivariate Statistical Methods: The RV- Coefficient. *Journal of the Royal Statistical Society Series C (Applied Statistics)*, 25(3), pp. 257–265.

Rolinek, M., Zietlow, D. and Martius, G. (2019). Variational Autoencoders Pursue PCA Directions (by Accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Roy, O. and Vetterli, M. (2007). The Effective Rank: a Measure of Effective Dimensionality. In *2007 15th European Signal Processing Conference*.

Sankararaman, K. A., De, S., Xu, Z., Huang, W. R. and Goldstein, T. (2020). The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In H. D. III and A. Singh, eds., *Proceedings of the*

37th International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 119, PMLR, pp. 8469–8479.

Schönemann, P. H. (1966). A generalized solution of the Orthogonal Procrustes problem. *Psychometrika*, 31(1), pp. 1–10.

Searle, S. R. and Khuri, A. I. (2017). *Matrix algebra useful for statistics*. John Wiley & Sons.

Silverman, B. W. (1998). *Density estimation for statistics and data analysis*. Routledge.

Slonim, N., Atwal, G. S., Tkačik, G. and Bialek, W. (2005). Information-based clustering. *Proceedings of the National Academy of Sciences*, 102(51), pp. 18297–18302, https://www.pnas.org/content/102/51/18297.full.pdf.

Sø nderby, C. K., Raiko, T., Maalø e, L., Sø nderby, S. r. K. and Winther, O. (2016). Ladder variational autoencoders. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon and R. Garnett, eds., *Advances in Neural Information Processing Systems*, vol. 29, Curran Associates, Inc.

Tipping, M. E. and Bishop, C. M. (1999). Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3), https://rss.onlinelibrary.wiley.com/optdoi/pdf/10.1111/1467-9868.00196.

Tishby, N., Pereira, F. C. and Bialek, W. (2000). The Information Bottleneck Method. *arXiv e-prints*, physics/0004057.

Tishby, N. and Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*.

Träuble, F., Creager, E., Kilbertus, N., Locatello, F., Dittadi, A., Goyal, A., Schölkopf, B. and Bauer, S. (2021). On disentangled representations learned from correlated data. In *Proceedings of the 38th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 139, pp. 10401–10412.

Vahdat, A. and Kautz, J. (2020). Nvae: A deep hierarchical variational autoencoder. In *Advances in Neural Information Processing Systems*, vol. 33, Curran Associates, Inc., pp. 19667–19679.

van Steenkiste, S., Locatello, F., Schmidhuber, J. and Bachem, O. (2019). Are Disentangled Representations Helpful for Abstract Visual Reasoning? In *Advances in Neural Information Processing Systems*, vol. 32.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y. and Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(110), pp. 3371–3408.

Wan, N., Li, D. and Hovakimyan, N. (2020). f-divergence variational inference. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan and H. Lin, eds., *Advances in Neural Information Processing Systems*, vol. 33, Curran Associates, Inc., pp. 17370–17379.

Wang, L., Hu, L., Gu, J., Wu, Y., Hu, Z., He, K. and Hopcroft, J. (2019). Towards understanding learning representations: To what extent do different neural networks learn the same representation. In *Advances in Neural Information Processing Systems*, vol. 32.

Watanabe, S. (1960). Information Theoretical Analysis of Multivariate Correlation. *IBM Journal of Research and Development*, 4(1).

Wilson, B., Hoffman, J. and Morgenstern, J. (2019). Predictive inequity in object detection. *arXiv e-print*, 1902.11097.

Xu, M., Quiroz, M., Kohn, R. and Sisson, S. A. (2019). Variance reduction properties of the reparameterization trick. In K. Chaudhuri and M. Sugiyama, eds., *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, *Proceedings of Machine Learning Research*, vol. 89, PMLR, pp. 2711–2720.

Yang, W., Gibson, J. and He, T. (2005). Coefficient rate and lossy source coding. *IEEE Transactions on Information Theory*, 51(1).

Yosinski, J., Clune, J., Bengio, Y. and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, vol. 27.

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T. and Lipson, H. (2015). Understanding neural networks through deep visualization. *arXiv e-prints*, 1506.06579.

Yu, S. and Príncipe, J. C. (2019). Understanding autoencoders with information theoretic concepts. *Neural Networks*, 117, pp. 104–123.

Zietlow, D., Rolinek, M. and Martius, G. (2021). Demystifying Inductive Biases for (Beta-) VAE Based Architectures. In *Proceedings of the 38th International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, vol. 139.