Lost in Latent Space: Examining failures of disentangled models at combinatorial generalisation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent research has shown that generative models with highly disentangled representations fail to generalise to unseen combination of generative factor values. These findings contradict earlier research which showed improved performance in out-of-training distribution settings when compared to entangled representations. Additionally, it is not clear if the reported failures are due to (a) encoders failing to map novel combinations to the proper regions of the latent space, or (b) novel combinations being mapped correctly but the decoder is unable to render the correct output for the unseen combinations. We investigate these alternatives by testing several models on a range of datasets and training settings. We find that (i) when models fail, their encoders also fail to map unseen combinations to correct regions of the latent space and (ii) when models succeed, it is either because the test conditions do not exclude enough examples, or because excluded cases involve combinations of object properties with it's shape. We argue that to generalise properly, models not only need to capture factors of variation, but also understand how to invert the process that causes the visual stimulus.

1 Introduction

Disentangled representations extract factors of variations from data, and learning them has been the focus of much research in recent years [10]. Several approaches have been proposed to induce disentanglement, including latent space penalization [10, 3, 17], different training regimes [23, 22], architectures [36], data-driven inductive biases [21] and model selection methods [5]. These have produced more interpretable representations [12] that improve sample efficiency and learning for downstream models [13, 33, 5].

Importantly, researchers have also hypothesized that disentangled representations could provide a way of improving generalisation performance by enabling the discovery of causal variables in data [29] or by capturing its compositional structure [5]. This claim is especially interesting as it allows machine learning systems to emulate a key property of human intelligence – the ability to generalise to unseen combinations of known elements. For example, if a model has learned to generate red triangles and blue squares, then the model should also be able to correctly generate blue triangles and red squares. This property, which we refer to as *combinatorial generalisation*, gives humans the ability to make "infinite use of finite means" [34, 4, 31, 25] and has been termed "a top priority for AI to achieve human-like abilities" [1]. Indeed, several authors have reported that unsupervised models that are better at disentangling generative factors are also better at some forms of combinatorial generalisation [10, 13, 36]

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

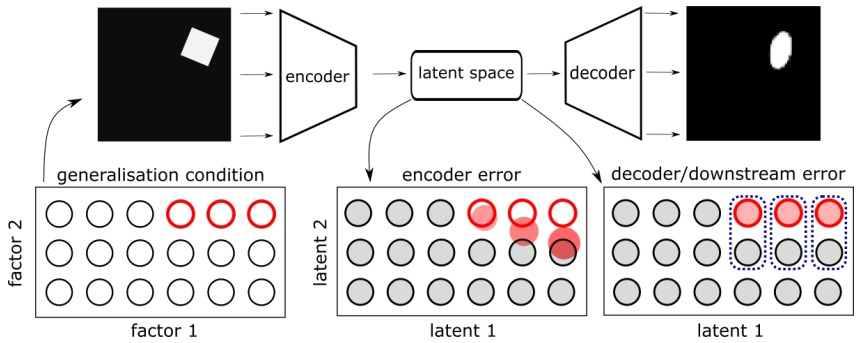

Figure 1: **Possible problems for combinatorial generalisation**. *Left*: visualisation of how a combinatorial generalisation condition can be defined for a dataset containing two generative factors. Black and red circles denote training and test examples, respectively. In the Middle and Right panels, the outline of each circle shows the target position of the representation in latent space, while a shaded circle shows the position at which the latent representation is projected by the encoder. *Middle*: first type of combinatorial generalisation error, encoder error. The encoder projects unseen inputs to different parts of the latent space than what was expected based on their generative factor values (shown as shaded circles falling outside their target outlines) *Right*: second type of combinatorial generalisation error, decoder error. Observed representations (shaded circles), are mapped to the correct position in latent space (circle outlines), but a decoder/downstream process mixes-up the black and red representations (blue dashed lines).

However, two recent studies have reported evidence that contradicts this hypothesis. Montero et al. [26] considered datasets where inputs varied along several dimensions (generative factors) and divided generalisation conditions into different types. They found that VAEs with highly disentangled latent representations succeeded at the easiest generalisation conditions where only one combination of all generative factors was excluded (a condition they termed *recombination-to-element*), but failed at more challenging conditions, where all combinations of a subset of generative factors were excluded (termed *recombination-to-range* by Montero et al. [26]). In another second study, Schott et al. [30] tested 17 learning approaches, expanding the analysis to other paradigms, architectures and task structures. They observed that models showed moderate success at generalising on some artificial datasets, such as 3DShapes. However, their ability to generalise dropped significantly on more realistic datasets and in more challenging generalisation conditions.

These contradictory results could be down to a number of differences between studies that report successes and those that report failures. For example, there were qualitative differences in the datasets that were tested and how the test conditions were set up. This is indeed what Montero et al. [26] conclude, writing that "it is not clear what exactly was excluded while training [previous models]", suggesting that previously observed successes may have been on the simplest generalisation condition where only very few combinations were left out. But this still leaves open the question of why models that achieve high degree of disentanglement nevertheless fail at combinatorial generalisation under more challenging conditions.

We see two possible reasons why this could happen. One possibility is that models correctly infer the values of latent variables (Fig. 1, left) for novel combinations of generative factors, but the decoder fails to map these unseen latent values to the (output) image space – *decoder error* (Fig. 1, right). Some evidence supporting this hypothesis was observed by Watters et al. [36], who looked at not only the output image reconstructions, but also at latent representations in a simple reconstruction task. Watters et al. [36] observed that, in their task, models that have more disentangled latent representations do indeed map unseen combinations of generative factors to the correct values in the latent space. But they only tested a very simple dataset, involving only two generative factors. Montero et al. [26] showed that models can sometimes succeed to generalise in simple settings where they can solve the combinatorial generalisation problem through interpolation, but struggle at more challenging settings where larger number of combinations are excluded from the training set.

Another possibility is that current models fail at harder forms of combinatorial generalisation (such as recombination-to-range) due to an encoding failure – *encoder error*. That is, the encoder fails to map these harder unseen combinations of generative factors to the correct values of variables in the latent space (Fig. 1, middle). If this were the case, then it reflects a more fundamental limitation of

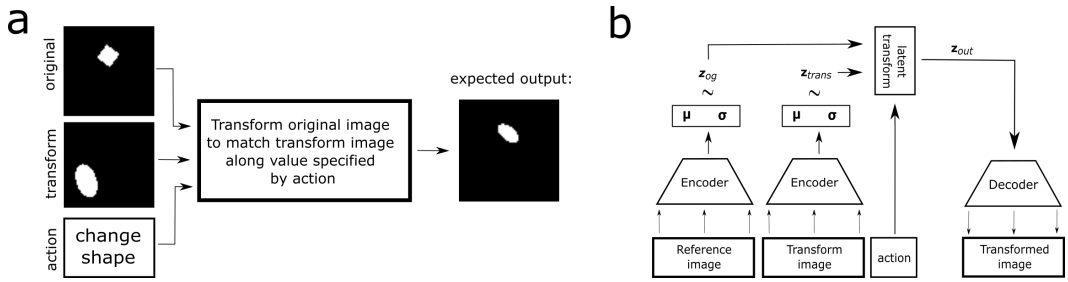

Figure 2: (a) **Image composition task**. An example of the composition task. In this case, the `[shape]` of the output must match the *transform* image and the rest of the values (`[position, orientation]`) must match the reference image. (b) **Model architecture**. The model uses the same encoder to represent both images. Then a transform takes these latent representation and combines them to produce a transformed representation, which is used to reconstruct the output image. Reproduced from Montero et al. with permission from the authors.

current models, showing that even though models can map observed combinations to correct values of latent representations, they do not understand how generative factors are combined. So when novel combinations are presented, models cannot infer the values of generative factors that led to the observed data.

In this work we explore these issues by examining the latent space of generative models. Extending the findings of Montero et al. [26] and Watters et al. [36], we show that failure to generalise in the output space is accompanied by failure to generalise in latent space across several datasets. We show that these results hold for a broad range of architectures (the standard *deconvolution decoder* as well as the *spatial broadcast decoder*), loss functions (β-*VAE* and *WAE*) and task settings. Finally, by looking at the latent space, we discover that, in addition to the difficulty of the generalisation task, a crucial condition for failure of combinatorial generalisation is the way in which generative factors are combined. These results not only challenge research that argues that disentanglement leads to better generalisation [13, 36], but also shows that the failure to generalise is a lot more entrenched in current models than suspected by Montero et al. [26] and Schott et al. [30].

2 Testing generalisation in latent representations

In this section we first describe a carefully designed semi-supervised task that enables us to achieve latent representations that are highly disentangled. This is crucial because standard disentanglement learning objectives, which rely on minimizing total correlation [10, 3, 17], struggle to reliably achieve a high degree of disentanglement. This is problematic because it is difficult to test whether entangled models are projecting test data to the expected position in the latent space (see Figure 1). Secondly, we describe our choice of corresponding model architectures as well as loss functions. For our results to be broadly applicable to existing research, we chose standard encoder and decoder architectures as well as a more sophisticated approach designed to induce a high degree of disentanglement [36]. Lastly, we describe some challenging test conditions for combinatorial generalisation on well known datasets.

2.1 Experimental Setup

2.1.1 Task

One of the main barriers to achieving disentanglement in standard unsupervised training is the non-identifiability of the models when using iid data [15, 21]. In short, this is because there are infinitely many linear combinations of the underlying generative factors that produce a valid basis on which they can be represented.

To get around this problem we used the composition task developed by Montero et al. [26] (see Appendix A.1 for further discussion). This task takes two images and a query vector as inputs (see Figure 2(a)). The goal of the task is to output an image that combines the two images based on an *action* in the query vector:

$$\textbf{input} = \mathbf{x_{og}},\ \mathbf{x_{trans}},\ \mathbf{q}$$
$$\textbf{output} = \mathbf{x_{out}}$$

where $\mathbf{x_{og}}$ and $\mathbf{x_{trans}}$ are the two input images, $\mathbf{x_{out}}$ is the output image, and $\mathbf{q}$ is the query vector. Following Montero et al., we used actions that involve replacing one of the properties (generative factors) of $\mathbf{x_{og}}$ with a property of $\mathbf{x_{trans}}$, based on the value of $\mathbf{q}$. This vector was a one-hot encoding of the generative factor that was required to be changed.

By combining a semi-supervised component (the query vector plus image reconstruction) with sparse transitions (only one generative factor needed to be modified at a time), the composition task provided a strong inductive bias towards high levels of disentanglement, as has been shown in previous work that exploits these approaches [23, 22, 21].

2.1.2 Models

The model architecture used to solve the composition task is shown in Figure 2(b). It involves two encoders that map each image to a latent space. The latent vectors are then "composed" (see below), based on the query (action) vector. This composed vector is mapped from the latent space into the (output) image space using a decoder. Encoders and decoders for generative models match the ones used in past work [10, 17] for both `dSprites` and `3DShapes` datasets. For `MPI3D` we added another 2 more layers, which we found was necessary for models to learn the task. For the Circles dataset we used the encoder defined in Watters et al, along with the corresponding Spatial Broadcast Decoder architecture. Latent representations for the generative models are standard diagonal Gaussians with input dependent means and variances, to which we applied both the Variational and Wasserstein loss [19, 32].

We can define the composition operation that combines both latent representations as:

$$\mathbf{z_{out}} = \mathbf{z_{og}} \odot (\mathbf{1} - \mathbf{c}) + \mathbf{z_{trans}} \odot \mathbf{c}$$

where $\mathbf{z_{out}}$, $\mathbf{z_{og}}$ and $\mathbf{z_{trans}}$ are the latent representations corresponding to the output, original, and transform images and $\odot$ is the element-wise product between vectors. The variable $\mathbf{c}$ is the interpolation vector of coefficients. We defined two ways of computing $\mathbf{c}$: (i) a learned interpolation function computed as $\sigma(W \cdot [\mathbf{z_{og}}; \mathbf{z_{trans}}; \mathbf{q}] + \mathbf{b})$ with parameters W and $\mathbf{b}$ where $[.;.]$ represents the concatenation of vectors; and (ii) a fixed interpolation function $\mathbf{c} = \text{padded}(\mathbf{q})$ which just pads $\mathbf{q}$ with zeros to match the vector length of the latent representation (see Appendix A.2 for more details).

To facilitate our goal of testing combinatorial generalisation, we augmented the target output by requiring the model to reconstruct the input images as well. Thus the trained encoder and decoder form a valid standard autoencoder. We can then probe the image reconstructions and the latent representations for unseen combinations to check if the models generalise. An illustration of this model can be found in Figure 2.

2.1.3 Datasets

We tested our models on three standard datasets: `dSprites` (a sythesised dataset of 2D shapes procedurally generated from 5 independent generative factors [24]), `3DShapes` (another synthesised dataset consisting of 3D shapes procedurally generated from 6 ground truth generative factors [2]) and `MPI3D` (a collection of four different datasets consisting of synthetic as well as real-world images procedurally generated from 7 generative factors [8]). These are the most popular datasets used in the disentanglement literature that also provide ground truth generative factor values. We tested combinatorial generalisation by systematically excluding combinations of values of generative factors from each dataset. Here we focus on the conditions described as *recombination-to-range* by Montero et al.. These are the most interesting conditions where, according to Montero et al., one would expect a model that learns disentangled representations to succeed at combinatorial generalisation, but tested models typically fail. Consider a dataset with, say, four generative factors $[g_1,\ g_2,\ g_3,\ g_4]$ where all $g_i \in [0, 1]$. The *recombination-to-range* condition creates a training/test split where all examples with combinations of a subset of generative factors are excluded from the training set and added to the test set. Thus, an example of a dataset that tests recombination-to-range may consist of a training set where all combinations where $[g_1 > 0.5,\ g_2 > 0.5]$ have been excluded from the training set and added to the test set. Note that the model trained on such a datasets would come across a number of

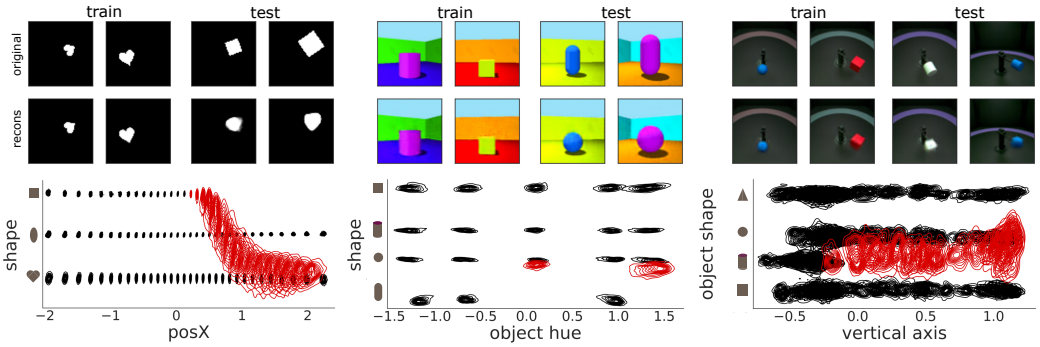

Figure 3: **Generalisation in latent space** *Top*: Typical examples of reconstructions for training and test images for models that learn highly disentangled representations. *Bottom*: Visualisation of the latent space of models trained on three datasets – `dSprites` (left), `3DShapes` (middle) and `MPI3D` (right). Each panel shows the joint distribution of two latent variables that best predict the corresponding generative factors. In all cases, the red distributions indicate test data and the black ones indicate training data. Note that the latent values in this figure are *not* necessarily the same as the values of generative factors excluded. This is because every model ends up with a different internal representation based on its initialisation and sequence of training trials. However, in each case, we observed that these internal representations were highly structured when the model showed a high degree of disentanglement.

examples where $[g_1 > 0.5]$ and also examples where $[g_2 > 0.5]$, but never be trained on an example where both these conditions are true simultaneously. This method was used to create training / test sets for each of the datsets in the following manner:

- `dSprites`: All images such that `[shape=square, posX`> 0.5`]` were excluded from the training set. Squares never appear on the right side of the image, but do appear on the left and other shapes (hearts, ellipsis) are observed on the right as well.
- `3DShapes`: All images such that `[shape=pill, object-hue=`> 0.5`]` were excluded from the training set. Thus, pills colored as any of the colors in the second half of the HSV spectrum did not appear in the training set. These colors (shades of blue, purple, etc) were observed on the other shapes, and the pill was observed with other colors such as red and orange.
- `MPI3D`: All images such that `[shape=cylinder, vertical axis`> 0.5`]` were excluded from the training set. We also excluded all images where `[shape=cone]` or `[shape=hexagonal]` as these shapes are very similar to the pyramid and cylinder, respectively and make it hard to access reconstruction accuracy.

2.1.4 Measuring Disentanglement

We used the DCI metric from Eastwood & Williams [6] to measure the degree of disentanglement (see Appendix A.5). This metric provides a set of regression weights between ground truth factors and latent variables. We selected the latent variable corresponding to each ground truth factor using a matching algorithm based on these weights. Crucially, for highly disentangled models any combination of generative factors will have a corresponding combination of latent variables of the same cardinality. This allows us to create visualisations of the latent representations analogous to the ones in Figure 1.

2.2 Results

We observed that most models trained on the composition task managed to achieve a reasonably high degree of disentanglement, and successfully reconstructed images in the training data (see Appendix B.1 and B.2 for quantitative results). Here we present results of typical models that achieved a very high degree of disentanglement (with alignment scores typically > 0.95, see Appendix B.1 for details). Figure 3 shows reconstructions as well latent space representations of three typical models trained on the `dSprites`, `3DShapes` and `MPI3D` datasets, respectively.

Replicating results of Montero et al. [26], we observed that models showed poor generalisation to unseen combinations for all three datasets (see reconstruction of test images in Figure 3). For

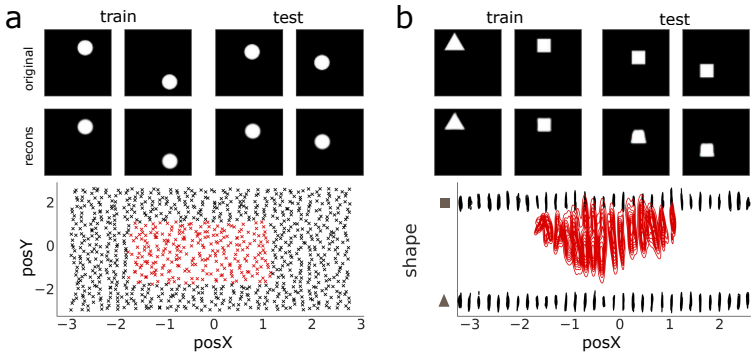

Figure 4: **Latent space induced by SBD** Visualisation of the latent space for models that use a Spatial Broadcast Decoder as described in Watters et al. Both visualisations correspond to a condition where combinations in the middle of the image are removed. (a) The result when this is done for the Circles dataset which uses only one shape and where the training set excludes all combinations `[shape=circle,` $0.35 <$ `posX` < 0.65, $0.35 <$ `posY` < 0.65`]` (b) The result for the a similar condition for the Simple dataset which uses two shapes and excludes all combinations `[shape=square,` $0.35 <$ `posX` < 0.65, $0.35 <$ `posY` < 0.65`]`.

`dSprites`, the models typically produced images with correct location, but made an error in the target shape (e.g., replacing the square with an ellipse). Similarly, with `3DShapes` and `MPI3D`, models typically reconstructed shapes with the correct colour hue (`3DShapes`) or vertical location (`MPI3D`), but made an error in the shape of the target object (e.g., replacing the pill (`3DShapes`) or cylinder (`MPI3D`) with a sphere(`3DShapes`) or pyramid (`MPI3D`).

Next, we looked at the latent representations of each of these cases by selecting a latent variable that showed highest correlation with each generative factor as described in Section 2.1.4. Figure 3 plots the distributions of the value of each latent variable for every combination of values of the relevant generative factors. Visualising these probability distributions for the combination of generative factors seen during training (in black) confirmed that the models learned highly disentangled representations (note the low variance of the probability distributions for trained combinations, especially for `dSprites` and `3DShapes`).

Crucially, we observed that the encoder failed to map the generative factors for the test combinations to the correct location in the latent space (red distributions). This was particularly true for `dSprites` and `3DShapes` dataset: note how the mean of the probability distributions for the left out combinations (`[shape=square, posX`> 0.5`]` for `dSprites` and `[shape=pill, object-hue=[blue, purple]]`) are shifted to overlap the latent representations seen during training. The shifts in the mean of distributions were less acute for `MPI3D`, but we nevertheless observed an increase in the variability of test distributions and a consistent shift in means for values of generative factors that were far from the trained values.

In summary, we made three key observations in these experiments. First, models replicated the negative results reported previously: i.e., a degradation in reconstruction performance in the challenging combinatorial generalisation conditions. Second, by visualising the latent space, we observed that failures of generalisation coincided with poor latent representations, showing that failures of generalisation are not entirely due to decoder errors. Third, the failures in latent representations as well as output reconstructions showed that different generative factors have important qualitative differences, with models making larger errors in generating the correct shape for unseen combinations than in reproducing the position, scale or color (see Appendix B.2).

2.3 Exploring different decoder architectures

In the experiments above, we used the deconvolution network as the decoder, which is the standard decoder used in VAEs [10]. However, some studies have shown that replacing the deconvolution network with a different architecture helps the model learn representations that are more disentangled. One such architecture is the Spatial Broadcast Decoder (SBD from hereon) developed by Watters et al. [36], who showed that such a decoder not only helped in learning representations that were highly disentangled, but also helped in solving the problem of combinatorial generalisation.

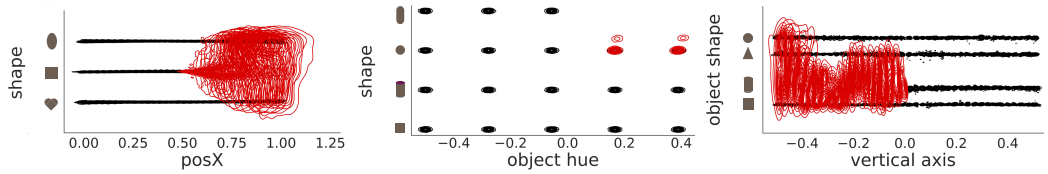

Figure 5: **Replacing decoder with ground-truth** Visualisation of the latent space for models trained to predict the correct generative factor values for each image. The visualizations correspond to the same test conditions as the ones presented in Figure 3: (a) `dSprites`, where all combinations where `[shape=square, posX` > 0.5`]` were excluded from the training set (here, this corresponds to the middle latent value) (b) `3DShapes`, where all combinations where `[shape=pill, object-hue` > 0.5`]` were excluded (corresponding to the top latent value), and (c) `MPI3D`, where all combinations where `[shape=cylinder, vertical-axis` > 0.5`]` were excluded (the second from bottom value).

In our next set of experiments, we tested whether using the SBD succeeds at combinatorial generalisation in more challenging settings, akin to the ones we tested above. We first replicated the results for both conditions tested by Watters et al. [36], using our composition task on the Circles dataset and replacing the deconvolution network with the SBD. Figure 4 (panel A) shows the results for the first condition, where the training set excluded images with circles presented in the middle of the canvas (see Appendix B.2 for results on the second condition). As we can see from this figure, the latent space learned by the model was indeed highly structured and, crucially, the model mapped the latent values for the left out combinations (red crosses) to the correct region of the latent space.

Next, we wanted to check how these results scaled when we made the generalisation test more challenging, excluding a combination of values for a range of variables (the *recombination-to-range* condition developed by Montero et al. [26]). We can do this with the Circles dataset by adding a third generative factor and the simplest way of doing this is by letting the shape take one of several values, instead of being a circle for all examples.

We tested this by constructing a new dataset called Simple. Like the Circles dataset, this dataset consisted of images that had a shape located at various (x and y) positions on the canvas. Unlike the Circles dataset, this shape could be either a triangle or a square (we chose these shapes as they are different enough for models to not confuse them by accident). We then tested the two conditions similar to the conditions tested by Watters et al. [36], excluding all combinations where one of the shapes was presented in the middle (first condition) or the top-right (second condition) of the canvas.

The results for the first condition are shown in panel B of Figure 4 (see Appendix for results of the second condition). Like the results for the first set of experiments above, we have plotted the probability distribution of combinations of values of the the latent variables (here, shape and posX) for all values of the third latent variable (here, posY). Similar to our earlier results, we observed that even the model using the SBD failed to map the unseen (test) combinations to the correct position in the latent space (compare the mean and variability of the red distributions compared to the black ones). Correspondingly, we also observed that models failed to correctly reconstruct output images for the test conditions, even when they had no problem reconstructing these images for combination of generative factors seen during training (See Appendix B.2).

3 Replacing decoder with ground-truth latent representations

We have shown that errors in generalisation occur in latent space as well in the reconstructions for generative models. To conclude this analysis we show that the problem does not lie in the generative nature of the task by testing the encoders on latent prediction task, where the model has to predict correct latent representations based on an input image. We used the same datasets and generalisation conditions as before.

3.1 Experimental setup

We trained models on a prediction task where the model was given a single image as the input and the target was to output the ground truth generative factor values. This ensures that the resulting latent representation will be completely disentangled, as the model must output each of the factor

values separately. We used the mean squared error of the output and target vectors as a learning signal. All other parameters remain same as the composition task. We used the same architectures for the feed-forward models as the encoders in the corresponding datasets above.

We evaluated the models using the R^2 metric as used in Schott et al. [30] to check whether the models had solved the task. We also use the same method to visualise the results as described above, plotting the joint probability distributions for the combination of generative factors along which the model was tested for generalisation. In this case, it is straightforward to do so as the relevant output dimensions shared the same index as the corresponding generative factor.

3.2 Results

The results for the three dataset are shown in Figure 5. We observed a very similar pattern for this task as the semi-supervised task above. For all three datasets, the probability distributions of latent values for the left out combinations (in red) showed a much higher variance and means that were shifted towards combinations of latent values that had been experienced during training. Thus, even when the encoders were trained to recognise perfectly disentangled generative factors, they failed to generalise to combinations that were not experienced during training.

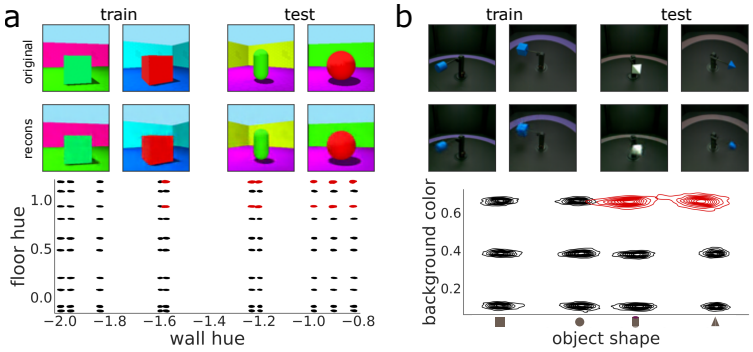

Figure 6: **Latent space for conditions where models succeed** Visualisation of the latent space for models trained on conditions in which they succeed at generalisation. The figure follows the same conventions as in Figure 3. (a) Results for 3DShapes when removing floor and wall color combinations. (b) Visualisation for the MPI3D dataset for combinations where the salmon background has not been seen with the cylinder shape.

4 Explaining contradictory findings

How can we reconcile the failures of generalisation observed above with with past studies that showed successful combinatorial generalisation on some datasets and conditions? One response is that the generalisation conditions we test are more challenging than the ones tested in previous studies because they exclude more data from the training set. However, it is also possible that the success of combinatorial generalisation is not only determined by how many combinations are excluded from the test condition, but also by which combination of generative factors are tested.

All failures observed above were for cases where the generative factor 'shape' is combined with another generative factor (position / color). It is possible that this was because shape combines with other factors in a qualitatively different manner than, say, the floor hue combines with the wall hue in `3DShapes`. In order to completely disentangle shape from position / color / orientation, the model must come up with a representation of shape that is invariant to a change in these generative factors. And it must do so by learning from training examples where shape and orientation *interact* – that is, the observed image is obtained via a composition of the two generative factors affecting the same part of the image.

But not all combinatorial generalisation problems face this problem of working out how generative factors combine to determine the same part of the observed image. In some cases, an image may be determined by a combination of generative factors, but each factor governs a different part of the image, such as the generative factors `[floor-hue, wall-hue]` in `3DShapes`. In such cases, learning

a representation of one factor that remains invariant to the other factor is trivial and the model may succeed at combinatorial generalisation, even though the same amount of data is excluded from the training set. To test this hypothesis, we carried out a set of experiments where the model had to solve the problem of combinatorial generalisation, but the combinations that were excluded did not interact in the training examples. We can do this for both the `3DShapes` dataset as well as the `MPI3D` dataset, as both datasets involve images where the canvas contains several disjoint elements. So, we repeated the semi-supervised learning experiments above, where models had to learn the composition task, but instead of excluding combinations where the generative factors interacted with each other, we excluded the following combinations from the training set:

- `3DShapes`: Exclude all combinations such that `[floor-hue` < 0.25`, wall hue` > 0.75`]`: none of the training images had the combination of floors with a "warm" hue and walls with a "cold" hue.
- `MPI3D`: Exclude all combinations such that `[shape={cylinder,sphere}, background color=salmon]`. Note, even though shape combinations are excluded, the combination of generative factors excluded do not interact (i.e. determine different parts of the image).

4.1 Results

Unsurprisingly, we again observed that models learned highly disentangled representations for this task. More interestingly, we also observed that models now succeeded at the combinatorial generalisation task. Figure 6 shows some typical reconstructions for training as well as test images. These reconstructions successfully reproduce the unseen combination of floor hues and wall hues for the `3DShapes` dataset and the combination of shape and background arc color for the `MPI3D` dataset. This figure also shows the joint probability distribution for trained as well as novel combination of latent values. In contrast to Figure 3, we now see that the means of test latent distributions (in red) fall in the expected location and a show a much smaller variance.

5 Discussion

Our world is inherently compositional – it can be decomposed into simpler parts and relationships between these parts. The idea behind learning disentangled representations is to recover this underlying compositional structure of the world from perceptual inputs. Unsupervised learning models, such as VAEs, aim to do this by separating out the factors that remain invariant under transformations of other factors [11]. It is therefore tempting to conclude that models that manage to show a high degree of disentanglement on the training set also capture the compositional structure of the world. If they did, they should be able to generalise to settings that present novel combination of the factors of variation. However, in our experiments, we observed that models that learned highly disentangled representations nevertheless failed at combinatorial generalisation, not only in the reconstruction space, but also in the latent space. These failures reproduced over several different datasets, different types of decoders, different loss functions and under a variety of different task settings.

Our interpretation of these results is that models manage to achieve a high degree of disentanglement by discovering factors that remain invariant over training examples and simply associating perceptual inputs with these factors. However, to capture the compositional structure of the world, models must additionally understand how factors interact to cause the perceptual input – that is, develop a good causal model of the world. We think the failure of models to form such a causal model is the reason behind why models succeeded at problems of combinatorial generalisation that do not involve an *interaction* between the left-out factors (Figure 6), but consistently failed at problems where these factors interacted (Figure 3). When generative factors do not interact, the model does not need to learn how the factors combine to determine the same part of the output space. Instead, it can simply map the value of each generative factor to a different location. It *appears* to solve combinatorial generalisation, but it does not understand how generative factors combine.

In summary, our work shows that the problem of combinatorial generalisation remains unsolved in both latent space and reconstruction space, even for highly disentangled models. By highlighting this limitation, we hope to inspire more work exploring new approaches that emulate this key capacity of human cognition, thus endowing models with a better understanding of the compositional structure of our world.

References

[1] Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

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

[3] Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. Understanding disentangling in β-VAE. *arXiv:1804.03599 [cs, stat]*, April 2018. URL http://arxiv.org/abs/1804.03599. arXiv: 1804.03599.

[4] Chomsky, N. *Aspects of the Theory of Syntax*, volume 11. MIT press, 2014.

[5] Duan, S., Matthey, L., Saraiva, A., Watters, N., Burgess, C. P., Lerchner, A., and Higgins, I. Unsupervised Model Selection for Variational Disentangled Representation Learning. *arXiv:1905.12614 [cs, stat]*, February 2020. URL http://arxiv.org/abs/1905.12614. arXiv: 1905.12614.

[6] Eastwood, C. and Williams, C. K. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, pp. 15, 2018.

[7] Fomin, V., Anmol, J., Desroziers, S., J., K., and Tejani, A. High-level library to help with training neural networks in pytorch. https://github.com/pytorch/ignite, 2020.

[8] Gondal, M. W., Wuthrich, M., Miladinovic, D., Locatello, F., Breidt, M., Volchkov, V., Akpo, J., Bachem, O., Schölkopf, B., and Bauer, S. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/d97d404b6119214e4a7018391195240a-Paper.pdf.

[9] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[10] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. β-VAE: Learning basic visual concepts with a constrained variational framework. pp. 13, 2017.

[11] Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., and Lerchner, A. Towards a Definition of Disentangled Representations. *arXiv:1812.02230 [cs, stat]*, December 2018. URL http://arxiv.org/abs/1812.02230. arXiv: 1812.02230.

[12] Higgins, I., Pal, A., Rusu, A. A., Matthey, L., Burgess, C. P., Pritzel, A., Botvinick, M., Blundell, C., and Lerchner, A. DARLA: Improving Zero-Shot Transfer in Reinforcement Learning. *arXiv:1707.08475 [cs, stat]*, June 2018. URL http://arxiv.org/abs/1707.08475. arXiv: 1707.08475.

[13] Higgins, I., Sonnerat, N., Matthey, L., Pal, A., Burgess, C. P., Bosnjak, M., Shanahan, M., Botvinick, M., Hassabis, D., and Lerchner, A. SCAN: Learning Hierarchical Compositional Visual Concepts. *arXiv:1707.03389 [cs, stat]*, June 2018. URL http://arxiv.org/abs/1707.03389. arXiv: 1707.03389.

[14] Hunter, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9 (3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

[15] Hyvarinen, A. and Morioka, H. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in Neural Information Processing Systems*, 29:3765–3773, 2016.

[16] Jeong, Y. and Song, H. O. Learning discrete and continuous factors of data via alternating disentanglement. In *International Conference on Machine Learning*, pp. 3091–3099. PMLR, 2019.

[17] Kim, H. and Mnih, A. Disentangling by Factorising. *arXiv:1802.05983 [cs, stat]*, July 2019. URL http://arxiv.org/abs/1802.05983. arXiv: 1802.05983.

[18] Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017. URL http://arxiv.org/abs/1412.6980. arXiv: 1412.6980.

[19] Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, December 2013. URL http://arxiv.org/abs/1312.6114. arXiv: 1312.6114.

[20] Klaus Greff, Aaron Klein, Martin Chovanec, Frank Hutter, and Jürgen Schmidhuber. The Sacred Infrastructure for Computational Research. In Katy Huff, David Lippa, Dillon Niederhut, and Pacer, M. (eds.), *Proceedings of the 16th Python in Science Conference*, pp. 49 – 56, 2017. doi: 10.25080/shinma-7f4c6e7-008.

[21] Klindt, D., Schott, L., Sharma, Y., Ustyuzhaninov, I., Brendel, W., Bethge, M., and Paiton, D. Towards nonlinear disentanglement in natural data with temporal sparse coding. *arXiv preprint arXiv:2007.10930*, 2020.

[22] Lin, Z., Thekumparampil, K., Fanti, G., and Oh, S. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans. In *International Conference on Machine Learning*, pp. 6127–6139. PMLR, 2020.

[23] Locatello, F., Poole, B., Rätsch, G., Schölkopf, B., Bachem, O., and Tschannen, M. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pp. 6348–6359. PMLR, 2020.

[24] Matthey, L., Higgins, I., Hassabis, D., and Lerchner, A. *dSprites: Disentanglement testing Sprites dataset*. 2017. URL https://github.com/deepmind/dsprites-dataset/.

[25] McCoy, R. T., Culbertson, J., Smolensky, P., and Legendre, G. Infinite use of finite means? evaluating the generalization of center embedding learned from an artificial grammar. 2021.

[26] Montero, M. L., Ludwig, C. J., Costa, R. P., Malhotra, G., and Bowers, J. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2020.

[27] Munkres, J. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[28] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

[29] Schölkopf, B., Locatello, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., and Bengio, Y. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

[30] Schott, L., von Kügelgen, J., Träuble, F., Gehler, P., Russell, C., Bethge, M., Schölkopf, B., Locatello, F., and Brendel, W. Visual representation learning does not generalize strongly within the same domain. *arXiv preprint arXiv:2107.08221*, 2021.

[31] Smolensky, P. *Connectionism, constituency, and the language of thought*. University of Colorado at Boulder, 1988.

[32] Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.

[33] van Steenkiste, S., Schmidhuber, J., Locatello, F., and Bachem, O. Are Disentangled Representations Helpful for Abstract Visual Reasoning? pp. 14, 2019.

[34] Von Humboldt, W., von Humboldt, W. F., et al. *Humboldt:'On language': On the diversity of human language construction and its influence on the mental development of the human species*. Cambridge University Press, 1999.

[35] Waskom, M. L. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60): 3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

[36] Watters, N., Matthey, L., Burgess, C. P., and Lerchner, A. Spatial Broadcast Decoder: A Simple Architecture for Learning Disentangled Representations in VAEs. *arXiv:1901.07017 [cs, stat]*, August 2019. URL http://arxiv.org/abs/1901.07017. arXiv: 1901.07017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Instead, we report probability distributions of relevant output variables.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No] We provide links to the Github repositories containing the datasets and their respective licenses.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]