A Heuristic for Unsupervised Model Selection for Variational Disentangled Representation Learning

Sunny Duan[1], **Nick Watters**[1], **Loic Matthey**[1], **Chris Burgess**[1], **Alexander Lerchner**[1] **and Irina Higgins**[1]
[1]DeepMind

Disentangled representations have recently been shown to improve data efficiency, generalisation, robustness and interpretability in simple supervised and reinforcement learning tasks. To extend such results to more complex domains, it is important to address a major shortcoming of the current state of the art unsupervised disentangling approaches – high convergence variance, whereby different disentanglement quality may be achieved by the same model depending on its initial state. The existing model selection methods require access to the ground truth attribute labels, which are not available for most datasets. Hence, the benefits of disentangled representations have not yet been fully explored in practical applications. This paper addresses this problem by introducing a simple yet robust and reliable method for unsupervised disentangled model selection. We show that our approach performs comparably to the existing supervised alternatives across 5400 models from six state of the art unsupervised disentangled representation learning model classes.

Introduction

Happy families are all alike; every unhappy family is unhappy in its own way. —

Leo Tolstoy, Anna Karenina

Despite the success of deep learning in the recent years (Espeholt et al., 2018; Hessel et al., 2017; Hu et al., 2018; Lample et al., 2018; Oord et al., 2016; Silver et al., 2018), the majority of state of the art approaches are still missing many basic yet important properties common to biological intelligence, such as data efficient learning, strong generalisation beyond the training data distribution, or the ability to transfer knowledge between tasks (Garnelo et al., 2016; Lake et al., 2016; Marcus, 2018). The idea that a good representation can help with such shortcomings is not new. However, it appears that end-to-end learning often struggles to discover such a good representation automatically. If good representations should be explicitly encouraged, what should they look like?

Recently a number of papers have demonstrated that models with *disentangled* representations show improvements in terms of the aforementioned shortcomings (Achille et al., 2018; Higgins et al., 2017b, 2018b; Laversanne-Finot et al., 2018; Nair et al., 2018; Steenbrugge et al., 2018). While an agreed upon definition of a disentangled representation is still missing, a common intuitive description is that a disentangled representation should reflect the factorised structure of the world. For example, to describe an object we often use words pertaining to its colour, position, shape and size. We can use different words to describe these properties because they relate to independent factors of variation in our world. For example, specifying the colour of an object does not typically affect its position or size. Hence, a disentangled representation should reflect this by also factorising into individual dimensions that represent the colour, position, shape and size properties of objects (Bengio et al., 2013).

The ability to automatically discover the generative factors of complex real datasets can be of great importance in many practical applications of machine learning and data science. However, it is

Corresponding author(s): sunnyd@google.com, irinah@google.com

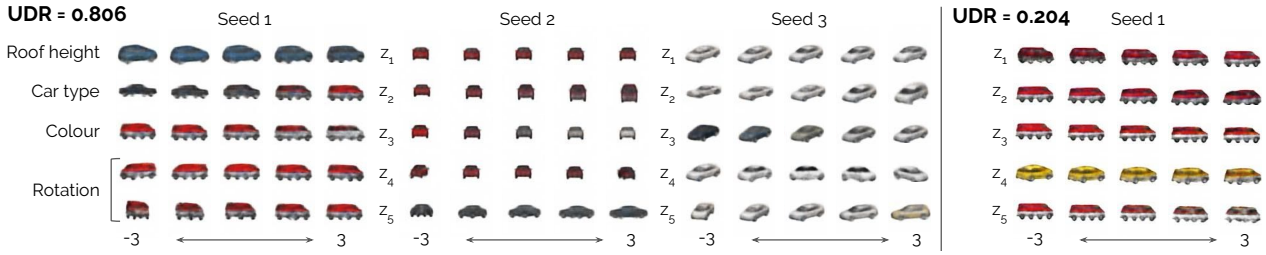

Figure 1 | Latent traversals for one of the best and worst ranked trained β-VAE models using the Unsupervised Disentanglement Ranking (UDR_L) method on the 3D Cars dataset. For each seed image we fix all latents z_i to the inferred value, then vary the value of one latent at a time to visualise its effect on the reconstructions. The high scoring model (left 3 blocks) appears well disentangled, since individual latents have consistent semantic meaning across seeds. The low scoring model (right block) is highly entangled, since the latent traversals are not easily interpretable.

important to be able to learn such representations in an unsupervised manner, since most interesting datasets do not have their generative factors fully labelled. For a long time scalable unsupervised disentangled representation learning was impossible, until recently a new class of models based on Variational Autoencoders (VAEs) (Kingma and Welling, 2014; Rezende et al., 2014) was developed. These approaches (Burgess et al., 2017; Chen et al., 2018; Higgins et al., 2017a; Kim and Mnih, 2018; Kumar et al., 2017) scale reasonably well and achieve current state of the art unsupervised disentangled representation learning. However, so far the benefits of these techniques have not been widely exploited because of a major shortcoming – these models suffer from high optimisation variance (Locatello et al., 2018). In particular, the quality of the achieved disentangling is sensitive to the choice of hyperparameters and even the model initialisation seed. Hence, in order to practically apply this class of unsupervised representation learning techniques, it is important to have a robust model selection process. This, however, is currently not possible without having access to the ground truth generative process and/or attribute labels (Chen et al., 2018; Eastwood and Williams, 2018; Higgins et al., 2017a; Kim and Mnih, 2018; Ridgeway and Mozer, 2018). Hence, the field finds itself in a predicament. From one point of view, there exists a set of approaches capable of reasonably scalable unsupervised disentangled representation learning. On the other hand, these models are hard to use in practice, because there is no easy way to do a hyperparameter search and model selection without access to the attribute labels.

This paper attempts to bridge this gap. We propose a simple yet effective heuristic for unsupervised model selection for the class of current state-of-the-art VAE-based unsupervised disentangled representation learning methods. Intuitively our approach leverages the fact that for a particular dataset, disentangled representations are all alike, while every entangled representation is entangled in its own way, to rephrase Tolstoy. Indeed, if we compare two models which learnt to disentangle the same dataset, we should expect their representations to be the same up to a permutation (the models learn the same data generative factors, but these are encoded by different individual latent dimensions), subsetting (one model learns a subset of the data generative factors that the other model learnt) and sign inversion (one model encodes object size as small-to-large, while the other encodes it as large-to-small). On the other hand, the representations learnt by two entangled models are likely to be quite different (Li et al., 2016; Morcos et al., 2018). Hence, our method relies on pair-wise comparisons of trained models obtained during a hyperparameter search.

We evaluate the validity of our unsupervised model selection metric against the four best existing supervised alternatives reported in the large scale study by Locatello et al. (2018): the β-VAE metric (Higgins et al., 2017a), the FactorVAE metric (Kim and Mnih, 2018), Mutual Information Gap (MIG) (Chen et al., 2018) and DCI Disentanglement scores (Eastwood and Williams, 2018). We do so for all existing state of the art disentangled representation learning approaches: β-VAE (Higgins et al., 2017a),

CCI-VAE (Burgess et al., 2017), FactorVAE (Kim and Mnih, 2018), TC-VAE (Chen et al., 2018) and two versions of DIP-VAE (Kumar et al., 2017). We validate our proposed method on two datasets with fully known generative processes commonly used to evaluate the quality of disentangled representations: dSprites (Matthey et al., 2017) and 3D Shapes (Burgess and Kim, 2018), and show that our unsupervised model selection method is able to match the supervised baselines in terms of guiding a hyperparameter search and picking the most disentangled trained models both quantitatively and qualitatively. We also apply our approach to the 3D Cars dataset (Reed et al., 2014), where the full set of ground truth attribute labels is not available, and confirm through visual inspection that the ranking produced by our method is meaningful (Fig. 1). Overall we evaluate 6 different model classes, with 6 separate hyperparameter settings and 50 seeds on 3 separate datasets, totalling 5400 models and show that our method is both accurate and consistent across models and datasets.

Operational definition of disentangling

Given a dataset of observations $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$, we assume that there exists a "true" generative process g that produces the observations from a small set of K independent generative factors according to $g : \boldsymbol{c}_n \mapsto \boldsymbol{x}_n$, where $p(\boldsymbol{c}_n) = \prod_{j=1}^{K} p(c_n^j)$. For simplicity we will assume that g is deterministic, however without loss of generality, g could also be stochastic. Disentangling is operationalised as the inverse of the generative process g, whereby we want to recover the latent representation $\boldsymbol{z} \in \mathbb{R}^L$ that best explains the observed data $p(\boldsymbol{z}, \boldsymbol{x}) \approx p(\boldsymbol{c}, \boldsymbol{x})$, and which factorises the same way as the data generative factors.

When talking about disentangled representations, three properties are generally considered: *modularity*, *compactness* and *explicitness*[1] (Ridgeway and Mozer, 2018). *Modularity* measures whether each latent dimension encodes only one data generative factor, *compactness* measures whether each data generative factor is encoded by a single latent dimension, and *explicitness* measures whether all the information about the data generative factors can be decoded from the latent representation.

We believe that *modularity* is the key aspect of disentangling, since it measures whether the representation is compositional, which gives disentangled representations the majority of their beneficial properties (see Sec. in Supplementary Materials for more details). *Compactness*, on the other hand, may not always be desirable. For example, rotation may be represented by a single latent unit $z_l \in \mathbb{R}^1$ encoding the rotation angle θ, however an alternative representation in $z_l \in \mathbb{R}^2$ with a $\sin(\theta)$ and $\cos(\theta)$ basis may be more faithful to the cyclic nature of the generative factor (see also Ridgeway and Mozer (2018) and Higgins et al. (2018b)). Finally, while *explicitness* is clearly desirable for preserving information about the data that may be useful for subsequent tasks, in practice models often fail to discover and represent the full set of the data generative factors. For example, the current state of the art approaches to unsupervised disentanglement often struggle to learn discrete data generative factors (e.g. the shape generative factor in dSprites dataset (Higgins et al., 2017a)). Hence, we suggest noting the explicitness of a representation, but not necessarily punishing its disentanglement ranking if it is not fully explicit. Instead, we suggest that the practitioner should have the choice to select the most disentangled model given a particular number of discovered generative factors. Hence, in the rest of the paper we will often use the terms disentanglement to refer to the compositional property of a representation.

A worked example A commonly used unit test for evaluating disentangling is the dSprites dataset (Matthey et al., 2017). This dataset consists of images of a single binary sprite pasted on a blank background and can be fully described by five generative factors: $C = \{$***sh***ape, *position* $\boldsymbol{x}$, *position* $\boldsymbol{y}$, ***s****ize*, ***r****otation*$\}$. The generative process for this dataset is fully deterministic, and hence g is a bijection between

[1] Similar properties have also been referred to as *disentanglement*, *completeness* and *informativeness* respectively in the independent yet concurrent paper by Eastwood and Williams (2018).

$\boldsymbol{c} \in \mathbb{R}^5$ and $\boldsymbol{x} \in \mathbb{R}^{64\times64}$. Hence, a fully disentangled representation of this dataset should be a latent space $\boldsymbol{z} \in \mathbb{R}^L$ that can be decomposed into five (in order to be fully explicit) independent subspaces: $\boldsymbol{z} = \boldsymbol{z}_{sh} \oplus \boldsymbol{z}_x \oplus \boldsymbol{z}_y \oplus \boldsymbol{z}_s \oplus \boldsymbol{z}_r$, where the subscripts point to the corresponding generative factors that the subspaces should learn to represent. Note that each such independent subspace should encode one and only one ground truth generative factor (to be fully modular), however its dimensionality may not match the dimensionality of the corresponding generative factor (lack of compactness is permissible).

Variational unsupervised disentangling

The current state of the art approaches to unsupervised disentangled representation learning are based on the Variational Autoencoder (VAE) framework (Kingma and Welling, 2014; Rezende et al., 2014). VAEs attempt to estimate the lower bound on the joint distribution of the data and the latent factors $p(\boldsymbol{x}, \boldsymbol{z})$ by optimising the following objective:

$$\mathcal{L}_{VAE} = \mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] - KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] \tag{1}$$

where, in the simplest case, the prior $p(\boldsymbol{z})$ is chosen to be an isotropic unit Gaussian. The objective in Eq. 1 does not in itself encourage disentangling, as discussed in Rolinek et al. (2018) and Locatello et al. (2018). Instead, it is the peculiarities of the particular VAE implementation choices that allow disentangling to emerge: the factorised prior and the pressure for the posterior covariance matrix to be diagonal. These implementation choices allow VAEs to be analysed from the rate-distortion theory perspective as optimising the trade-off between the capacity of an information bottleneck with independent sources of noise, and the quality of the resulting reconstruction (Burgess et al., 2017). This trade-off can be exploited in various ways to encourage disentangling by decomposing the objective in Eq. 1 into various terms and changing their relative weighting. In this paper we will consider six state of the art approaches to unsupervised disentangled representation learning that can be grouped into three broad classes based on how they modify the objective in Eq. 1: 1) β-VAE (Higgins et al., 2017a) and CCI-VAE (Burgess et al., 2017) upweight the KL term; 2) FactorVAE (Kim and Mnih, 2018) and TC-VAE (Chen et al., 2018) introduce a total correlation penalty; and 3) two different implementations of DIP-VAE (-I and -II) (Kumar et al., 2017) penalise the deviation of the the marginal posterior from a factorised prior. See Sec. in Supplementary Material for details.

Unsupervised disentangled model selection

We are interested in developing a method for unsupervised disentangled model selection which, at the minimum, works reliably well for the existing class of variational disentangled representation learning methods described in Sec. . In order to be practically useful, this method should have two properties. It should: 1) help with hyperparameter tuning, e.g. through evolutionary or Bayesian methods (Bergstra et al., 2011; Hutter et al., 2011; Jaderberg et al., 2018; Miikkulainen et al., 2017; Snoek et al., 2012; Thornton et al., 2012); 2) rank trained models based on their disentanglement quality. To this end, we develop the Unsupervised Disentanglement Ranking (UDR) method, which consists of four steps (illustrated in Fig. 5 in Supplementary Material):

1. Train $M = H \times S$ models, where H is the number of different hyperparameter settings, and S is the number of different initial model weight configurations (seeds).
2. For each trained model $i \in \{1, ..., M\}$, sample without replacement $P \leq S$ other trained models with the same hyperparameters but different seeds.

3. Perform P pairwise comparisons per trained model and calculate the respective UDR_{ij} scores, where $i \in \{1, ..., M\}$ is the model index, and $j \in \{1, ..., P\}$ is its unique pairwise match from Step 2.
4. Aggregate UDR_{ij} scores for each model i to report the final $\text{UDR}_i = \text{avg}_j(\text{UDR}_{ij})$ scores, where $\text{avg}_j(\cdot)$ is the median over P scores.

The key part of the UDR method is Step 3, where we calculate the UDR_{ij} score that summarises how similar the representations of the two models i and j are. The two respective latent representations $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ should be scored as highly similar if they axis align with each other. Given the optimisation variance characteristic of deep learning (Li et al., 2016; Morcos et al., 2018), individual latent dimensions $z_{i,a}$ and $z_{j,b}$ (where $a, b \in \{1, ..., L\}$ and $\boldsymbol{z}_i, \boldsymbol{z}_j \in \mathbb{R}^L$) are unlikely to be axis aligned by chance. However, such axis alignment should emerge if both $z_{i,a}$ and $z_{j,b}$ learn to represent the same ground truth generative factor c_k. Hence, we are looking for pairs of axis aligned latent dimensions with the following considerations in mind:

1. **Permutation** – the same ground truth factor c_k may be encoded by different latent dimensions within the two models, $z_{i,a}$ and $z_{j,b}$ where $a \neq b$.
2. **Sign inverse** – the two models may learn to encode the values of the generative factor in the opposite order compared to each other, $z_{i,a} = -z_{j,b}$.
3. **Subsetting** – one model may learn a subset of the ground truth factors that the other model has learnt.

In order for the UDR to be invariant to the first scenario, we propose calculating a full $L \times L$ similarity matrix R_{ij} between the individual dimensions of $\boldsymbol{z}_i \in \mathbb{R}^L$ and $\boldsymbol{z}_j \in \mathbb{R}^L$ (see Fig. 6 in Supplementary Material). In order to address the second point, we take the absolute value of the similarity matrix $|R_{ij}|$. Finally, to address the third point, we divide the UDR score by the average number of the ground truth factors discovered by the two models (in practice we do not have access to the ground truth factors, so we approximate this by counting the number of informative latents in each model).

To populate the similarity matrix R_{ij} we calculate each matrix element as the similarity between two vectors $\boldsymbol{z}_{i,a}$ and $\boldsymbol{z}_{j,b}$, where $\boldsymbol{z}_{i,a}$ is a response of a single latent dimension z_a of model i over the entire ordered dataset (see Sec. in Supplementary Material for details). We considered two versions of the UDR score based on the method used for calculating the vector similarity: the non-parametric UDR_S, using Spearman's correlation; and the parametric UDR_L, using Lasso regression following past work on evaluating representations (Eastwood and Williams, 2018; Li et al., 2016). In practice the Lasso regression version worked slightly better, so the remainder of the paper is restricted to UDR_L (we use UDR_L and UDR interchangeably to refer to this version), while UDR_S is discussed in the Supplementary Materials.

Score aggregation Given a similarity matrix R_{ij}, we want to find one-to-one correspondence between all the informative latent dimensions within the chosen pair of models. Hence, we want to see at most a single strong correlation in each row and column of the similarity matrix. To that accord, we step through the matrix $R = |R_{ij}|$, one column and row at a time, looking for the strongest correlation, and weighting it by the proportional weight it has within its respective column or row. We then average all such weighted scores over all the informative row and column latents to calculate the final UDR_{ij} score:

$$\frac{1}{d_a + d_b}\left[\sum_b \frac{r_a^2 * I_{KL}(b)}{\sum_a R(a,b)} + \sum_a \frac{r_b^2 * I_{KL}(a)}{\sum_b R(a,b)}\right] \tag{2}$$

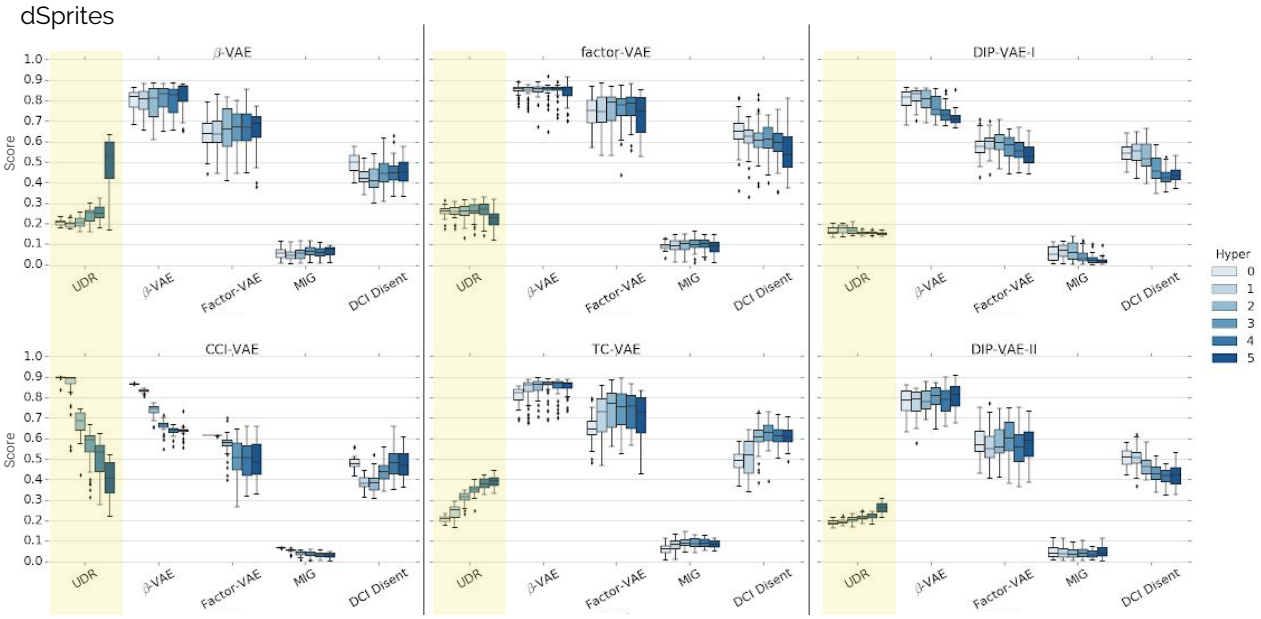

Figure 2 | Hyperparameter search results for six unsupervised disentangling model classes on the dSprites dataset. All models are evaluated using the unsupervised UDR and the supervised β-VAE, FactorVAE, MIG and DCI Disentangling metrics and trained on dSprites dataset.

where $r_a = \max_a R(a, b)$ and $r_b = \max_b R(a, b)$. I_{KL} indicates an "informative" latent within a model and d is the number of such latents: $d_a = \sum_a I_{KL}(a)$ and $d_b = \sum_b I_{KL}(b)$. We define a latent dimension as "informative" if it has learnt a latent posterior which diverges from the prior:

$$I_{KL}(a) = \begin{cases} 1 & KL(q_\phi(z_a) \,||\, p(z_a)) > 0.01 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

UDR variations We explored whether doing all-to-all pairwise comparisons, with models in Step 2 sampled from the set of all M models rather than the subset of S models with the same hyperparameters, would produce more accurate results. Additionally we investigated the effect of choosing different numbers of models P for pairwise comparisons by sampling $P \sim \mathrm{U}[5, 45]$.

UDR assumptions and limitations Note that our approach is based on a number of assumptions and has certain limitations discussed below:

1. **High optimisation variance** – we assume that two representations of the same dataset are unlikely to be axis aligned, unless they are disentangled. This is currently true for deep learning approaches (Li et al., 2016; Morcos et al., 2018), but may not hold in the future. Hence, UDR should be applied to non-VAE representation learning approaches with caution. For example, it is likely to be appropriate for InfoGAN (Chen et al., 2016) or traditional autoencoders (Vincent et al., 2010), but it would give high false positive scores to non-parametric approaches, like PCA, which would consistently produce the same representation for a particular dataset.
   We suggest using a quantitative measure of the dissociation between seed variance and the effect of hyperparameters on disentanglement quality to evaluate the suitability of UDR for a particular model class. The measure, originally described in Locatello et al. (2018), involves training models with different hyperparameters and seeds on a labelled disentanglement dataset (e.g. dSprites).

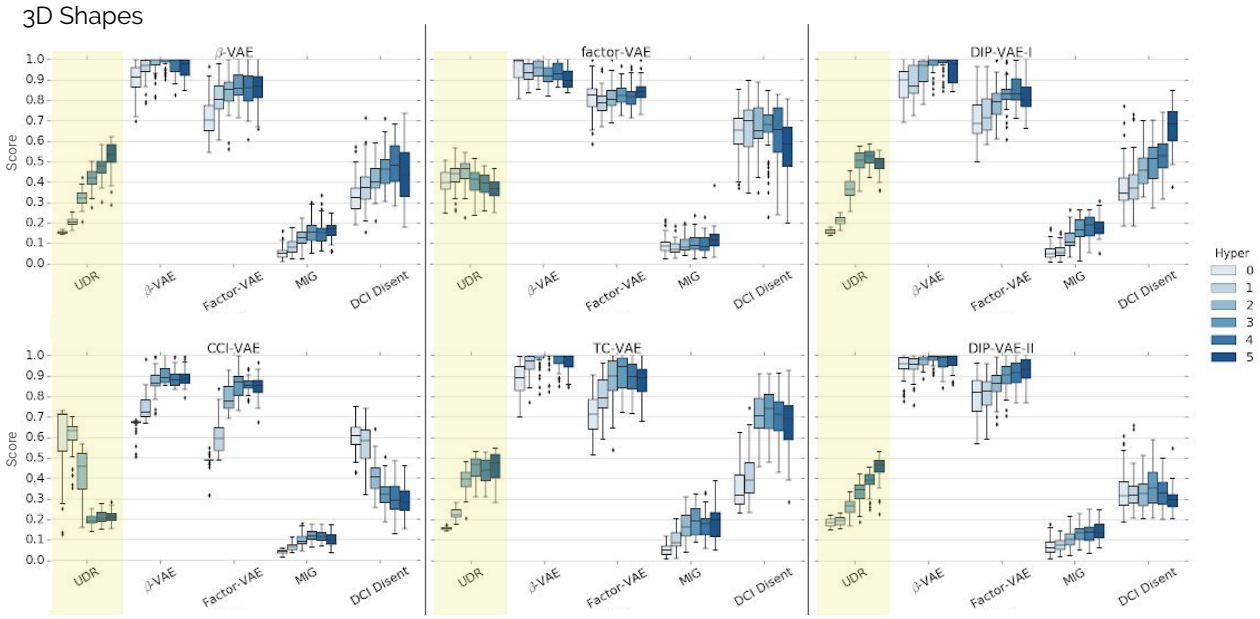

Figure 3 | Hyperparameter search results for six unsupervised disentangling model classes on the dSprites dataset. All models are evaluated using the unsupervised UDR and the supervised β-VAE, FactorVAE, MIG and DCI Disentangling metrics and trained on 3D Shapes dataset.

A supervised disentanglement metric is then used to choose the most disentangled model within a particular seed value, and measuring how likely this model is to perform at least as good as a randomly sampled model from the full hyperparameter search in terms of disentangling (see Sec. 5.4 in Locatello et al. (2018)). A method like PCA would score 100% and would be unsuitable for UDR. The state of the art disentangling VAEs score 80.7%, and we show in this paper that UDR works well for them.

2. **Biased disentanglement** – related to the point above, we assume that when two seeds of the same model converge to a disentangled representation, these representations are axis aligned up to a permutation, sign inverse and subsetting. This is true for the current state of the art variational unsupervised disentangling approaches described in Sec. , but may not hold more generally.
3. **Continuous, monotonic and scalar factors** – UDR assumes that these properties hold for the data generative factors and their representations. This is true for the disentangling approaches described in Sec. , but may not hold more generally. It is likely that UDR can be adapted to work with other kinds of generative factors (e.g. factors with special or no geometry) by exchanging the similarity calculations in Step 3 with an appropriate measure, however we leave this for future work.
4. **Herd effect** – since UDR detects disentangled representations through pairwise comparisons, the score it assigns to each individual model will depend on the nature of the other models involved in these comparisons. This means that UDR is unable to detect a single disentangled model within a hyperparameter sweep. It also means that when models are only compared within a single hyperparameter setting, individual model scores may be over/under estimated as they tend to be drawn towards the mean of the scores of the other models within a hyperparameter group. Thus, it is preferable to perform the UDR-A2A during model selection and UDR during hyperparameter selection.
5. **Explicitness bias** – UDR does not penalise models that learn a subset of the data generative factors. In fact, such models often score higher than those that learn the full set of generative factors, because the current state of the art disentangling approaches tend to trade-off the number of discovered factors for cleaner disentangling. As discussed in Sec. , we provide the practitioner with the ability to

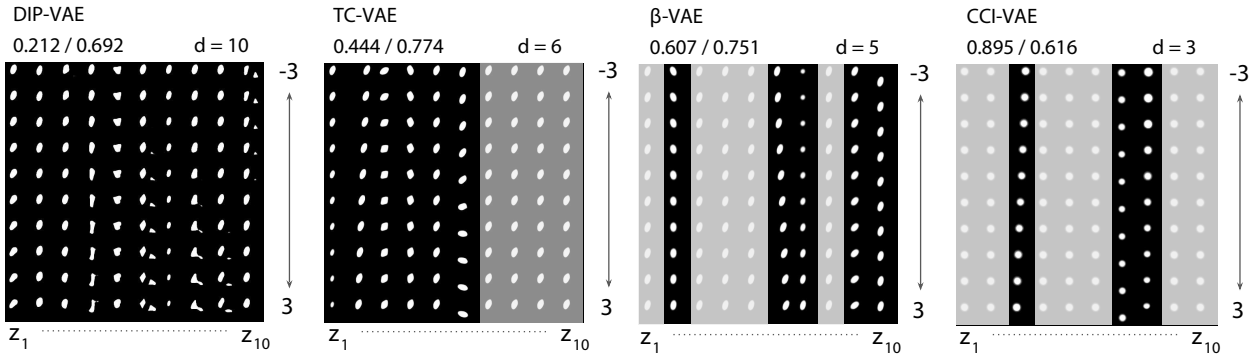

Figure 4 | Latent traversals of the top ranked trained DIP-VAE-I, TC-VAE, CCI-VAE and β-VAE according to the UDR method. At the top of each plot the two presented scores are UDR/FactorVAE metric. d is the number of informative latents. The uninformative latents are greyed out.

choose the most disentangled model per number of factors discovered by approximating this with the d score in Eq. 2.

6. **Computational cost** – UDR requires training a number of seeds per hyperparameter setting and $M \times P$ pairwise comparisons per hyperparameter search, which may be computationally expensive. Saying this, training multiple seeds per hyperparameter setting is a good research practice to produce more robust results. Furthermore, these computations are highly parallelisable, which at least makes our approach scalable.

To summarise, UDR relies on a number of assumptions and has certain limitations that we hope to relax in future work. Therefore, we do not claim that our proposed method is general or even principled. However, Sec. will empirically validate that UDR provides accurate and consistent rankings for 5400 models, all six state of the art unsupervised disentangled learning approaches and across three diverse datasets. Hence, we believe that UDR can be a useful method for unlocking the power of unsupervised disentangled representation learning to real-life practical applications, at least in the near future.

Experiments

We use the trained model checkpoints and supervised scores from Locatello et al. (2018) to evaluate β-VAE, CCI-VAE, FactorVAE, TC-VAE, DIP-VAE-I and DIP-VAE-II on two benchmark datasets: dSprites (Matthey et al., 2017) and 3D Shapes (Burgess and Kim, 2018) (see Sec. for details). Each model is trained with $H = 6$ different hyperparameter settings (detailed in Sec. in Supplementary Material), with $S = 50$ seeds per setting.

UDR correlates well with the supervised metrics. To validate UDR, we calculate Spearman's correlation between its model ranking and that produced by four existing supervised disentanglement metrics found to be the most meaningful in the large scale comparison study by Locatello et al. (2018): the original β-VAE metric (Higgins et al., 2017a), FactorVAE metric (Kim and Mnih, 2018), Mutual Information Gap (MIG) (Chen et al., 2018) and DCI Disentanglement (Eastwood and Williams, 2018) (see Sec. in Supplementary Material for metric details). The average correlation for UDR is 0.54 ± 0.06 and for UDR-A2A is 0.60 ± 0.11. This is comparable to the average Spearman's correlation between the model rankings produced by the different supervised metrics: 0.67 ± 0.2. The variance in rankings produced by the different metrics is explained by the fact that the metrics capture different aspects of disentangling in terms of the modularity, compactness and explicitness (see Tbl. 5 in Supplementary Materials). UDR is most similar to DCI Disentanglement, and hence correlates with it the most. See

Sec. in Supplementary Materials for a discussion of how UDR relates to other representation comparison methods.

UDR is useful for hyperparameter selection. Figs. 2-3 compares the scores produced by UDR and the four supervised metrics for 3600 trained models, split over six model classes, two datasets and six hyperparameter settings. We consider the median score profiles across the six hyperparameter settings to evaluate whether a particular setting is better than others. It can be seen that UDR broadly agrees with the supervised metrics on which hyperparameters are more promising for disentangling. This holds across datasets and model classes. Hence, UDR may be useful for evaluating model fitness for disentangled representation learning as part of an evolutionary algorithm or Bayesian hyperparameter tuning (Bergstra et al., 2011; Hutter et al., 2011; Jaderberg et al., 2018; Miikkulainen et al., 2017; Snoek et al., 2012; Thornton et al., 2012).

UDR is useful for model selection. Figs. 2-3 can also be used to examine whether a particular trained model has learnt a good disentangled representation. We see that some models reach high UDR scores. For example, more models score highly as the value of the β hyperparameter is increased in the β-VAE model class. This is in line with the previously reported results (Higgins et al., 2017a). Note that the 0th hyperparameter setting in this case corresponds to $\beta = 1$, which is equivalent to the standard VAE objective (Kingma and Welling, 2014; Rezende et al., 2014). As expected, these models score low in terms of disentangling.

We also see that for some model classes (e.g. DIP-VAE-I, DIP-VAE-II and FactorVAE on dSprites) no trained model scores highly according to UDR. This suggests that none of the hyperparameter choices explored were good for this particular dataset, and that no instance of the model class learnt to disentangle well. To test this, we use latent traversals to qualitatively evaluate the level of disentanglement achieved by the models, ranked by their UDR scores. This is a common technique to qualitatively evaluate the level of disentanglement on simple visual datasets where no ground truth attribute labels are available. Such traversals involve changing the value of one latent dimension at a time and evaluating its effect on the resulting reconstructions to understand whether the latent has learnt to represent anything semantically meaningful. Fig. 4 demonstrates that the qualitative disentanglement quality is reflected well in the UDR scores. The figure also highlights that the supervised metric scores can sometimes be overoptimistic. For example, compare TC-VAE and β-VAE traversals in Fig. 4. These are scored similarly by the supervised metric (0.774 and 0.751) but differently by UDR (0.444 and 0.607). Qualitative evaluation of the traversals clearly shows that β-VAE learnt a more disentangled representation than TC-VAE, which is captured by UDR but not by the supervised metric. Fig. 10 in Supplementary Material provides more examples.

UDR works well even with five pairwise comparisons. We test the effect of the number of pairwise comparisons P on the variance and accuracy of the UDR scores. Tbl. 1 reports the changes in the rank correlation with the β-VAE metric on the dSprites dataset as P is varied between 5 and 45. We see that the correlation between the UDR and the β-VAE metric becomes higher and the variance decreases as the number of seeds is increased. However, even with $P = 5$ the correlation is reasonable.

UDR generalises to a dataset with no attribute labels. We check whether UDR can be useful for selecting well disentangled models trained on the 3D Cars (Reed et al., 2014) dataset with poorly labelled attributes, which makes it a bad fit for supervised disentanglement metrics. Fig. 1 shows that a highly ranked model according to UDR appears disentangled, while a poorly ranked one appears entangled.

Sample # (P)	5	10	15	20	25	30	35	40	45
Correlation	0.51 ± 0.07	0.57 ± 0.03	0.57 ± 0.05	0.6 ± 0.03	0.59 ± 0.03	0.61 ± 0.02	0.61 ± 0.02	0.61 ± 0.01	0.61 ± 0.01

Table 1 | Rank correlations of the UDR score with the β-VAE metric on the dSprites dataset for a β-VAE hyperparameter search as the number of pairwise comparisons P per model were changed.

Fig. 11 in Supplementary Material provides more examples of high and low scoring models according to the UDR method.

Conclusion

We have introduced UDR, the first empirically validated heuristic for unsupervised model selection for variational disentangled representation learning. We have validated our approach on 5400 models covering all six state of the art unsupervised disentangled representation learning model classes. We compared UDR to four existing supervised disentanglement metrics both quantitatively and qualitatively, and demonstrated that our approach works reliably well across three different datasets. This is an important missing step towards unlocking the power of unsupervised disentangled representation learning to real-life applications. We appreciate that our approach relies on a number of assumptions and has several limitations, however we hope to address these in future work. In the short term, we have empirically demonstrated that UDR can be a useful tool for helping apply the current state of the art unsupervised disentangled representation learning methods to domains where no supervised attribute labels exist. In the long term, we hope that this work can be a useful starting point on the way to finding a more principled unsupervised disentanglement metric.

Acknowledgements

We thank Olivier Bachem and Francesco Locatello for helping us re-use their code and model checkpoints, and Neil Rabinowitz for useful feedback.

References

A. Achille, T. Eccles, L. Matthey, C. P. Burgess, N. Watters, A. Lerchner, and I. Higgins. Life-long disentangled representation learning with cross-domain latent homologies. *NIPS*, 2018.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

J. Bergstra, R. Bardenet, Y. Bengio, and B. Kegl. Algorithms for hyper-parameter optimization. *NIPS*, 2011.

C. Burgess and H. Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-VAE. *NIPS Workshop of Learning Disentangled Features*, 2017.

T. Q. Chen, X. Li, R. Grosse, and D. Duvenaud. Isolating sources of disentanglement in variational autoencoders. *NIPS*, 2018.

X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv*, 2016.

T. Cohen and M. Welling. Group equivariant convolutional networks. *ICML*, 2016.

C. Eastwood and C. K. I. Williams. A framework for the quantitative evaluation of disentangled representations. *ICLR*, 2018.

L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu. IMPALA: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arxiv*, 2018.

M. Garnelo, K. Arulkumaran, and M. Shanahan. Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518*, 2016.

R. Gens and P. M. Domingos. Deep symmetry networks. *NIPS*, 2014.

D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis; an overview with application to learning methods. *Neural Computation*, 16(12):2639 – 2664, 2004.

M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. *arxiv*, 2017.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2017a.

I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner. DARLA: Improving zero-shot transfer in reinforcement learning. *ICML*, 2017b.

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

I. Higgins, N. Sonnerat, L. Matthey, A. Pal, C. Burgess, M. Bosnjak, M. Shanahan, M. Botvinick, D. Hassabis, and A. Lerchner. SCAN: Learning hierarchical compositional visual concepts. *ICLR*, 2018b.

J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. *CVPR*, 2018.

F. Hutter, H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. *Learning and Intelligent Optimization*, 2011.

M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, C. Fernando, and K. Kavukcuoglu. Population based training of neural networks. *arXiv*, 2018.

H. Kim and A. Mnih. Disentangling by factorising. *ICLR*, 2018.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *ICLR*, 2014.

N. Kriegeskorte, M. Mur, and P. Bandettini. Representational similarity analysis – connecting the branches of systems neuroscience. *Front Syst Neurosci.*, 4(2), 2008.

A. Kumar, P. Sattigeri, and A. Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *arxiv*, 2017.

B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016.

G. Lample, M. Ott, A. Conneau, L. Denoyer, and M. Ranzato. Phrase-based & neural unsupervised machine translation. *arxiv*, 2018.

A. Laversanne-Finot, A. Péré, and P.-Y. Oudeyer. Curiosity driven exploration of learned disentangled goal spaces. *arxiv*, 2018.

Y. Li, J. Yosinski, J. Clune, H. Lipson, and J. Hopcroft. Convergent learning: Do different neural networks learn the same representations? *ICLR*, 2016.

F. Locatello, S. Bauer, M. Lucic, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *arXiv preprint arXiv:1811.12359*, 2018.

G. Marcus. Deep learning: A critical appraisal. *arxiv*, 2018.

L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner. dsprites: Disentanglement testing sprites dataset, 2017. URL https://github.com/deepmind/dsprites-dataset/.

R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, and B. Hodjat. Evolving Deep Neural Networks. *arxiv*, 2017.

A. S. Morcos, M. Raghu, and S. Bengio. Insights on representational similarity in neural networks with canonical correlation. *NIPS*, 2018.

A. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual reinforcement learning with imagined goals. *arxiv*, 2018.

A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *NIPS*, 2017.

S. Reed, K. Sohn, Y. Zhang, and H. Lee. Learning to disentangle factors of variation with manifold interaction. *ICML*, 2014.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, 32(2):1278–1286, 2014.

K. Ridgeway and M. C. Mozer. Learning deep disentangled embeddings with the f-statistic loss. *NIPS*, 2018.

M. Rolinek, D. Zietlow, and G. Martius. Variational autoencoders pursue pca directions (by accident). *arxiv*, 2018.

J. Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6): 863–869, 1992.

D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404.

J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. *arXiv*, 2012.

S. Soatto. Steps toward a theory of visual information. *Technical Report UCLA-CSD100028*, 2010.

X. Steenbrugge, S. Leroux, T. Verbelen, and B. Dhoedt. Improving generalization for abstract reasoning tasks using disentangled feature representations. *arxiv*, 2018.

R. Suter, D. Miladinovic, S. Bauer, and B. Scholkopf. Interventional robustness of deep latent variable models. *arxiv*, 2018.

C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms. *arXiv*, 2012.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *NIPS*, 2010.

Supplementary Material

Useful properties of disentangled representations

Disentangled representations are particularly useful because they re-represent the information contained in the data in a way that enables semantically meaningful *compositionality*. For example, having discovered that the data is generated using two factors, colour and shape, such a model would be able to support meaningful reasoning about fictitious objects, like pink elephants, despite having never seen one during training (Higgins et al., 2017b, 2018b). This opens up opportunities for counterfactual reasoning, more robust and interpretable inference and model-based planning (Higgins et al., 2018a; Suter et al., 2018). Furthermore, such a representation would support more data efficient learning for subsequent tasks, like a classification objective for differentiating elephants from cats. This could be achieved by ignoring the nuisance variables irrelevant for the task, e.g. the colour variations, by simply masking out the disentangled subspaces that learnt to represent such nuisances, while only paying attention to the task-relevant subspaces, e.g. the units that learnt to represent shape (Cohen and Welling, 2016; Gens and Domingos, 2014; Soatto, 2010). Hence, the semantically meaningful compositional nature of disentangled representations is perhaps the most sought after aspect of disentangling, due to its strong implications for generalisation, data efficiency and interpretability (Bengio et al., 2013; Higgins et al., 2018a; Schmidhuber, 1992).

Dataset details

dSprites A commonly used unit test for evaluating disentangling is the dSprites dataset (Matthey et al., 2017). This dataset consists of images of a single binary sprite pasted on a blank background and can be fully described by five generative factors: shape (3 values), position x (32 values), position y (32 values), size (6 values) and rotation (40 values). All the generative factors are sampled from a uniform distribution. Rotation is sampled from the full 360 degree range. The generative process for this dataset is fully deterministic, resulting in 737,280 total images produced from the Cartesian product of the generative factors.

3D Shapes A more complex dataset for evaluating disentangling is the 3D Shapes dataset (Burgess and Kim, 2018). This dataset consists of images of a single 3D object in a room and is fully specified by six generative factors: floor colour (10 values), wall colour (10 values), object colour (10 values), size (8 values), shape (4 values) and rotation (15 values). All the generative factors are sampled from a uniform distribution. Colours are sampled from the circular hue space. Rotation is sampled from the [-30, 30] degree angle range.

3D Cars This dataset was adapted from Reed et al. (2014). The full data generative process for this dataset is unknown. The labelled factors include 199 car models and 24 rotations sampled from the full 360 degree out of plane rotation range. An example of an unlabelled generative factor is the colour of the car – this varies across the dataset.

Unsupervised disentangled representation learning models

As mentioned in Sec. , current state of the art approaches to unsupervised disentangled representation learning are based on the VAE (Kingma and Welling, 2014; Rezende et al., 2014) objective presented in Eq. 1. These approaches decompose the objective in Eq. 1 into various terms and change their relative weighting to exploit the trade-off between the capacity of the latent information bottleneck with independent sources of noise, and the quality of the resulting reconstruction in order to learn a

disentangled representation. The first such modification was introduced by Higgins et al. (2017a) in their β-VAE framework:

$$\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \beta\ KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] \tag{4}$$

In order to achieve disentangling in β-VAE, the KL term in Eq. 4 is typically up-weighted by setting $\beta > 1$. This implicitly reduces the latent bottleneck capacity and, through the interaction with the reconstruction term, encourages the generative factors c_k with different reconstruction profiles to be encoded by different independent noisy channels z_l in the latent bottleneck. Building on the β-VAE ideas, CCI-VAE (Burgess et al., 2017) suggested slowly increasing the bottleneck capacity during training, thus improving the final disentanglement and reconstruction quality:

$$\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \gamma\ |KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z})) - C|\ ] \tag{5}$$

Later approaches (Chen et al., 2018; Kim and Mnih, 2018; Kumar et al., 2017) showed that the KL term in Eq. 1 can be further decomposed according to:

$$\mathbb{E}_{p(\boldsymbol{x})}[\ KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] = I(\boldsymbol{x};\boldsymbol{z}) + KL(q_\phi(\boldsymbol{z})\ ||\ p(\boldsymbol{z})) \tag{6}$$

Hence, penalising the full KL term as in Eqs. 4-5 is not optimal, since it unnecessarily penalises the mutual information between the latents and the data. To remove this undesirable side effect, different authors suggested instead adding more targeted penalised terms to the VAE objective function. These include different implementations of the total correlation penalty (FactorVAE by Kim and Mnih (2018) and TC-VAE by Chen et al. (2018)):

$$\mathcal{L}_{VAE} - \gamma\ KL(q_\phi(\boldsymbol{z})\ ||\ \prod_{j=1}^{M} q_\phi(z_j)) \tag{7}$$

and different implementations of the penalty that pushes the marginal posterior towards a factorised prior (DIP-VAE by Kumar et al. (2017)):

$$\mathcal{L}_{VAE} - \gamma\ KL(q_\phi(\boldsymbol{z})\ ||\ p(\boldsymbol{z})) \tag{8}$$

Related work

Methods for evaluating and comparing representations have been proposed in the past. The most similar approaches to ours are the DCI Disentanglement score from Eastwood and Williams (2018) and the axis alignment comparison of representations in trained classifiers proposed in Li et al. (2016). The former is not directly applicable for unsupervised disentangled model selection, since it requires access to the ground truth attribute labels. Even when adapted to compare two latent representations, our preliminary experiments suggested that the entropy based aggregation proposed in Eastwood and Williams (2018) is inferior to our aggregation in Eq. 2 when used in the UDR setup. The approach by Li et al. (2016) shares the similarity matrix calculation step with us, however they never go beyond that quantitatively, opting for qualitative evaluations of model representations instead. Hence, their approach is not directly applicable to quantitative unsupervised disentangled model ranking.

Other related approaches worth mentioning are the Canonical Correlation Analysis (CCA) and its modifications (Hardoon et al., 2004; Morcos et al., 2018; Raghu et al., 2017). These approaches, however, tend to be invariant to invertible affine transformations and therefore to the axis alignment of individual neurons, which makes them unsuitable for evaluating disentangling quality. Finally, Representation Similarity Matrix (RSM) (Kriegeskorte et al., 2008) is a commonly used method in Neuroscience for comparing the representations of different systems to the same set of stimuli. This technique, however, is not applicable for measuring disentangling, because it ignores dimension-wise response properties.

Table 2 | Encoder and Decoder Implementation details shared for all models

Encoder	Decoder
Input: $64 \times 64 \times$ number of channels	Input: $\mathbb{R}^{10}$
4×4 conv, 32 ReLU, stride 2	FC, 256 ReLU
4×4 conv, 32 ReLU, stride 2	FC, $4 \times 4 \times 64$ ReLU
4×4 conv, 64 ReLU, stride 2	FC, 4×4 upconv, 64 ReLU, stride 2
4×4 conv, 64 ReLU, stride 2	FC, 4×4 upconv, 32 ReLU, stride 2
FC 256, F2 2×10	4×4 upconv, 32 ReLU, stride 2
	4×4 upconv, number of channels, stride 2

Table 3 | Hyperparameters used for each model architecture

Model	Parameters	Values
β-VAE	β	[1, 2, 4, 6, 8, 16]
CCI-VAE	$c_{\max}$	[5, 10, 25, 50, 75, 100]
	iteration threshold	100000
	γ	1000
FactorVAE	γ	[10, 20, 30, 40, 50, 100]
DIP-VAE-I	λ_{od}	[1, 2, 5, 10, 20, 50]
	λ_d	$10\lambda_{od}$
DIP-VAE-II	λ_{od}	[1, 2, 5, 10, 20, 50]
	λ_d	λ_{od}
TC-VAE	β	[1, 2, 4, 6, 8, 10]

Model implementation details

We re-used the trained checkpoints from Locatello et al. (2018), hence we recommend the readers to check the original paper for model implementation details. Briefly, the following architecture and optimiser were used.

For consistency, all the models were trained using the same architecture, optimiser, and hyperparameters. All of the methods use a deep neural network to encode and decode the latent embedding and the parameters of the latent factors are predicted using a Gaussian encoder whose architecture is specified in Table 2. All of the models predict a latent vector with 10 factors. Each model was also trained with 6 different levels of regularisation strength specified in Table 3. The ranges of the hyperparameters used for the various levels of regularisation were specified to show a diversity of different performance on different datasets without relying on pre-existing intuition on good hyperparameters, however ranges were based on hyperparameters that were used previously in literature. For each of the model classes outlined above, we tried 6 hyperparameter values with 50 seeds each.

β-VAE The β-VAE (Higgins et al., 2017a) model is similar to the vanilla VAE model but with an additional hyperparameter β to modify the strength of the KL regulariser.

(a) Common hyperparameters across all models

Parameter	Values
Batch Size	64
Latent space dimension	10
Optimizer	Adam
Adam: beta1	0.9
Adam: beta2	0.999
Adam: epsilon	1e-8
Adam: learning rate	0.0001
Decoder type	Bernoulli

(b) FactorVAE discriminator architecture

Discriminator
FC, 1000 leaky ReLU
FC, 1000 leaky ReLU
FC, 1000 leaky ReLU
FC, 1000 leaky ReLU
FC, 1000 leaky ReLU
FC, 1000 leaky ReLU
FC, 2

(c) FactorVAE discriminator parameters

Parameter	Values
Batch size	64
Optimizer	Adam
Adam: beta1	0.5
Adam: beta2	0.9
Adam: epsilon	1e-8
Adam: learning rate	0.0001

Table 4 | Miscellaneous model details

$$\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \beta\ KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] \tag{9}$$

where a β value of 1 corresponds to the vanilla VAE model. Increasing β enforces a stronger prior on the latent distribution and encourages the representation to be independent.

CCI-VAE The CCI-VAE model (Burgess et al., 2017) is a variant of the β-VAE where the KL divergence is encouraged to match a controlled value C which is increased gradually throughout training. This yields us the objective function for CCI-VAE.

$$\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \beta\ |KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z})) - C|\ ] \tag{10}$$

FactorVAE FactorVAE (Kim and Mnih, 2018) specifically penalises the dependencies between the latent dimensions such that the "Total Correlation" term is targeted yielding a modified version of the β-VAE objective.

$$\begin{aligned}\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] -\ & KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] \\ & -\beta KL(q(\boldsymbol{z})||\prod_j q(\boldsymbol{z}_j))\end{aligned} \tag{11}$$

The "Total Correlation" term is intractable in this case so for FactorVAE, samples are used from both $q(\boldsymbol{z}|\boldsymbol{x})$ and $q(\boldsymbol{z})$ as well as the density-ratio trick to compute an estimate of the "Total Correlation" term. FactorVAE uses an additional discriminator network to approximate the density ratio in the KL divergence. The implementation details for the discriminator network and its hyperparameters can be found in Table 3(b) and 3(c).

TC-VAE The TC-VAE model (Chen et al., 2018) which independently from FactorVAE has a similar objective KL regulariser which contains a "Total Correlation" term. In the case of TC-VAE the "Total Correlation" term is estimated using a biased Monte-Carlo estimate.

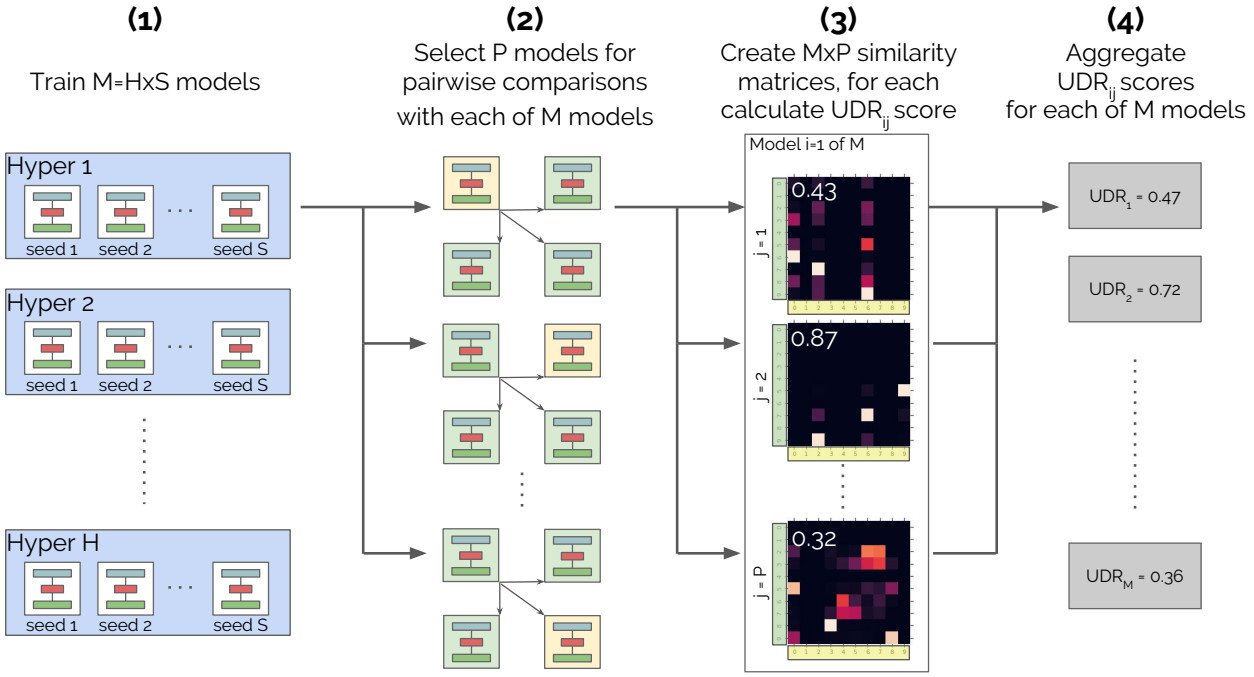

Figure 5 | Schematic illustration of the UDR method. See details in text.

DIP-VAE The DIP-VAE model also adds regularisation to the aggregated posterior but instead an additional loss term is added to encourage it to match the factorised prior. Since the KL divergence is intractable, other measures of divergence are used instead. $Cov_{p(\boldsymbol{x})}[\mu_\phi(\boldsymbol{x})]$ can be used, yielding the DIP-VAE-I objective

$$\begin{aligned}\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] -\ KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] \\ -\lambda_{od}\sum_{i\neq j}[Cov_{p(\boldsymbol{x})}[\mu_\phi(\boldsymbol{x})]]_{ij}^2 \\ -\lambda_d\sum_i([Cov_{p(\boldsymbol{x})}[\mu_\phi(\boldsymbol{x})]]_{ii}-1)^2\end{aligned} \tag{12}$$

or $Cov_{q_\phi}[\boldsymbol{z}]$ is used instead yielding the DIP-VAE-II objective.

$$\begin{aligned}\mathbb{E}_{p(\boldsymbol{x})}[\ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log\ p_\theta(\boldsymbol{x}|\boldsymbol{z})] -\ KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}))\ ] \\ -\lambda_{od}\sum_{i\neq j}[Cov_{q_\phi}[\boldsymbol{z}]]_{ij}^2 \\ -\lambda_d\sum_i([Cov_{q_\phi}[\boldsymbol{z}]]_{ii}-1)^2\end{aligned} \tag{13}$$

UDR implementation details

Similarity matrix To compute the similarity matrix R_{ij} we follow the approach of Li et al. (2016) and Morcos et al. (2018). For a given dataset $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., , \boldsymbol{x}_N\}$ and a neuron $a \in \{1, ..., L\}$ of model i (denoted as $z_{i,a}$), we define $\boldsymbol{z}_{i,a}$ to be the vector of mean inferred posteriors $q_i(\boldsymbol{z}_i|\boldsymbol{x}_i)$ across the full dataset: $\boldsymbol{z}_{i,a} = (z_{i,a}(\boldsymbol{x}_1), ..., z_{i,a}(\boldsymbol{x}_N)) \in \mathbb{R}^N$. Note that this is different from the often considered notion

of a "latent representation vector". Here $z_{i,a}$ is a response of a single latent dimension over the entire dataset, not an entire latent response for a single input. We then calculate the similarity between each two of such vectors $z_{i,a}$ and $z_{j,b}$ using either Lasso regression or Spearman's correlation.

Lasso regression (UDR_L) We trained L lasso regressors to predict each of the latent responses $z_{i,a}$ from z_j using the dataset of latent encodings $Z_{i,a} = \{(z_{j,1}, z_{i,a,1}), ..., (z_{j,N}, z_{i,a,N})\}$. Each row in $R_{ij}(a)$ is then filled in using the weights of the trained Lasso regressor for $z_{i,a}$. The lasso regressors were trained using the default Scikit-learn multi-task lasso objective $\min_w \frac{1}{2n_{samples}}||XW - Y||^2_{Fro} + \lambda||W||_{21}$ where Fro is the Frobenius norm: $||A||_{Fro} = \sqrt{\sum_{ij} a_{ij}^2}$ and the $l_1 l_2$ loss is computed as $||A||_{21} = \sum_i \sqrt{\sum_j a_{ij}^2}$. λ is chosen using cross validation and the lasso is trained until convergence until either 1000 iterations have been run or our updates are below a tolerance of 0.0001. Lasso regressors were trained on a dataset of 10000 latents from each model and training was performed using coordinate descent over the entire dataset. R_{nm} is then computed by extracting the weights in the trained lasso regressor and computing their absolute value (Eastwood and Williams, 2018). It is important that the representations are normalised per-latent such that the relative importances computed per latent are scaled to reflect their contribution to the output. Normalising our latents also ensures that the weights that are computed roughly lie in the interval $[-1, 1]$.

Spearman's based similarity matrix (UDR_S) We calculate each entry in the similarity matrix according to $R_{ij}(a, b) = \text{Corr}(z_{i,a}, z_{j,b})$, where Corr stands for Spearman's correlation. We use Spearman's correlation to measure the similarity between $z_{i,a}$ and $z_{j,b}$, because we do not want to necessarily assume a linear relationship between the two latent encodings, since the geometry of the representational space is not crucial for measuring whether a representation is disentangled (see Sec.), but we do hope to find a monotonic dependence between them. Spearman correlation coefficients were computed by extracting 1000 samples from each model and computing the Spearman correlation over all the samples on a per-latent basis.

All-to-all calculations To make all-to-all comparisons, we picked 10 random seeds per hyperparameter setting and limited all the calculations to those models. Hence we made the maximum of (60 choose 2) pairwise model comparisons when calculating UDR-A2A.

Informative latent thresholding Uninformative latents typically have KL$\ll$ 0.01 while informative latents have KL$\gg$ 0.01, so KL= 0.01 threshold in Eq. 3 is somewhat arbitrarily chosen to pick out the informative latents z.

Sample reduction experiments We randomly sampled without replacement 20 different sets of P models for pairwise comparison from the original set of 50 models with the same hyperparameter setting for UDR or 60 models with different seeds and hyperparameters for UDR-A2A.

Supervised metric implementation details

Original β-VAE metric. First proposed in Higgins et al. (2017a), this metric suggests sampling two batches of observations $\boldsymbol{x}$ where in both batches the same single data generative factor is fixed to a particular value, while the other factors are sampled randomly from the underlying distribution. These two batches are encoded into the corresponding latent representations $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ and the pairwise differences

Table 5 | Disentangled model selection metrics comparison. M - modularity, C - compactness, E - explicitness (Ridgeway and Mozer, 2018)

Metric	M	C	E
β-VAE	√	×	√
FactorVAE	√	√	√
MIG	√	√	√
DCI Disentanglement	√	×	×
UDR	√	×	×

between the corresponding mean latent values from the two batches are taken. Disentanglement is measured as the ability of a linear classifier to predict the index of the data generative factor that was fixed when generating $\boldsymbol{x}$.

We compute the β-VAE score by first randomly picking a single factor of variation and fixing the value of that factor to a randomly sampled value. We then generate two batches of 64 where all the other factors are sampled randomly and take the mean of the differences between the latent mean responses in the two batches to generate a training point. This process is repeated 10000 times to generate a training set by using the fixed factor of variation as the label. We then train a logistic regression on the data using Scikit-learn and report the evaluation accuracy on a test set of 5000 as the disentanglement score.

FactorVAE metric. Kim and Mnih (2018) proposed a modification on the β-VAE metric which made the classifier non-parametric (majority vote based on the index of the latent dimension with the least variance after the pairwise difference step). This made the FactorVAE metric more robust, since the classifier did not need to be optimised. Furthermore, the FactorVAE metric is more accurate than the β-VAE one, since the β-VAE metric often over-estimates the level of disentanglement by reporting 100% disentanglement even when only $K - 1$ factors were disentangled.

The Factor VAE score is computed similarly to the β-VAE metric but with a few modifications. First we draw a set of 10000 random samples from the dataset and we estimate the variance of the mean latent responses in the model. Latents with a variance of less than 0.05 are discarded. Then batches of 64 samples are generated by a random set of generative factors with a single fixed generative factor. The variances of all the latent responses over the 64 samples are computed and divided by the latent variance computed in the first step. The variances are averaged to generate a single training point using the fixed factor of variation as the label. 10000 such training points are generated as the training set. A majority vote classifier is trained to pick out the fixed generative factor and the evaluation accuracy is computed on test set of 5000 and reported as the disentanglement score.

Mutual Information Gap (MIG). The MIG metric proposed in Chen et al. (2018) proposes estimating the mutual information (MI) between each data generative factor and each latent dimension. For each factor, they consider two latent dimensions with the highest MI scores. It is assumed that in a disentangled representation only one latent dimension will have high MI with a single data generative factor, and hence the difference between these two MI scores will be large. Hence, the MIG score is calculated as the average normalised difference between such pairs of MI scores per each data generative factor. Chen et al. (2018) suggest that the MIG score is more general and unbiased than the β-VAE and FactorVAE metrics.

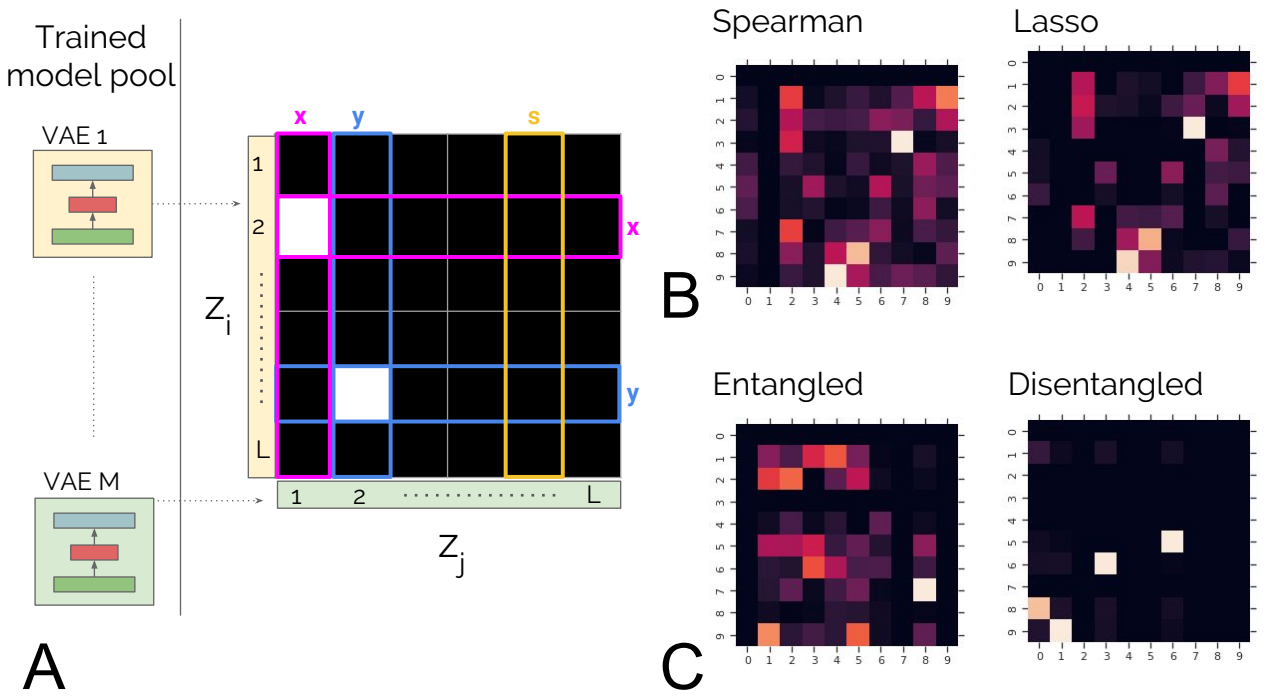

Figure 6 | **A:** Schematic illustration of the pairwise model comparison. Two trained models i and j are sampled for pairwise comparison. Both models learnt a perfectly disentangled representation, learning to represent two (positions x/y) and three (positions x/y, and size) generative factors respectively. Similarity matrix R_{ij}: white – high similarity between latent dimensions, black – low. **B:** Similarity matrix R_{ij} for the same pair of models, calculated using either Spearman correlation or Lasso regression. The latter is often cleaner. **C:** Examples of Lasso similarity matrices of an entangled vs a disentangled model.

We compute the Mutual Information Gap by taking the discretising the mean representation of 10000 samples into 20 bins. The disentanglement score is then derived by computing, per generative factor, the difference between the top two latents with the greatest mutual information with the generative factor and taking the mean.

$$\frac{1}{K}\sum_{k=1}^{K}\frac{1}{H_{v_k}}\left(I(z_j^{(k)}) - \max_{j\neq j_k} I(z_j, v_k)\right) \tag{14}$$

where K is the number of generative factors, from which v_k is a single generative factor z_j is the mean representation and $j^(k) = \text{argmax}_j I_n(z_j; v_k)$ is the latent representation with the greatest mutual information with the generative factor. H_{v_k} is the computed entropy of the generative factor.

DCI Disentanglement. This is the disentanglement part of the three-part metric proposed by Eastwood and Williams (2018). The DCI disentanglement metric is somewhat similar to our unsupervised metric, whereby the authors train a random forest classifier to predict the ground truth factors from the corresponding latent encodings $q(\boldsymbol{z}|\boldsymbol{x})$. They then use the resulting $M \times N$ matrix of feature importance weights to calculate the difference between the entropy of the probability that a latent dimension is important for predicting a particular ground truth factor weighted by the relative importance of each dimension.

The DCI disentanglement metric is an implementation of the disentanglement metric as described in Eastwood and Williams (2018) using a gradient boosted tree. It was computed by first extracting

Table 6 | Rank correlations between each of the scores produced by the four versions of UDR and four supervised metrics. The scores are averaged over three model classes, two datasets and four supervised metrics. See Supplementary Material for details.

UDR	Lasso	Spearman	Supervised
Hyper	0.54 ± 0.06	0.53 ± 0.07	0.67 ± 0.2
All-to-all	0.60 ± 0.11	0.59 ± 0.10	

the relative importance of each latent mean representation as a predictor for each generative factor by training a gradient boosted tree using the default Scikit-learn model on 10000 training and 1000 test points and extracting the importance weights. The weights are summarised into an importance matrix R_{ij} with the number of rows equal to the number of generative factors and columns equal to the number of latents. The disentanglement score for each column is computed as $D_i = (1 - H_K(P_i))$ where $H_K(P_i) = -\sum_{k=0}^{K-1} P_{ik} log_K P_{ik}$ denotes the entropy. $P_{ik} = R_{ij}/\sum_{k=0}^{K-1}$ is the probability of the latent factor i in being important for predicting factor k. The weighted mean of the scores for the column is computed using the relative predictive importance of each column as the weight $D = \sum_i p_i * D_i$ where $p_i = \sum_j R_{ij}/\sum_{ij} R_{ij}$.

Additional results

We evaluated four UDR versions, which differed in terms of whether Spearman- and Lasso-based similarity matrices R_{ij} were used (subscripts S and L respectively), and whether the models for pairwise similarity comparison are picked from the pool of different seeds trained with the same hyperparameters or from the pool of all models (the latter indicated by the A2A suffix). The A2A correlations in Tbl. 6 are on average slightly higher, however these scores are more computationally expensive to compute due to the higher number of total pairwise similarity calculations. For that reason, the scores presented in the table are calculated using only 20% of all the trained models. Hence, the results presented in the main text of the paper are computed using the UDR_L score, which allowed us to evaluate all 5400 models and performed slightly better than the UDR_S score. Figs. 7-9 provide more details on the performance of the different UDR versions.

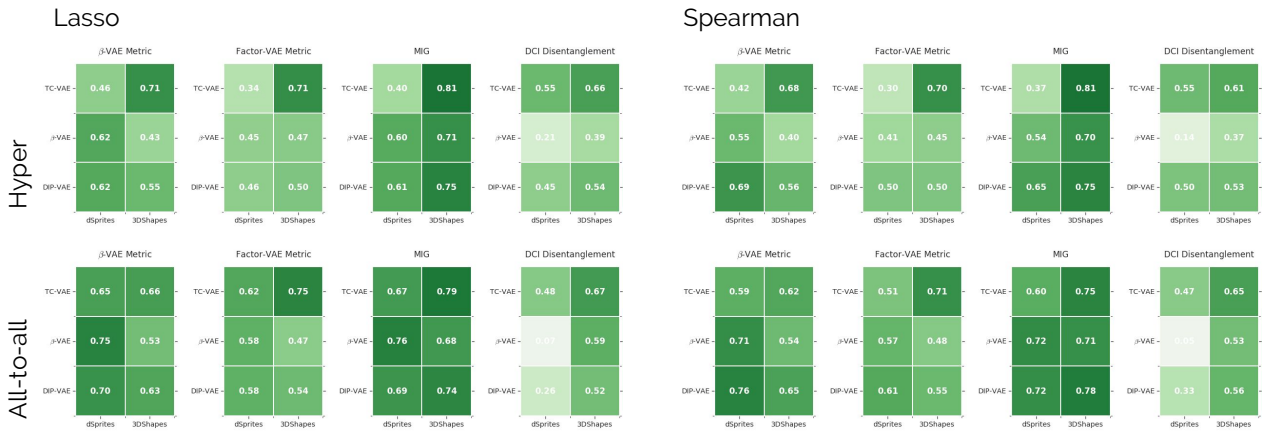

Figure 7 | Rank correlation between different versions of UDR with different supervised metrics across two datasets and three model classes. We see that the UDR_L approaches slightly outperform the UDR_S ones.

To qualitatively validate that the UDR method is ranking models well, we look into more detail into the β-VAE model ranking when evaluated with the DCI disentanglement metric on the dSprites dataset. This scenario resulted in the worst disagreement between UDR and the supervised metric as shown in Fig. 7. We consider the UDR_L version of our method, since it appears to give the best trade off between overall correlations with the supervised metrics and hyperparameter selection accuracy. Fig. 10 demonstrates that the poor correlation between UDR_L and DCI Disentanglement is due to the supervised metric. Models ranked highly by UDR_L but poorly by DCI Disentanglement appear to be qualitatively disentangled through visual inspection of latent traversals. Conversely, models scored highly by DCI Disentanglement but poorly by UDR_L appear entangled.

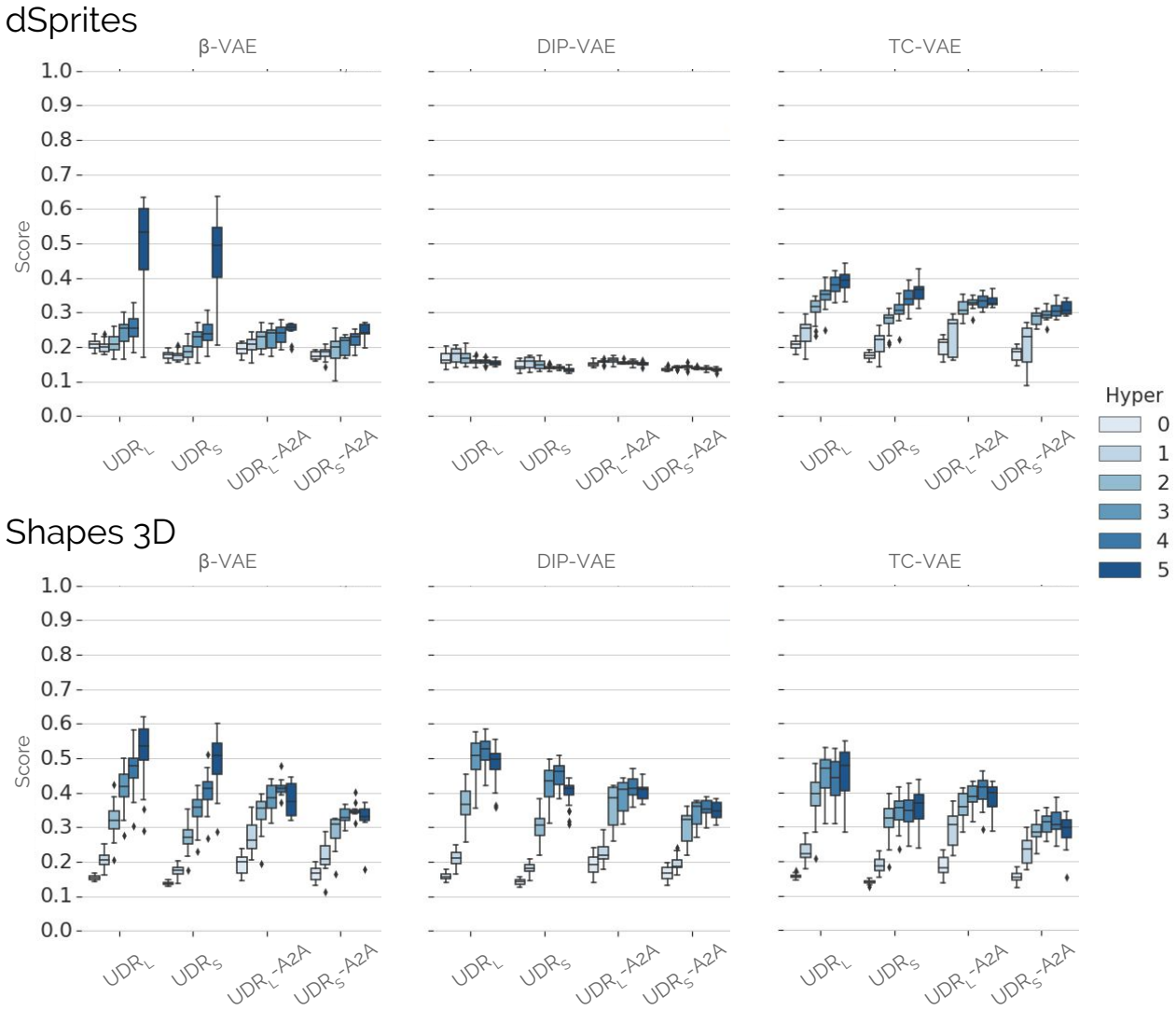

Figure 8 | The range of scores for each hyperparameter setting for the dSprites and 3D Shapes datasets for various models and metrics. We see that the different versions of the UDR method broadly agree with each other.

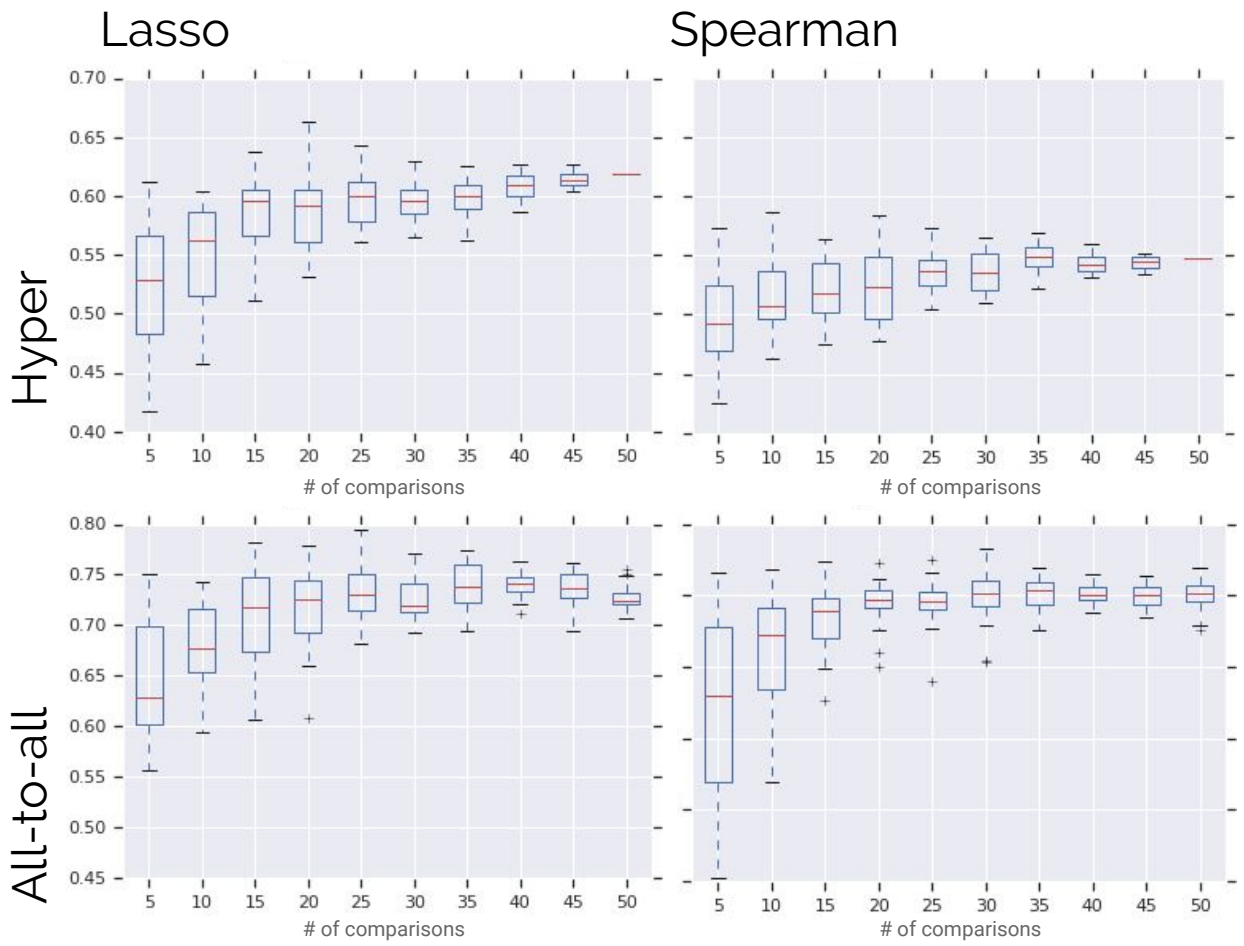

Figure 9 | Rank correlations of the different versions of the UDR score with the β-VAE metric on the dSprites dataset for a β-VAE hyperparameter search as the number of pairwise comparisons per model were changed. Higher number of comparisons leads to more accurate and more stable rankings, however these are still decent even with 5 pairwise comparisons per model.

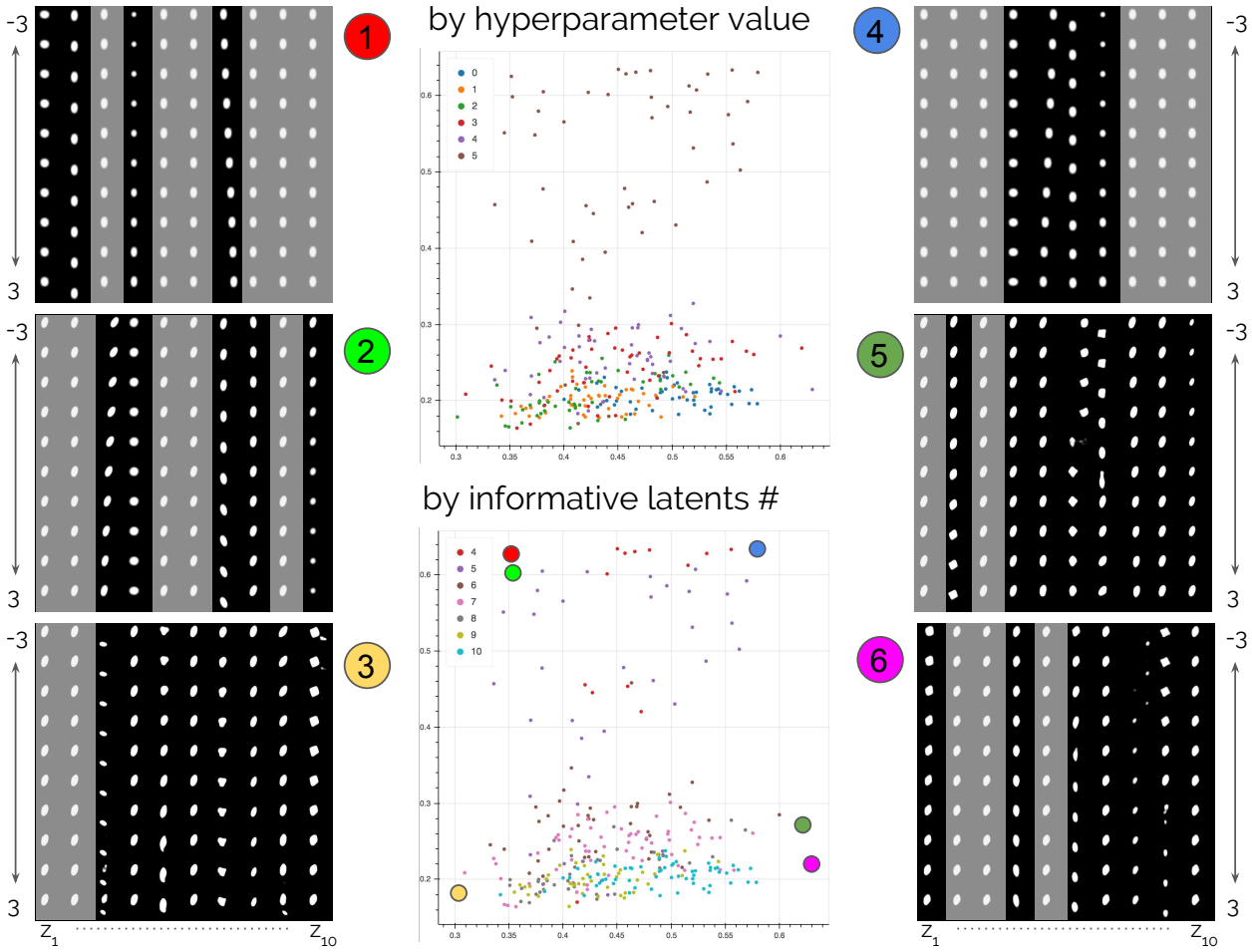

Figure 10 | Example latent traversals of some of the best and worst ranked β-VAE models using the UDR_L (ordinate) and DCI Disentanglement (abscissa) metrics, coloured either by hyperparameter value (top) or final informative latent number (bottom). Uninformative units are greyed out. The models ranked highly by UDR_L do appear to be well disentangled, despite being ranked poorly by DCI Disentanglement (1, 2, 4). On the other hand, models ranked well by DCI Disentanglement but poorly by UDR_L look quite entangled (5, 6). Finally, models ranked poorly by both metrics do appear entangled (3).

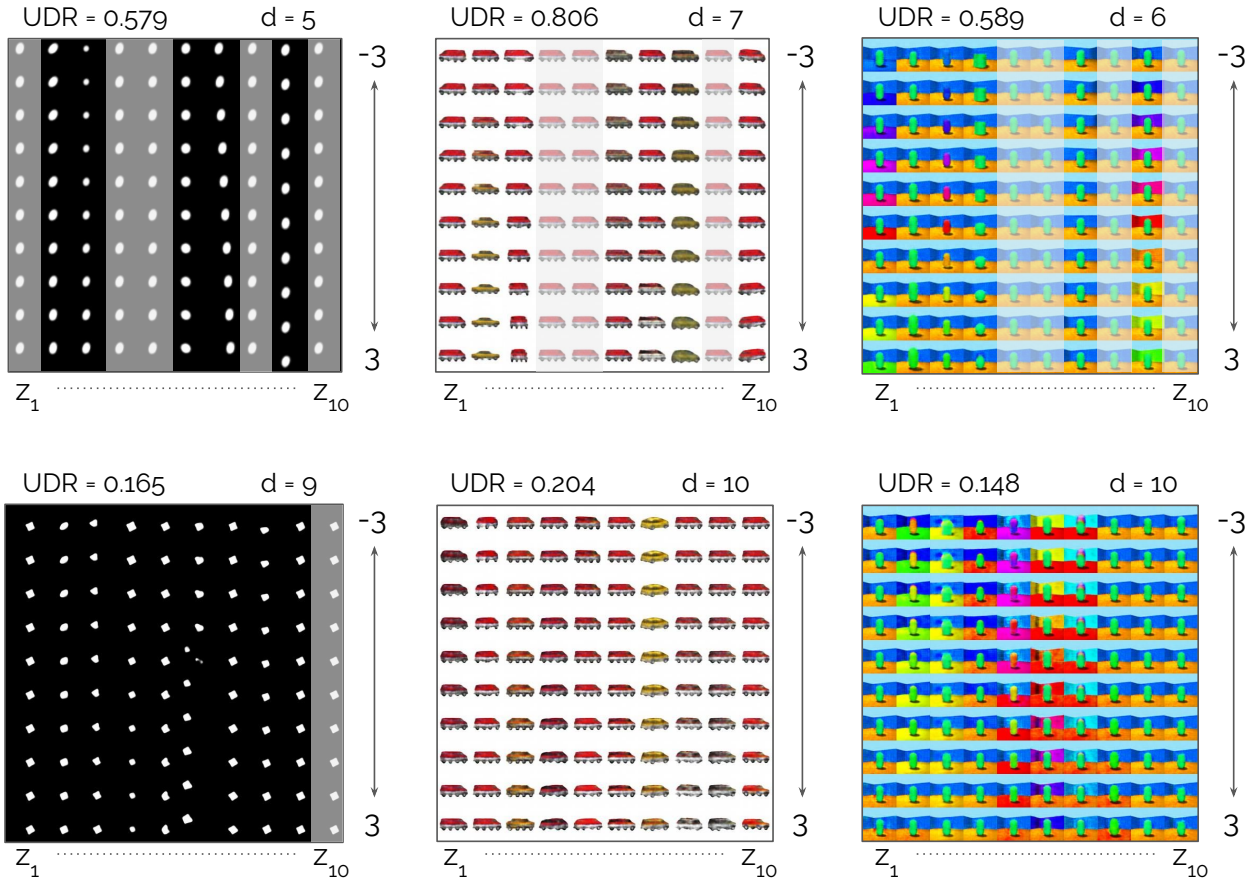

Figure 11 | Example latent traversals of some of the best and worst ranked β-VAE models using the UDR_L scores. Uninformative latents are greyed out.