HOW DO VARIATIONAL AUTOENCODERS LEARN? INSIGHTS FROM REPRESENTATIONAL SIMILARITY

Anonymous authors
Paper under double-blind review

ABSTRACT

The ability of Variational Autoencoders (VAEs) to learn disentangled representations has made them popular for practical applications. However, their behaviour is not yet fully understood. For example, the questions of when they can provide disentangled representations, or suffer from posterior collapse are still areas of active research. Despite this, there are no layerwise comparisons of the representations learned by VAEs, which would further our understanding of these models. In this paper, we thus look into the internal behaviour of VAEs using representational similarity techniques. Specifically, using the CKA and Procrustes similarities, we found that the encoders' representations are learned long before the decoders', and this behaviour is independent of hyperparameters, learning objectives, and datasets. Moreover, the encoders' representations in all but the mean and variance layers are similar across hyperparameters and learning objectives.

1 INTRODUCTION

Variational Autoencoders (VAEs) are considered state-of-the-art techniques to learn unsupervised disentangled representations, and have been shown to be beneficial for fairness (Locatello et al., 2019a). As a result, VAEs producing disentangled representations have been extensively studied in the last few years (Locatello et al., 2019b; Mathieu et al., 2019; Rolinek et al., 2019; Zietlow et al., 2021), but they still suffer from poorly understood issues such as posterior collapse (Dai et al., 2020). While some work using explainability techniques has been done to shed light on the behaviour of VAEs (Liu et al., 2020), a comparison of the representations learned by different methods is still lacking (Zietlow et al., 2021). Moreover, the layer-by-layer similarity of the representations within models has yet to be investigated.

Fortunately, the domain of deep representational similarity is an active area of research and metrics such as SVCCA (Raghu et al., 2017; Morcos et al., 2018), Procrustes distance (Schönemann, 1966), or Centred Kernel Alignment (CKA) (Kornblith et al., 2019) have proven very useful in analysing the learning dynamics of various models (Wang et al., 2019; Kudugunta et al., 2019; Raghu et al., 2019; Neyshabur et al., 2020), and even helped to design UDR (Duan et al., 2020), an unsupervised metric for model selection for VAEs.

The fact that good models are more similar to each other than bad ones in the context of classification (Morcos et al., 2018) generalised well to Unsupervised Disentanglement Ranking (UDR) (Rolinek et al., 2019; Duan et al., 2020). However, such a generalisation may not always be possible, and without sufficient evidence, it would be wise to expect substantial differences between the representations learned by supervised and unsupervised models. In this paper, our aim is to take a first step toward investigating the representational similarity of generative models by analysing the similarity scores obtained for a variety of VAEs learning disentangled representations, and by providing some insights into why various VAE-specific methods preventing posterior collapse (Bowman et al., 2016; He et al., 2019) or providing better reconstruction (Liu et al., 2021) are successful.

Our contributions are as follows:

(i) We provide the first experimental study of the representational similarity between VAEs, and have released more than 45 million similarity scores (`https://t.ly/0GLe3`[1]).
(ii) We have released the library created for this experiment (`https://t.ly/VMIm`). It can be reused with other similarity metrics or models for further research in the domain.
(iii) During our analysis, we found that (1) the encoder is learned before the decoder; (2) all the layers of the encoder, except the mean and variance layers, learn very similar representations regardless of the learning objective and regularisation strength used; and (3) linear CKA could be an efficient tool to track posterior collapse.

2 BACKGROUND

2.1 VARIATIONAL AUTOENCODERS

Variational Autoencoders (VAEs) (Kingma & Welling, 2014; Rezende & Mohamed, 2015) are deep probabilistic generative models based on variational inference. The encoder, $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, maps some input $\mathbf{x}$ to a latent representation $\mathbf{z}$, which the decoder, $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, uses to attempt to reconstruct $\mathbf{x}$. This can be optimised by maximising $\mathcal{L}$, the evidence lower bound (ELBO)

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)}_{\text{regularisation term}}, \tag{1}$$

where $p(\mathbf{z})$ is generally modelled as a multivariate Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$ to permit closed form computation of the regularisation term (Doersch, 2016). We refer to the regularisation term of Equation 1 as regularisation in the rest of the paper, and we do not tune any other forms of regularisation (e.g., L1, dropout).

Polarised regime and posterior collapse 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 the remaining ones (Rolinek et al., 2019; Dai et al., 2020). The existence of the polarised regime is a necessary condition for the VAEs to provide a good reconstruction (Dai & Wipf, 2018; Dai et al., 2020). However, when the weight on the regularisation term of the ELBO given in Equation 1 becomes too large, the representations collapse to the prior (Lucas et al., 2019a; Dai et al., 2020). Recently, Bonheme & Grzes (2021) have also shown that the passive variables, which are "shut down" during training, are very different in mean and sampled representations (see Appendix B). This indicates that representational similarity could be a valuable tool in the study of posterior collapse.

2.2 REPRESENTATIONAL SIMILARITY METRICS

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).

Centred Kernel Alignment Centred Kernel Alignment (CKA) (Cortes et al., 2012; Cristianini et al., 2002) is a normalised version of the Hillbert-Schmit Independence Criterion (HSIC) (Gretton et al., 2005). As its name suggests, it 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 (Robert & Escoufier, 1976). Given the centered layer activations $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times p}$ taken over n data examples, linear CKA is defined as:

$$CKA(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\|\boldsymbol{Y}^T \boldsymbol{X}\|_F^2}{\|\boldsymbol{X}^T \boldsymbol{X}\|_F \|\boldsymbol{Y}^T \boldsymbol{Y}\|_F},$$

where $\|\cdot\|_F$ is the Frobenius norm. For conciseness, we will refer to linear CKA as CKA in the rest of this paper. Note that CKA takes values between 0 (not similar) and 1 ($\boldsymbol{X} = \boldsymbol{Y}$).

[1] Due to their size and to preserve anonymity, the 300 models trained for this paper will be released after the review.

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

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

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

$$P_d(\boldsymbol{X}, \boldsymbol{Y}) = \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\|_F^2 - 2\|\boldsymbol{Y}^T\boldsymbol{X}\|_*, \tag{3}$$

where $\|\cdot\|_*$ is the nuclear norm (see Golub & Van Loan (2013, pp. 327-328) for the full derivation from Equation 2 to Equation 3). To easily compare the results of Equation 3 with CKA, we first bound its results between 0 and 2 using normalised $\dot{\boldsymbol{X}}$ and $\dot{\boldsymbol{Y}}$, as detailed in Appendix C. Then, we transform the result to a similarity metric ranging from 0 (not similar) to 1 ($\boldsymbol{X} = \boldsymbol{Y}$),

$$P_s(\boldsymbol{X}, \boldsymbol{Y}) = 1 - \frac{1}{2}\left(\|\dot{\boldsymbol{X}}\|_F^2 + \|\dot{\boldsymbol{Y}}\|_F^2 - 2\|\dot{\boldsymbol{Y}}^T\dot{\boldsymbol{X}}\|_*\right). \tag{4}$$

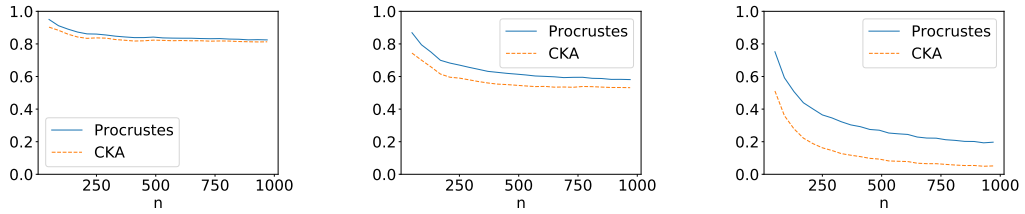
We will refer to Equation 4 as Procrustes similarity in the following sections.

2.3 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 paper, $sim(\cdot, \cdot)$ 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 1a and 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 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 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 et al. (2021).

(a) Representation $\boldsymbol{A}$

(b) Representation $\boldsymbol{B}$

Figure 2: (a) shows a disentangled representation $\boldsymbol{A}$ where the latent dimensions $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are aligned with some ground truth factors shape and scale. (b) shows a representation $\boldsymbol{B}$ which corresponds to an orthogonal transformation of $\boldsymbol{A}$. It is entangled as $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are not aligned with the ground truth factors anymore, but $sim(\boldsymbol{A}, \boldsymbol{B}) = 1$ because Procrustes and CKA are invariant under orthogonal transformations.

Similarity and disentanglement It is important to keep in mind that CKA and Procrustes similarities are invariant to orthogonal transformation. Thus, they will consider a disentangled representation similar to a rotated (and possibly entangled) representation, as illustrated in Figure 2. Note that CKA and Procrustes are also invariant to isotropic scaling, though this does not affect disentanglement.

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. Lastly, when we say that representations are similar, we mean that they are similar up to orthogonal rotations and isotropic scaling. Hence, two similar representations may not be equally disentangled.

3 Experimental setup

The goal of this experiment is to study the training dynamics of VAEs and the impact of initialisation, learning objectives, and regularisation on the representations learned by each layer. To do so, we measure the representational similarity of:

(i) One model at different epochs.
(ii) Two models with the same learning objective and different regularisation strengths at the same epoch.
(iii) Two models with different learning objectives and equivalent regularisation strength at the same epoch.

Learning objectives We will focus on learning objectives whose goal is to produce disentangled representations. Specifically, we use β-VAE (Higgins et al., 2017), β-TC VAE (Chen et al., 2018), Annealed VAE (Burgess et al., 2018), and DIP-VAE II (Kumar et al., 2018). A description of these methods can be found in Appendix A. To fairly provide complementary insights into previous observations of such models (Locatello et al., 2019b; Bonheme & Grzes, 2021), we will follow the experimental design of Locatello et al. (2019b) regarding the architecture, learning objectives, and regularisation used. Moreover, `disentanglement lib`[2] will be used as a codebase for our experiment. The complete details are available in Appendix C.

[2] `https://github.com/google-research/disentanglement_lib`

Datasets We use three datasets which, based on the results of Locatello et al. (2019b), are increasingly difficult for VAEs in terms of reconstruction loss: dSprites[3] (Higgins et al., 2017), cars3D (Reed et al., 2015), and smallNorb (LeCun et al., 2004).

Training process 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 I explains our epoch selection methodology.

Similarity measurement Given the computational complexity detailed below, 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. We will refer to the similarity scores of a group of one or more layers with itself as *self-similarity*. 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 obtained similar results for the two metrics on cars3D, thus we only reported CKA results in the main paper. Procrustes results can be found in Appendix D.

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. Overall, 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 our metric scores at `https://t.ly/0GLe3`. We hope that this will help to prevent unnecessary recomputation should others wish to reuse our results.

4 Results

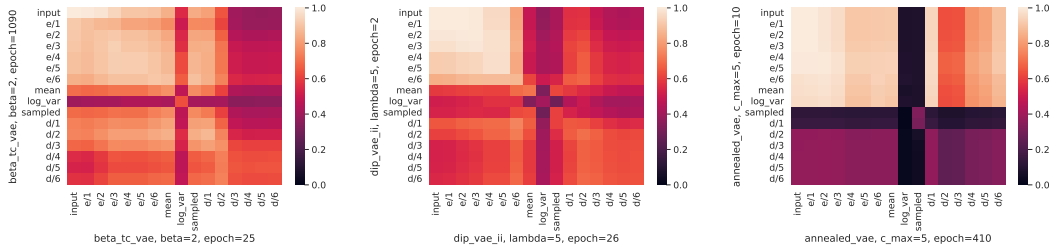
4.1 How are representations learned as training progresses?

In this section, we will analyse the learning dynamics of VAEs to answer objective (i) of Section 3. To monitor this, we will compare the representations learned by the first and last recorded snapshot of VAEs. Note that our choice of snapshots and snapshot frequency does not influence the results as verified in Appendices I and J.

VAE learning is bottom-up As shown in Figure 3, the encoder is learned first, and the representations of its layers become similar to the input after a few epochs. The decoder then progressively learns representations that gradually become closer to the input. We observed a similar trend on fully-connected architectures, as reported in Appendix F. This result can explain why, when a decoder has access to the input (as it is the case in some autoregressive VAEs), it ignores the latent representations (Bowman et al., 2016; Li et al., 2019). Indeed, in this case the decoder is not constrained to wait for the encoder to converge before improving its reconstruction since it has direct access to the input sequences. In this paper, we used standard VAEs, where the decoder is not autoregressive and must rely on the latent representations as the sole source of information about the input. It thus needs to wait for the encoder to converge before being able to converge itself, effectively preventing posterior collapse when the regularisation is not too strong.

Implications The concurrent learning of encoder and decoder representations can be problematic. Indeed, if the encoder learns poor representations, the decoder will not be able to provide an accurate

[3]Licensed under an Apache 2.0 licence.



(a) Trained on cars3D (b) Trained on dSprites (c) Trained on smallNorb

Figure 3: (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 $c_{max} = 5$. 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)). However, the self-similarity between the decoder, mean and variance representations is lower, indicating some further changes in the representations after the first few epochs of training (see the dark cells in the bottom-right quadrant of parts (a), (b), and (c)). These results are averaged over 5 seeds.

reconstruction. As a result, the encoder may struggle to provide better representations and this may lead to non-optimal results. We believe methods that facilitate the incremental learning of the encoder and decoder (e.g., by slowly increasing the complexity of the data, as in Progressive GANs (Karras et al., 2018)) as well as methods where the encoder is explicitly learned first (He et al., 2019) are promising ways to mitigate this issue. Moreover, the early convergence of the encoder can explain the success of "warm-up" methods such as annealing (Bowman et al., 2016) or aggressive inference (He et al., 2019) in the case where the encoder (quickly) converges to a bad local minima resulting in posterior collapse.

4.2 What is the influence of the hyperparameters on the learned representations?

So far, we have compared different snapshots of the same model. In this section, we will perform a fine-grained analysis of the impact of hyperparameters to answer objective (ii) of Section 3.

(a) β-VAE (b) DIP-VAE II

Figure 4: (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 (a) and (b), the activations are taken after complete training, and all the results are averaged over 5 seeds. In both figures, we can see that the self-similarity of decoder representations is low between models trained with low and high regularisation (dark cells in the bottom-right quadrants) while the encoder representations stay very similar (bright cells in the top-left quadrants), except for the mean, variance and sampled representations.

(a) CKA (mean, sampled)

(b) CKA (input, sampled)

Figure 5: (a) and (b) show the CKA scores of β-VAEs trained on dSprites with β from 1 to 8. (a) is the CKA between mean and sampled representations while (b) is between inputs and sampled representations. Both figures are consistent with posterior collapse caused by over-regularisation (Rolinek et al., 2019; Bonheme & Grzes, 2021) where the mean and sampled representations present a growing number of passive variables which, in the case of sampled representations, leads to high dissimilarity with the input in (b).

(a) $\beta = 1$

(b) $\beta = 8$

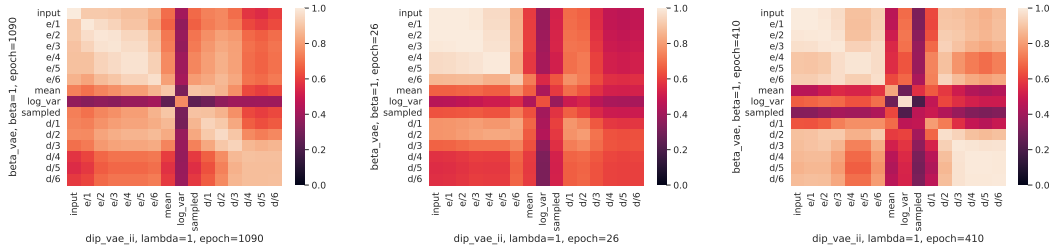
Figure 6: (a) shows the CKA scores between the inputs and the activations of the first 6 layers of the encoder of β-VAEs trained on cars3D with $\beta = 1$. (b) shows the scores between the same representations with $\beta = 8$. We can see that we have low variance across seeds (i.e., the shaded area corresponding to the variance interval around each line is small) and the encoder layers have an equivalent similarity with the input, regardless of the regularisation strength.

Impact of regularisation As discussed in (Bonheme & Grzes, 2021), passive variables have different values in the mean and sampled representations. This phenomenon is increasingly visible with higher regularisation, as more variables must become passive in order to lower the KL divergence (Rolinek et al., 2019; Dai et al., 2020). However, little is known about how this behaviour impacts the other layers of the encoder and decoder. As shown in Figure 4, the decoder representations change more than the encoder representations with the increased regularisation strength. This can be explained by Figure 5 where the sampled representations drift away from the input and mean representations, which is consistent with posterior collapse (Rolinek et al., 2019; Dai & Wipf, 2018; Dai et al., 2020). This is further confirmed by the fact that posterior collapse was already reported in (Bonheme & Grzes, 2021) for these configurations. Thus, our results indicate that CKA, which is quick to compute, could be a good tool to monitor the polarised regime and posterior collapse — its pathological counterpart — and differentiate well the two behaviours (see Appendix G). Interestingly, apart from the mean and variance, the representations learned by the encoder in Figure 4 stay very similar. We can further see in Figure 6 that this is consistent across seeds and very different regularisation strengths, and can also be observed with fully-connected architectures in Appendix F.

Implications The polarised regime (Rolinek et al., 2019; Zietlow et al., 2021; Dai et al., 2020) and posterior collapse (Dai & Wipf, 2018; Lucas et al., 2019a;b; Dai et al., 2020) do not seem to affect the representations learned by the encoder before the mean and variance layers (see Figure 6). Intuitively, this would imply that the encoder learns similar representations, regardless of the regularisation strength, and then "fine-tunes" them in its mean and variance layers.

4.3 WHAT IS THE INFLUENCE OF THE LEARNING OBJECTIVES ON THE LEARNED REPRESENTATIONS?

We have seen in Section 4.2 that models with the same learning objective (e.g., β-VAE, DIP-VAE II or any other learning objective defined in Appendix A) but different regularisation were generally learning similar representations in the encoder except for the mean and variance layers. Starting from these layers and moving toward the decoder, the representational similarity decreases as the regularisation strength increases. Indeed, passive variables have different mean and variance representations than active variables (Rolinek et al., 2019; Bonheme & Grzes, 2021) and, as their number grows, so does the dissimilarity with the representations obtained at lower regularisation, which have fewer passive variables. Now we can wonder whether this pattern is also occuring in the context of objective (iii) of Section 3. For an equivalent regularisation strength but different learning objectives, do the models still learn similar representations in the layers of the encoder that are before the mean and variance layers?



(a) Trained on cars3D (b) Trained on dSprites (c) Trained on smallNorb

Figure 7: (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. These 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 very high (CKA stays close to 1). However, 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.

The representations learned by the early layers of the encoders are similar across learning objectives As seen in Section 4.2, the representations learned by the encoder are similar across hyperparameters, with mean and variance representations being less similar for high regularisation. In this experiment, we show that this observation also holds across learning objectives. We can see in Figure 7 that the representational similarity of different learning objectives with equivalent regularisation strength is quite high in all the encoder layers except mean and variance. This result is consistent across datasets, learning objectives, and well as architectures (see Appendix F), and the learned representations are also close to what is learned by a classifier with equivalent architecture (see Appendix H).

The representations learned by the mean and variance layers can vary The similarity of the representations learned by the mean and variance layers (and consequently by the decoder) tends to vary across learning objectives and seems to be influenced by the dataset. Indeed, by looking at the center diagonal values of Figure 7, we can see that the similarity of the mean, variance, and consequently sampled representations across different learning objectives is high for cars3D and smallNorb, but low for dSprites. This may indicate that, for a given dataset, different learning objectives can find different local optima, leading to lower similarity between mean and variance representations, and ultimately to different representations in the decoder. It seems that this phenomenon is very dataset-dependent, and is especially present in dSprites (see Figure 7b), which is one of the most used datasets for disentangled representation learning. The same phenomenon can be observed with fully-connected architectures in Appendix F.

Implications The representations learned by most of the encoder layers are very similar across learning objectives, indicating that the encoder may be learning some general features from the inputs, or the only features that can be learned when the decoder performs poorly. Indeed, since the decoder initially struggles to learn, the encoder may only be able to learn very general properties. The learning of general representations in early layers is consistent with Bansal et al. (2021), who, in the context of classification, observed that neural networks were learning similar representations regardless of the initialisation, architecture or learning objective used. As such, the encoder may be viewed as a feature extractor which is fine-tuned using a mean and variance layer to produce the sampled representations that will be used by the decoder. Our observations also suggest that some learning objectives may favour distinct local optima whose existence has been previously discussed (Alemi et al., 2017; Zietlow et al., 2021). Such a model-specific choice of local optima may explain why some learning objectives obtained better disentanglement scores than others on specific datasets but performed worse on others (Locatello et al., 2019b).

5 Conclusion

Bottom-up learning and posterior collapse As reported in Section 4.1, the encoder is learned before the decoder, which could indicate that the decoder struggles to converge before the mean and variance representations are learned. This would explain why one can observe posterior collapse in a setting where the decoder has access to the input, and thus can infer the mapping on its own (Bowman et al., 2016; Li et al., 2019).

Different models encode similar representations We have seen in Section 4 that the encoders, prior to their mean and variance layers, learned remarkably similar representations regardless of the initialisation, regularisation, and learning objective used to train the model. It is especially intriguing to see that even posterior collapse does not seem to affect these representations. The representational similarity of the mean, variance, and decoder representations across learning objectives generally vary depending on the dataset, indicating that different learning objectives may find different local optima for a given dataset. Note that this behaviour is more visible on some datasets (e.g., dSprites) than others (e.g., cars3D).

Other applications While our main focus was to compare similarity of models across a variety of settings, this study also demonstrated that CKA, whose computational cost is very low, can be an efficient tool to detect posterior collapse. Indeed, one can directly compare the similarity between mean and sampled representations, which strongly decreases as the number of collapsed variables grows (Bonheme & Grzes, 2021). We believe that this could be a complementary tool to the more costly mutual information generally used for such purpose.

Limitations We limited our study to similarity metrics that measure difference in the geometry of the representations. While this gave us compelling insights, these metrics have some limitations, discussed in Section 2.3, and may underestimate the similarity between layers with different architectures (Maheswaranathan et al., 2019). We believe that further research using dynamics-based metrics, such as fixed-point topology (Maheswaranathan et al., 2019), could provide additional insights into the representations learned by VAEs.

REFERENCES

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep Variational Information Bottleneck. In *International Conference on Learning Representations*, volume 5, 2017.

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. Revisiting Model Stitching to Compare Neural Representations. In *Advances in Neural Information Processing Systems*, 2021.

Lisa Bonheme and Marek Grzes. Be More Active! Understanding the Differences between Mean and Sampled Representations of Variational Autoencoders. *arXiv e-prints*, 2021.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 2016.

Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding Disentangling in β-VAE. *arXiv e-prints*, 2018.

Ricky T. Q. Chen, Xuechen Li, Roger B. Grosse, and David K. Duvenaud. Isolating Sources of Disentanglement in Variational Autoencoders. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for Learning Kernels Based on Centered Alignment. *J. Mach. Learn. Res.*, 13(1), 2012.

Nello Cristianini, John Shawe-Taylor, André Elisseeff, and Jaz S Kandola. On Kernel-Target Alignment. In *Advances in Neural Information Processing Systems*, volume 14. 2002.

Bin Dai and David Wipf. Diagnosing and Enhancing VAE Models. In *International Conference on Learning Representations*, volume 6, 2018.

Bin Dai, Ziyu Wang, and David Wipf. The Usual Suspects? Reassessing Blame for VAE Posterior Collapse. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Frances Ding, Jean-Stanislas Denain, and Jacob Steinhardt. Grounding Representation Similarity Through Statistical Testing. In *Advances in Neural Information Processing Systems*, 2021.

Carl Doersch. Tutorial on Variational Autoencoders. *arXiv e-prints*, 2016.

Sunny Duan, Loic Matthey, Andre Saraiva, Nick Watters, Chris Burgess, Alexander Lerchner, and Irina Higgins. Unsupervised model selection for variational disentangled representation learning. In *International Conference on Learning Representations*, volume 8, 2020.

Gene H. Golub and Charles F. Van Loan. *Matrix computations*. The Johns Hopkins University Press, fourth edition edition, 2013. ISBN 9781421407944.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring Statistical Dependence with Hilbert-Schmidt Norms. In *Algorithmic Learning Theory*, 2005. ISBN 978-3-540-31696-1.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. Lagging Inference Networks and Posterior Collapse in Variational Autoencoders. In *International Conference on Learning Representations*, volume 7, 2019.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Mohamed Shakir, and Alexander Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations*, volume 5, 2017.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*, volume 6, 2018.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, volume 2, 2014.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of Neural Network Representations Revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2019.

Sneha Kudugunta, Ankur Bapna, Isaac Caswell, and Orhan Firat. 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)*, 2019.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational Inference of Disentangled Latent Concepts from Unlabeled Observations. In *International Conference on Learning Representations*, volume 6, 2018.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*, 2019.

Yann LeCun, Fu Jie Huang, and Léon Bottou. 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.*, volume 2, 2004.

Ruizhe Li, Xiao Li, Chenghua Lin, Matthew Collinson, and Rui Mao. A Stable Variational Autoencoder for Text Modelling. In *Proceedings of the 12th International Conference on Natural Language Generation*, 2019.

Wenqian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J Radke, and Octavia Camps. Towards Visually Explaining Variational Autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Zhi-Song Liu, Wan-Chi Siu, and Li-Wen Wang. 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, June 2021.

Francesco Locatello, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem. On the Fairness of Disentangled Representations. In *Advances in Neural Information Processing Systems*, volume 32, 2019a.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2019b.

James Lucas, George Tucker, Roger B. Grosse, and Mohammad Norouzi. Understanding Posterior Collapse in Generative Latent Variable Models. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop*, 2019a.

James Lucas, George Tucker, Roger B. Grosse, and Mohammad Norouzi. Don't Blame the ELBO! A linear VAE Perspective on Posterior Collapse. In *Advances in Neural Information Processing Systems*, volume 32, 2019b.

Niru Maheswaranathan, Alex Williams, Matthew Golub, Surya Ganguli, and David Sussillo. Universality and Individuality in Neural Dynamics Across Large Populations of Recurrent Networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. Disentangling Disentanglement in Variational Autoencoders. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2019.

Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on Representational Similarity in Neural Networks with Canonical Correlation. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is Being Transferred in Transfer Learning? In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and Interpretability. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding Transfer Learning for Medical Imaging. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Scott Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep Visual Analogy-Making. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Danilo Rezende and Shakir Mohamed. Variational Inference with Normalizing Flows. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, 2015.

P. Robert and Y. Escoufier. A Unifying Tool for Linear Multivariate Statistical Methods: The RV-Coefficient. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 25(3):257–265, 1976.

Michal Rolinek, Dominik Zietlow, and Georg Martius. Variational Autoencoders Pursue PCA Directions (by Accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Peter H. Schönemann. A generalized solution of the Orthogonal Procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

Liwei Wang, Lunjia Hu, Jiayuan Gu, Yue Wu, Zhiqiang Hu, Kun He, and John Hopcroft. Towards understanding learning representations: To what extent do different neural networks learn the same representation. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv e-prints*, 2015.

Dominik Zietlow, Michal Rolinek, and Georg Martius. Demystifying Inductive Biases for (Beta-) VAE Based Architectures. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 18–24 Jul 2021.

A DISENTANGLED REPRESENTATION LEARNING

As mentioned in Section 1, we are interested in the family of methods modifying the weight on the regularisation term of Equation 1 to encourage disentanglement. In our paper, the term regularisation refers to the moderation of this parameter only. To achieve this, our experiment will focus on the models described below.

β-VAE The goal of this method (Higgins et al., 2017) is to penalise the regularisation term of Equation 1 by a factor $\beta > 1$, such that

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - \beta D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})). \tag{5}$$

Annealed VAE Burgess et al. (2018) proposed to gradually increase the encoding capacity of the network during the training process. The goal is to progressively learn latent variables by decreasing order of importance. This leads to the following objective, where C is a parameter that can be understood as a channel capacity and γ is a hyper-parameter penalising the divergence, similarly to β in β-VAE:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - \gamma\left|D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) - \mathrm{C}\right|. \tag{6}$$

As the training progresses, the channel capacity C is increased, going from zero to its maximum channel capacity $\mathrm{C}_{\max}$ and allowing a higher value of the KL divergence term. VAEs that use Equation 6 as a learning objective are referred to as Annealed VAEs in this paper.

β-TC VAE Chen et al. (2018) argued that only the distance between the estimated latent factors and the prior should be penalised to encourage disentanglement, such that

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = \mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)\right] - \lambda D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z})||p(\mathbf{z})). \tag{7}$$

Here, $D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z})||p(\mathbf{z}))$ is approximated by penalising the dependencies between the dimensions of $q_{\boldsymbol{\phi}}(\mathbf{z})$:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) \approx \frac{1}{n}\sum_{i=1}^{n}\left[\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\mathbf{z})\right] - D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\boldsymbol{x}_i)||p(\mathbf{z})\right)\right] - \underbrace{\lambda D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z})||\prod_{j=1}^{\mathrm{D}} q_{\boldsymbol{\phi}}(\mathbf{z}_j))}_{\text{total correlation}}. \tag{8}$$

The total correlation of Equation 8 is then approximated over a mini-batch of samples $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ as follows:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}[\log q_{\boldsymbol{\phi}}(\mathbf{z})] \approx \frac{1}{M}\sum_{i=1}^{M}\left(\log \frac{1}{NM}\sum_{j=1}^{M} q_{\boldsymbol{\phi}}(z(\boldsymbol{x}_i)|\boldsymbol{x}_j)\right),$$

where $z(\boldsymbol{x}_i)$ is a sample from $q_{\boldsymbol{\phi}}(\mathbf{z}|\boldsymbol{x}_i)$, M is the number of samples in the mini-batch, and N the total number of input examples. $\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_i)}[\log q_{\boldsymbol{\phi}}(\mathbf{z}_i)]$ can be computed in a similar way.

DIP-VAE Similarly to Chen et al. (2018), Kumar et al. (2018) proposed to regularise the distance between $q_{\phi}(\mathbf{z})$ and $p(\mathbf{z})$ using Equation 7. The main difference is that here $D_{\mathrm{KL}}(q_{\phi}(\mathbf{z})||p(\mathbf{z}))$ 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

$$\mathrm{Cov}_{q_{\phi}(\mathbf{z})}[\mathbf{z}] = \mathrm{Cov}_{p(\mathbf{x})}\left[\mu_{\phi}(\mathbf{x})\right] + \mathbb{E}_{p(\mathbf{x})}\left[\Sigma_{\phi}(\mathbf{x})\right]. \tag{9}$$

DIP-VAE II penalises both terms of Equation 9 such that

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

where λ_d and λ_{od} are the penalisation terms for the diagonal and off-diagonal values respectively.

B ADDITIONAL DETAILS ON MEAN, VARIANCE AND SAMPLED REPRESENTATIONS

This section presents a concise illustration of what mean, variance and sampled representations are. As shown in Figure 8, the mean, variance and sampled representations are the last 3 layers of the encoder, where the sampled representation, $\boldsymbol{z}$, is the input of the decoder. These representations, specific to VAEs, influence the models' behaviour that can be quite different from other deep learning models, as shown by research on polarised regime and posterior collapse, for example.

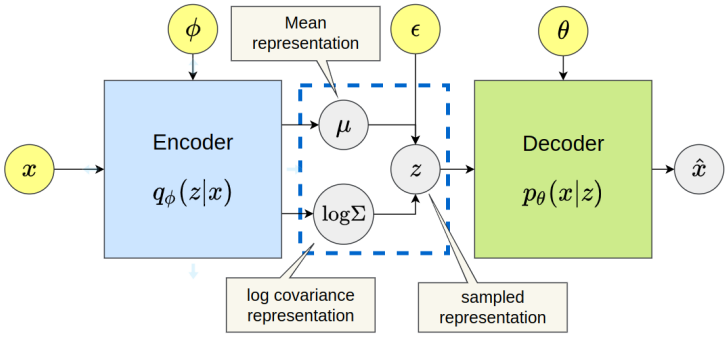


Figure 8: The structure of a VAE

C EXPERIMENTAL SETUP

To facilitate the reproducibility of our experiment, we detail below the Procrustes normalisation process and the configuration used for model training.

Procrustes normalisation Similarly to Ding et al. (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

$$\dot{\boldsymbol{X}} = \frac{\boldsymbol{X} - \bar{\boldsymbol{X}}}{\|\boldsymbol{X} - \bar{\boldsymbol{X}}\|_F}. \tag{10}$$

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), Equation 3 lies in $[0, 2]$, and Equation 4 in $[0, 1]$.

VAE training Our implementation uses the same hyperparameters as Locatello et al. (2019b), and the details are listed in Table 1 and 2. We reimplemented Locatello et al. (2019b) code base, designed for Tensorflow 1, in Tensorflow 2 using Keras. The model architecture used is also identical, as described in Table 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 F, we used the same architecture and hyperparameters as those implemented in `disentanglement lib` of Locatello et al. (2019b), and the details are presented in Table 4 and 5.

D CONSISTENCY OF THE RESULTS WITH PROCRUSTES SIMILARITY

As mentioned in Section 3, in this section we provide a comparison between the CKA scores reported in the main paper, and the Procrustes scores for the cars3D dataset. We can see in Figures 9 to 11 that Procrustes and CKA provide similar results. Figures 9 and 10 show that Procrustes tends to overestimate the similarity between high-dimensional inputs, as mentioned in Section 2.3 (recall

Table 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 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

Table 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

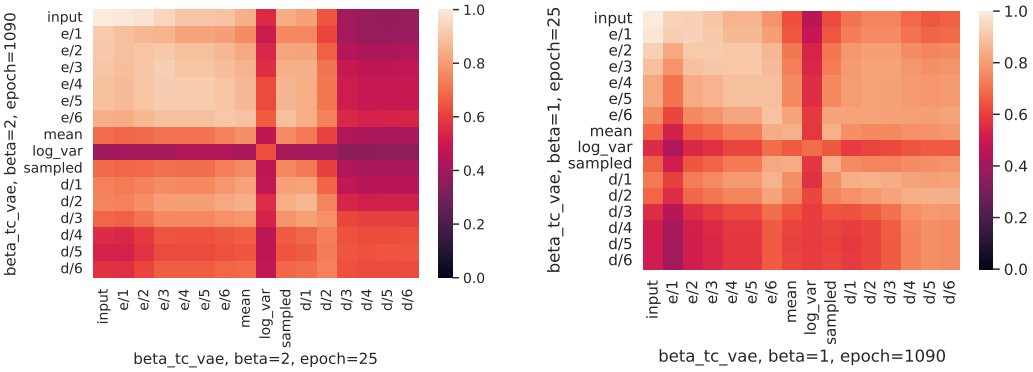
Table 4: Fully-connected architecture

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

the example given in Figure 1). In Figure 11, we observe a slightly lower similarity with Procrustes than CKA on the 5th and 6th layers of the encoder, indicating that some small changes in the representations may have been underestimated by CKA, as discussed in Section 2.3 and by Ding et al. (2021). Note that the difference between the CKA and Procrustes similarity scores in Figure 11 remains very small (around 0.1) indicating consistent results between both metrics.

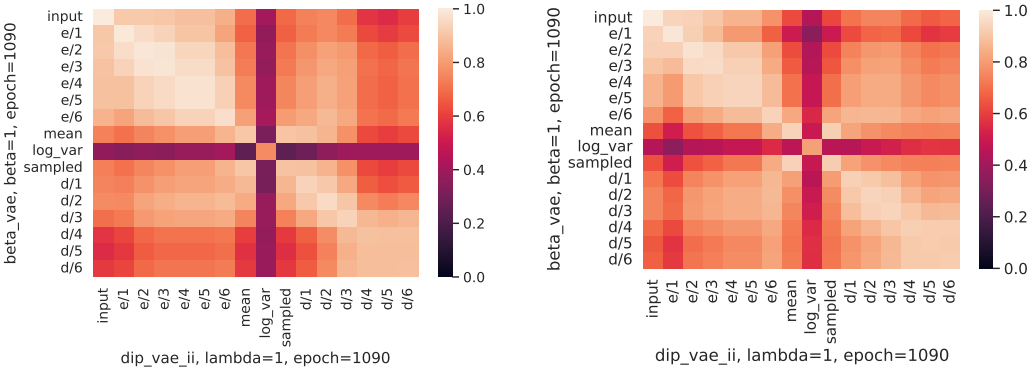
Table 5: Hyperparameters of fully-connected models

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



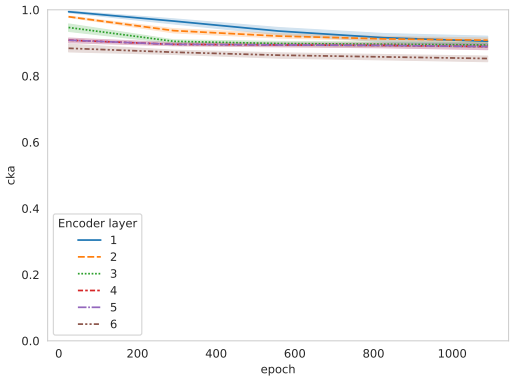
(a) CKA (b) Procrustes

Figure 9: As reported in Figure 3a of the main paper, (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 2.3.

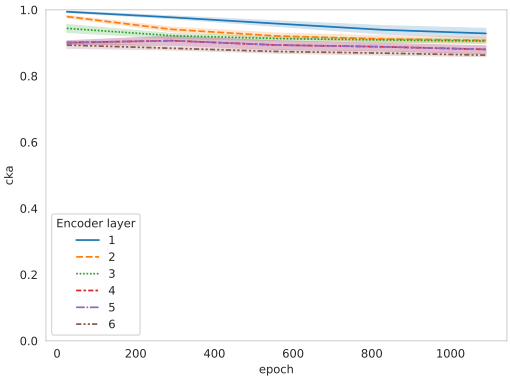


(a) CKA (b) Procrustes

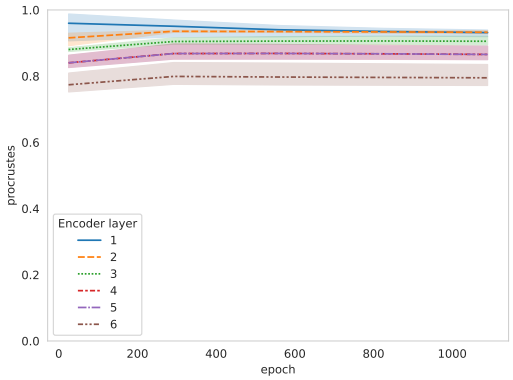
Figure 10: As in Figure 7a, (a) shows the CKA similarity scores of activations of β-VAE and DIP-VAE II trained on cars3D with $\beta = 1$, and $\lambda = 1$, respectively. (b) shows the Procrustes similarity scores using the same configuration. We observe the same trend with both metrics with Procrustes slightly overestimating the similarity between high dimensional activations (bottom-right quadrants) (c.f. Section 2.3).



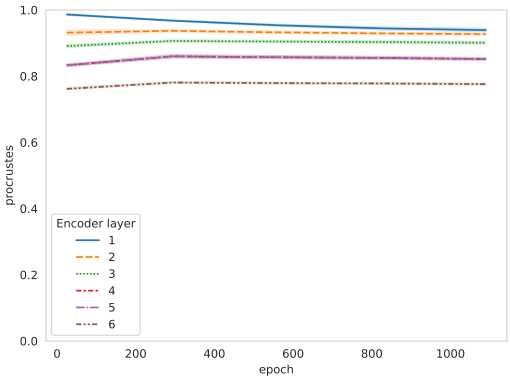
(a) CKA for $\beta = 1$



(b) CKA for $\beta = 8$



(c) Procrustes for $\beta = 1$



(d) Procrustes for $\beta = 8$

Figure 11: (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 $\beta = 8$. (a) and (b) are discussed in Figure 6 of the main paper. (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 2.3).

E RESOURCES

As mentioned in Sections 1 and 3, we released the code of our experiment, the pre-trained models and similarity scores:

- The similarity scores can be downloaded from an anonymous Google account using the following tiny URL `https://t.ly/0GLe3`
- The code can also be downloaded from an anonymous Google account using another tiny URL `https://t.ly/VMIm`
- Our pre-trained models are large (around 80 GB in total), and it was not feasible to make them available to the reviewers using an anonymous link. The URL to the models will, however, be available in the non-anonymised version of this paper.

F CKA ON FULLY-CONNECTED ARCHITECTURES

In order to assess the generalisability of our findings, we have repeated our observations on the fully-connected VAEs that are described in Appendix C. We can see in Figures 12, 13, and 14 that the same general trend as for the convolutional architectures can be identified (see Figures 3c, 4a, and 7 of Sections 4.1, 4.2 and 4.3 for a comparison with convolutional networks).

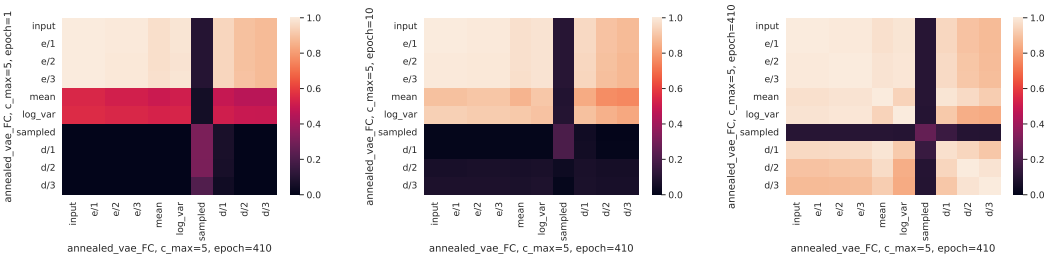
Learning in fully-connected VAEs is also bottom-up We can see in Figure 12 that, similarly to the convolutional architectures shown in Figure 3c, the encoder is learned early in the training process. Indeed between epochs 1 and 10, the encoder representations become highly similar to the representations of the fully trained model (see Figures 12a and 12b). The decoder is then learned with its representational similarity with the fully trained decoder raising after epoch 10 (see Figure 12c).

Impact of regularisation As in convolutional architectures shown in Figure 4a, the variance and sampled representations retain little similarity with the encoder representations in the case of posterior collapse, as shown in Figure 13. 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. Despite this difference, it can still be used to monitor posterior collapse with fully-connected architecture by relying on the similarity scores between the encoder representations, and the mean, variance and sampled representations. This property is consistent between both architectures.

Impact of learning objective Figure 14 provides results similar to the convolutional VAEs observed in Figure 7, 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., 2019b), one 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.

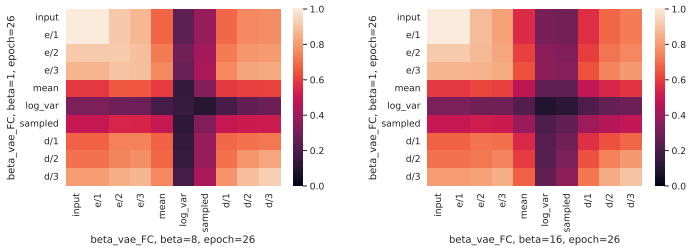
G HOW WELL DOES CKA DISTINGUISH POLARISED REGIME FROM POSTERIOR COLLAPSE?

In Section 4.2, we stated that CKA could be a useful tool to monitor posterior collapse. However, one can wonder whether CKA can lead to false positives when VAEs contain passive variables for non-pathological reasons. For example, due to the polarised regime, if one provides more latent variables than needed by the VAEs, some of the variables will be collapsed to reduce the KL divergence. As in posterior collapse, the decoder will ignore these passive variables. However, contrary to



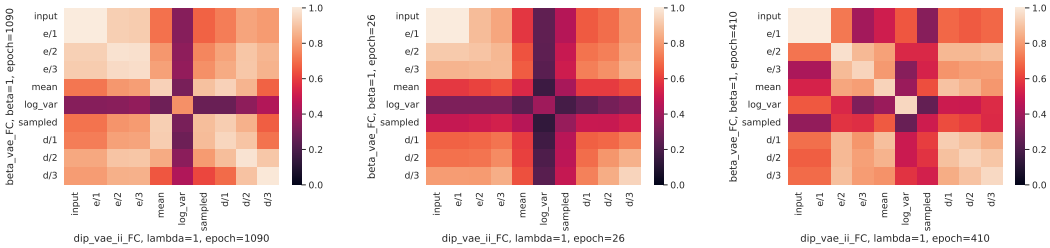
(a) Epoch 1 (b) Epoch 10 (c) Epoch 410

Figure 12: (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. Similarity to Figure 3c of Section 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 13: (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 4a.



(a) Trained on cars3D (b) Trained on dSprites (c) Trained on smallNorb

Figure 14: (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, 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.

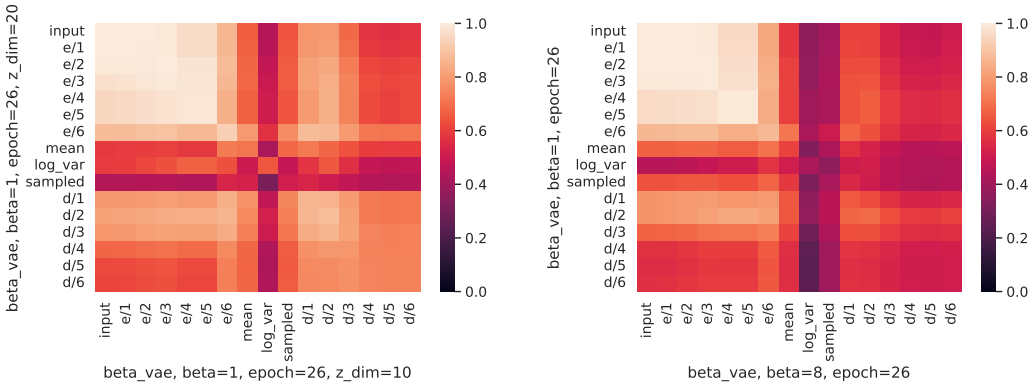
collapsed models, when passive variables are a result of the polarised regime, the decoder will still have access to meaningful information and will be able to correctly reconstruct the image, learning similar representations as a good model with fewer latent variables. We can see in Figure 15 that, in opposition to posterior collapse, the variance and sampled representations retain much higher similarity scores with the representations learned by other layers. Thus, one can differentiate between the two scenarios using CKA. Given that the CKA scores for the variance and sampled representations vary similarly in fully-connected architectures, CKA seems to consistently be a good predictor of posterior collapse across learning objectives and architectures while being robust to false positives. While one could be tempted to monitor posterior collapse using the changes of similarity scores in the decoder, we have seen in Appendix F that the fully-connected decoders could retain a relatively high similarity in the case of posterior collapse. Thus, we recommend relying on the CKA scores of the mean, variance and sampled representations for a better robustness across architectures.

H 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 16, 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 et al., 2021) and also suggests that the weights of the pre-trained architecture could be used as-is without further updates. While using pre-trained encoders may be beneficial in the context of transfer learning, domain adaptation (Pan & Yang, 2009), or simply reconstruction quality (Liu et al., 2021), one should not expect a dramatic improvement of the training time given that the encoder is learned very early during the training (see Section 4.1).

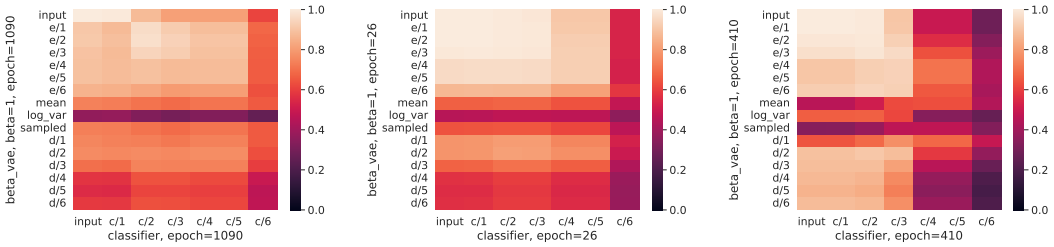
I Representational similarity of VAEs at different epochs

The results obtained in Section 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 3. 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.



(a) $\beta = 1$ (b) $\beta = 8$

Figure 15: (a) shows the CKA similarity scores between the activations of two β-VAEs trained on dSprites with $\beta = 1$, the first with 10 latent variables, and the second with 20. (b) shows the CKA similarity scores between the activations of two β-VAEs with 10 latent variables trained on dSprites with $\beta = 1$, and $\beta = 8$. For (a) and (b), the activations are taken after complete training, and all the results are averaged over 5 seeds. We can see that we retain a higher similarity in the decoder representations in the case of polarised regime (bottom-right quadrant in (a)) than posterior collapse (see the darker bottom-right quadrant in (b)). The representational similarity of the mean, variance and sampled representations is also higher in the case of polarised regime than posterior collapse.



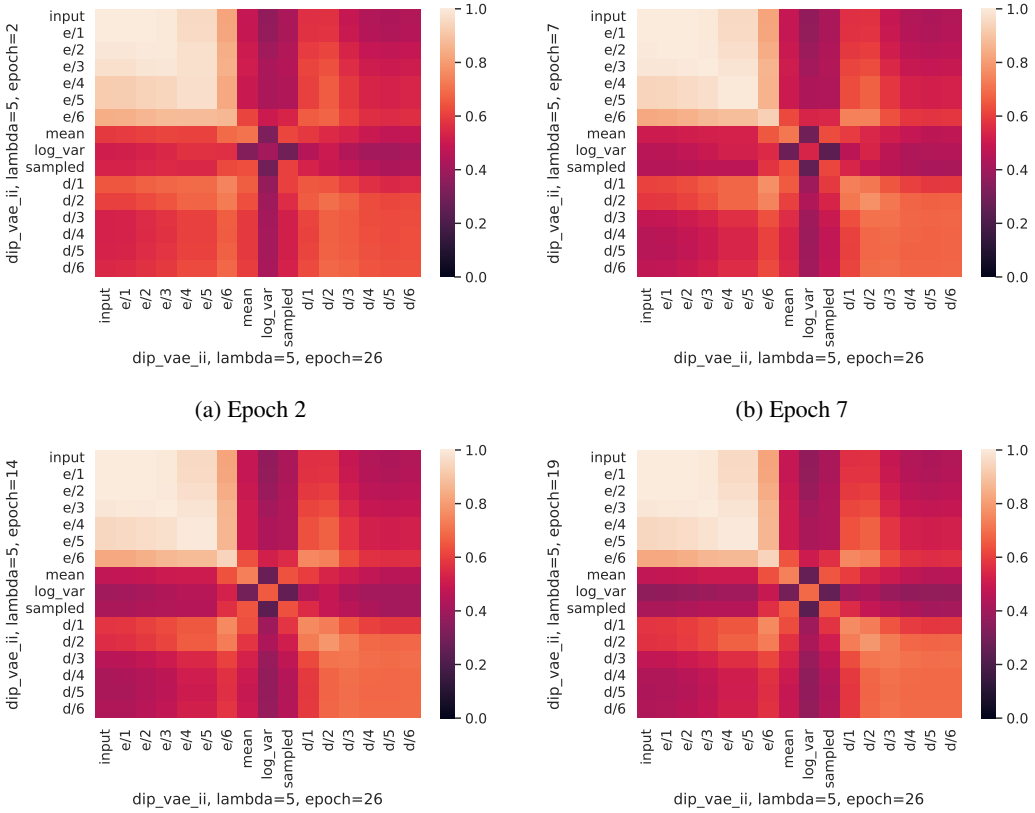
(a) Trained on cars3D (b) Trained on dSprites (c) Trained on smallNorb

Figure 16: (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, sampled representations (center diagonal values) are different from the representations learned by the classifier.

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 4 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 the main paper, and they do not depend on specific epochs.

Similarity changes over multiple epochs In Figures 17, 18, and 19, we can observe the same trend of learning phases as in Figure 3. First, the encoder is learned, as shown by the high representational similarity of the upper-left quadrant of Figures 17a, 18a, and 19a. Then, the decoder is learned, as shown by the increased representational similarity of the bottom-right quadrant of Figures 17b, 18b, and 19b. 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 17c, 18c, and 19c, and Figures 17d, 18d, and 19d.



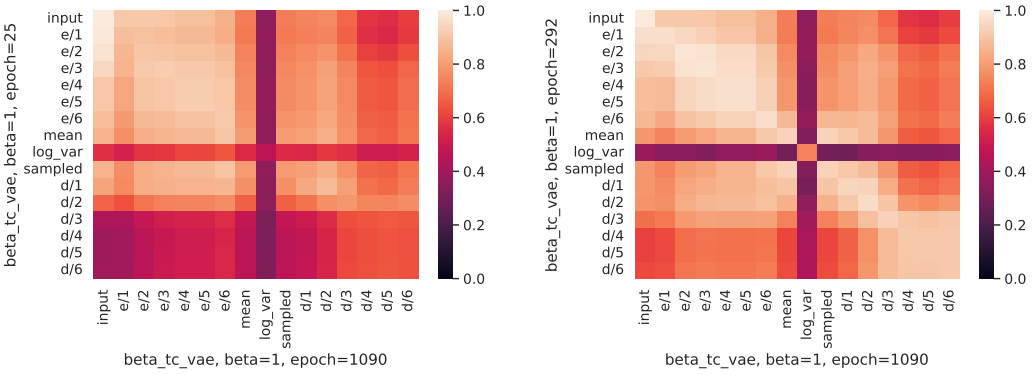
(a) Epoch 2

(b) Epoch 7

(c) Epoch 14

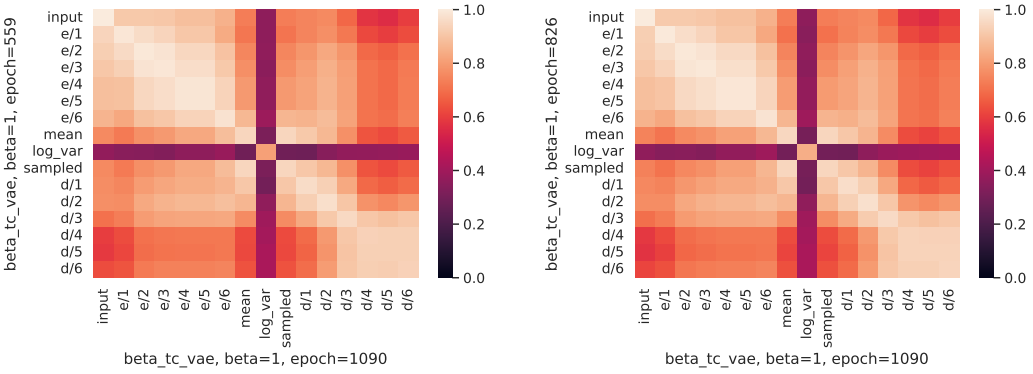
(d) Epoch 19

Figure 17: (a), (b), (c), and (d) show the representational similarity between DIP-VAE II after full training, and at epochs 2, 7, 14, and 19, respectively. All models are trained on dSprites and the results are averaged over 5 runs.



(a) Epoch 25

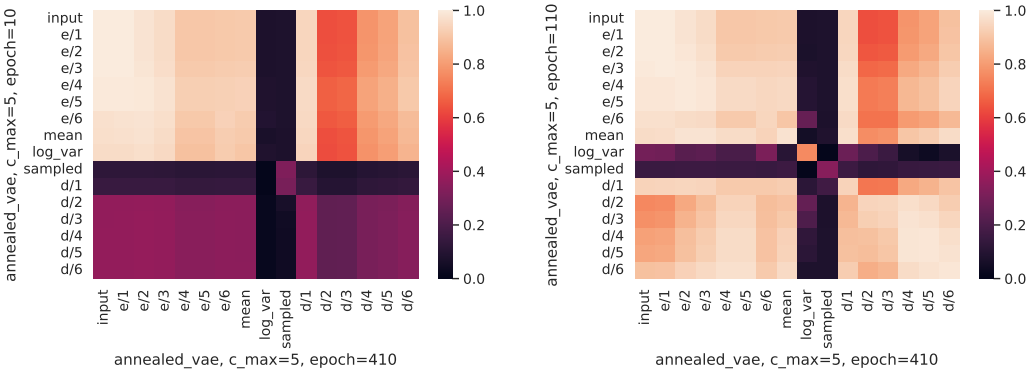
(b) Epoch 292



(c) Epoch 559

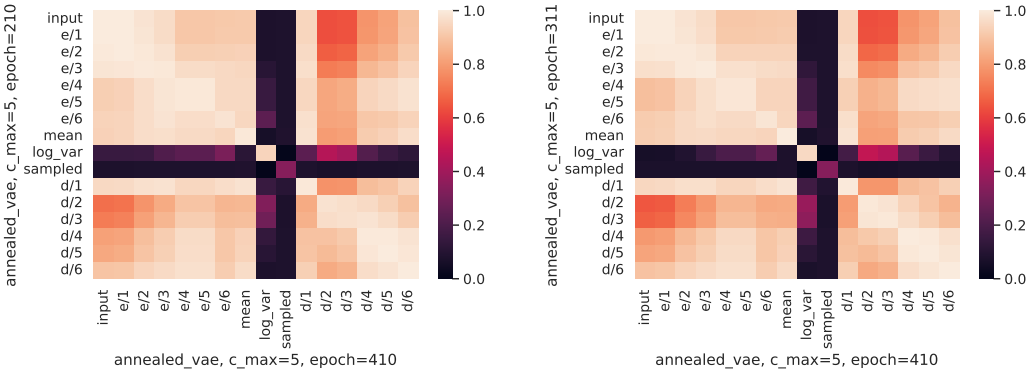
(d) Epoch 826

Figure 18: (a), (b), (c), and (d) show the representational similarity between β-TC VAE after full training, and at epochs 25, 292, 559, and 826, respectively. All models are trained on cars3D and the results are averaged over 5 runs.



(a) Epoch 10

(b) Epoch 110



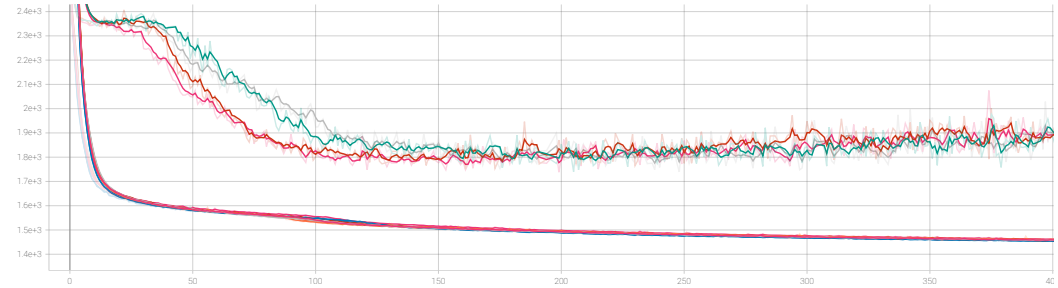
(c) Epoch 210

(d) Epoch 311

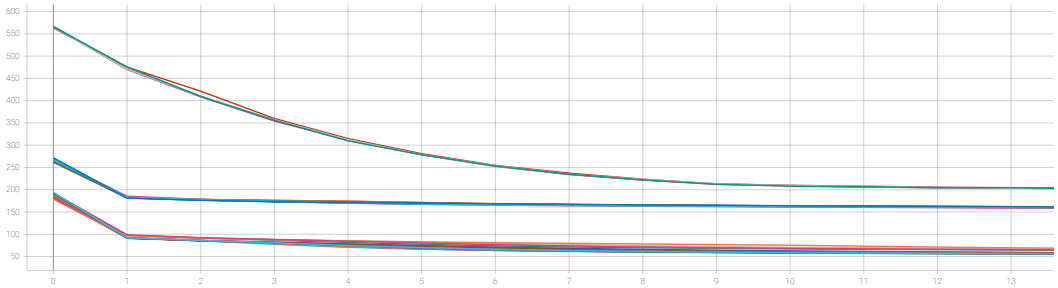
Figure 19: (a), (b), (c), and (d) show the representational similarity between Annealed VAE after full training, and at epochs 10, 110, 210, and 311, respectively. All models are trained on smallNorb and the results are averaged over 5 runs.

J CONVERGENCE RATE OF DIFFERENT VAES

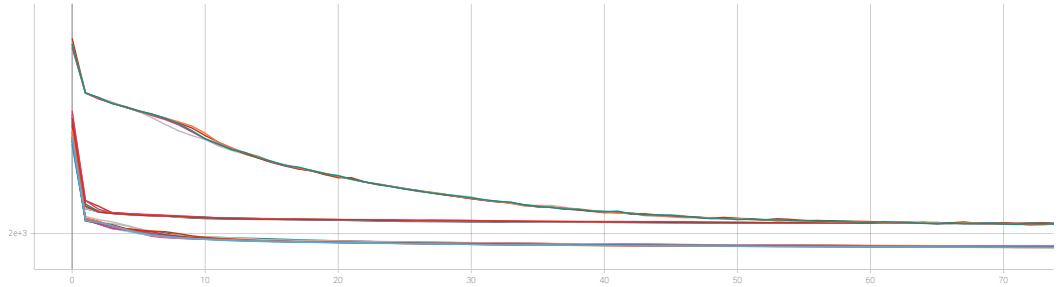
We can see in Figure 20 that all the models converge at the same epoch, with less regularised models reaching lower losses. While annealed VAEs start converging together with the other models, they then take longer to plateau, due to the annealing process. We can see them distinctly in the upper part of Figure 20. Overall, the epochs at which the models start to converge are consistent with our choice of epoch for early training in Section 4.1.



(a) Convergence on cars3D



(b) Convergence on dSprites



(c) Convergence on smallNorb

Figure 20: 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.