Spatial Broadcast Decoder: A Simple Architecture for Learning Disentangled Representations in VAEs

Nicholas Watters, Loic Matthey, Christopher P. Burgess, Alexander Lerchner
DeepMind
London, United Kingdom
{nwatters, lmatthey, cpburgess, lerchner}@google.com

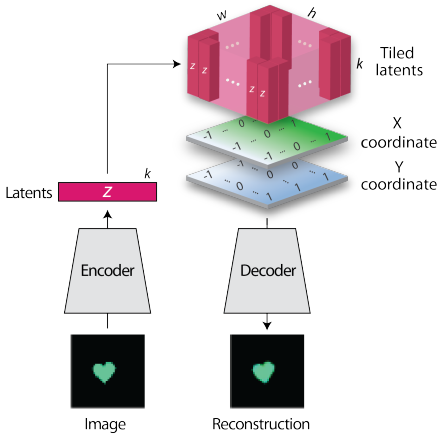
Abstract

We present a simple neural rendering architecture that helps variational autoencoders (VAEs) learn disentangled representations. Instead of the deconvolutional network typically used in the decoder of VAEs, we tile (broadcast) the latent vector across space, concatenate fixed X- and Y-"coordinate" channels, and apply a fully convolutional network with 1×1 stride. This provides an architectural prior for dissociating positional from non-positional features in the latent distribution of VAEs, yet without providing any explicit supervision to this effect. We show that this architecture, which we term the *Spatial Broadcast decoder*, improves disentangling, reconstruction accuracy, and generalization to held-out regions in data space. It provides a particularly dramatic benefit when applied to datasets with small objects. We also emphasize a method for visualizing learned latent spaces that helped us diagnose our models and may prove useful for others aiming to assess data representations. Finally, we show the Spatial Broadcast Decoder is complementary to state-of-the-art (SOTA) disentangling techniques and when incorporated improves their performance.

1 Introduction

Knowledge transfer and generalization are hallmarks of human intelligence. From grammatical generalization when learning a new language to visual generalization when interpreting a Picasso, humans have an extreme ability to recognize and apply learned patterns in new contexts. Current machine learning algorithms pale in contrast, suffering from overfitting, adversarial attacks, and domain specialization [Lake et al., 2016, Marcus, 2018]. We believe that one fruitful approach to improve generalization in machine learning is to learn compositional representations in an unsupervised manner. A compositional representation consists of components that can be recombined, and such recombination underlies generalization. For example, consider a pink elephant. With a representation that composes color and object independently, imagining a pink elephant is trivial. However, a pink elephant may not be within the scope of a representation that mixes color and object. Compositionality comes in a variety of flavors, including feature compositionality (e.g. pink elephant), multi-object compositionality (e.g. elephant next to a penguin), and relational compositionality (e.g. the smallest elephant). In this paper we will focus on feature compositionality.

Representations with feature compositionality are sometimes referred to as "disentangled" representations [Bengio et al., 2013]. However, there is as yet no consensus on the definition



Algorithm 1 Spatial Broadcast

Input: latents $\mathbf{z} \in \mathcal{R}^k$, width w, height h

Output: tiled latents $\mathbf{z}_{sb} \in \mathcal{R}^{h \times w \times (k+2)}$

1: $\mathbf{z}_b = \text{TILE}(\mathbf{z}, (h, w, 1))$

2: $\mathbf{x} = \text{LINSPACE}(-1, 1, w)$

3: $\mathbf{y} = \text{LINSPACE}(-1, 1, w)$

4: $\mathbf{x}_b, \mathbf{y}_b = \text{MESHGRID}(\mathbf{x}, \mathbf{y})$

5: $\mathbf{z}_{sb} = \text{CONCAT}([\mathbf{z}_b, \mathbf{x}_b, \mathbf{y}_b], \text{axis} = -1)$

6: **return** $\mathbf{z}_{sb}$

Figure 1: *(left)* Schematic of the Spatial Broadcast VAE. In the decoder, we broadcast (tile) a latent sample of size k to the image width w and height h, and concatenate two "coordinate" channels. This is then fed to an unstrided convolutional decoder. *(right)* Pseudo-code of the spatial broadcast operation, assuming access to a `numpy` / Tensorflow-like API.

of a disentangled representation [Higgins et al., 2018, Locatello et al., 2018, Higgins et al., 2017a]. In this paper, when evaluating disentangled representations we both employ standard metrics [Kim and Mnih, 2017, Chen et al., 2018, Locatello et al., 2018] and (in view of their limitations) emphasize visualization analysis.

Learning disentangled representations from images has recently garnered much attention. However, even in the best understood conditions, finding hyperparameters to robustly obtain such representations still proves quite challenging [Locatello et al., 2018]. Here we present a simple modification of the variational autoencoder (VAE) [Kingma and Welling, 2014, Rezende et al., 2014] decoder architecture that can dramatically improve both the robustness to hyperparameters and the quality of the learned representations in a variety of VAE-based models. We call this the Spatial Broadcast decoder, which we propose as an alternative to the standard MLP/deconvolutional network (which we refer to as the DeConv decoder). See Figure 1 for a schematic.

In this paper we show benefits of using the Spatial Broadcast decoder for image representation-learning, including:

- Improvements to both disentangling and reconstruction accuracy on datasets of simple objects.
- Complementarity to (and improvement of) state-of-the-art disentangling techniques.
- Particularly significant benefits when the objects in the dataset are small, a regime that is notoriously difficult for standard VAE architectures.
- Improved representational generalization to out-of-distribution test datasets involving both interpolation and extrapolation in latent space.

We also introduce a simple method for visualizing latent space geometry that we found helpful for gaining better insights when qualitatively assessing models and may prove useful to others interested in analyzing autoencoder representations

2 Spatial Broadcast Decoder

When modeling the distribution of images in a dataset with a variational autoencoder [Kingma and Welling, 2014, Rezende et al., 2014], standard architectures use an encoder consisting of a downsampling convolutional network followed by an MLP and a decoder consisting of an MLP followed by an upsampling deconvolutional network. The convolutional and deconvolutional networks share features across space, improving training efficiency for VAEs much like they do for all models that use image data.

However, while convolution surely lends some useful spatial inductive bias to the representations, a standard VAE learns highly entangled representations in an effort to represent the data as closely as possibly to its Gaussian prior (e.g. see Figure 7).

A number of new variations of the VAE objective have been developed to alleviate this problem, though all of them introduce additional hyperparameters [Higgins et al., 2017a, Burgess et al., 2018, Kim and Mnih, 2017, Chen et al., 2018]. Furthermore, a recent study found them to be extremely sensitive to these hyperparameters [Locatello et al., 2018].

Meanwhile, upsampling deconvolutional networks (like the one in the standard VAE's decoder) have been found to pose optimization challenges, such as producing checkerboard artifacts [Odena et al., 2016] and spatial discontinuities [Liu et al., 2018], effects that seem likely to raise problems for representation-learning in the VAE's latent space.

Intuitively, asking a deconvolutional network to render an object at a particular position is a tall order — the network's filters have no explicit spatial information, in other words they don't "know where they are." Hence the network must learn to propagate spatial asymmetries down from its highest layers and in from the spatial boundaries of the layers. This requires learning a complicated function, so optimization is difficult. To remedy this, in the Spatial Broadcast decoder we remove all upsampling deconvolutions from the network, instead tiling (broadcasting) the latent vector across space, appending fixed coordinate channels, then applying an unstrided fully convolutional network. This operation is depicted and described in Figure 1. With this architecture, rendering an object at a position becomes a very simple function (essentially just a thresholding operation in addition to the local convolutional features), though the network still has capacity to represent some more complex datasets (e.g. see Figure 5). Such simplicity of computation yields ease of optimization, and indeed we find that the Spatial Broadcast decoder greatly improves performance in a variety of VAE-based models.

Note that the Spatial Broadcast decoder does not provide any additional supervision to the generative model. The model must still learn to encode spatial information in its latent space in order to reconstruct accurately. The Spatial Broadcast decoder only allows the model to use the encoded spatial information in its latent space very efficiently.

In addition to better disentangling, the Spatial Broadcast VAE can on some datasets yield better reconstructions (see Figure 4), all with shallower networks and fewer parameters than a standard deconvolutional architecture. However it is worth noting that if the data does not benefit from having access to an absolute coordinate system, the Spatial Broadcast decoder could hurt. A standard DeConv decoder may in some cases more easily place patterns relative to each other or capture more extended spatial correlations. As we show in Section 4.2, we did not find this to be the case in the datasets we explored, but it is still a possible limitation of our model.

While the Spatial Broadcast decoder can be applied to any generative model that renders images from a vector representation, here we only explore its application to VAE models.

3 Related Work

Some of the ideas behind the Spatial Broadcast decoder have been explored extensively in many other bodies of work.

The idea of appending coordinate channels to convolutional layers has recently been highlighted (and named CoordConv) in the context of improving positional generalization [Liu et al., 2018]. However, the CoordConv technique had been used beforehand [Zhao et al., 2015, Liang et al., 2015, Watters et al., 2017, Wojna et al., 2017, Perez et al., 2017, Nash et al., 2017, Ulyanov et al., 2017] and its origin is unclear. The CoordConv VAE [Liu et al., 2018] incorporates CoordConv layers into an upsampling deconvolutional network in a VAE (which, as we show in Appendix D, does not yield representation-learning benefits comparable to those of the Spatial Broadcast decoder). To our knowledge no prior work has combined CoordConv with spatially tiling a generative model's representation as we do here.

Another line of work that has used coordinate channels extensively is language modeling. In this context, many models use fixed or learned position embeddings, combined in different ways to the input sequences to help compute translations depending on the sequence context in a differentiable manner [Gu et al., 2016, Vaswani et al., 2017, Gehring et al., 2017, Devlin et al., 2018].

We can draw parallels between the Spatial Broadcast decoder and generative models that learn "where" to write to an image [Jaderberg et al., 2015, Gregor et al., 2015, Reed et al., 2016], but in comparison these models render local image patches whereas we spatially tile a learned latent embedding and render an entire image. Work by Dorta et al. [2017] explored using a Laplacian Pyramid arrangement for VAEs, where some parts of the latent distributions would have global effects, whereas others would modify fine scale details about the reconstruction. Similar constraints on the extent of the spatial neighbourhood affected by the latent distribution has also been explored in Parmar et al. [2018]. Other types of generative models already use convolutional latent distributions [Finn et al., 2015, Levine et al., 2015, Eslami et al., 2018], unlike the flat vectors we consider here. In this case the tiling operation is not necessary, but adding coordinate channels might be of help.

4 Results

We present here a select subset of our assessment of the Spatial Broadcast decoder. Interested readers can refer to Appendices A-G for a more thorough set of observations and comparisons.

4.1 Performance on colored sprites

The Spatial Broadcast decoder was designed with object-feature representations in mind, hence to initially showcase its performance we use a dataset of simple objects: colored 2-dimensional sprites. This dataset is described in the literature [Burgess et al., 2018] as a colored version of the dSprites dataset [Matthey et al., 2017]. One advantage of the colored sprites dataset is it has known factors of variation, of which there are 8: X-position, Y-position, Size, Shape, Angle, and three-dimensional Color. Thus we can evaluate disentangling performance with metrics that rely on ground truth factors of variation [Chen et al., 2018, Kim and Mnih, 2017].

To quantitatively evaluate disentangling, we focus primarily on the Mutual Information Gap (MIG) metric [Chen et al., 2018]. The MIG metric is defined by first computing the mutual information matrix between the latent distribution means and ground truth factors,

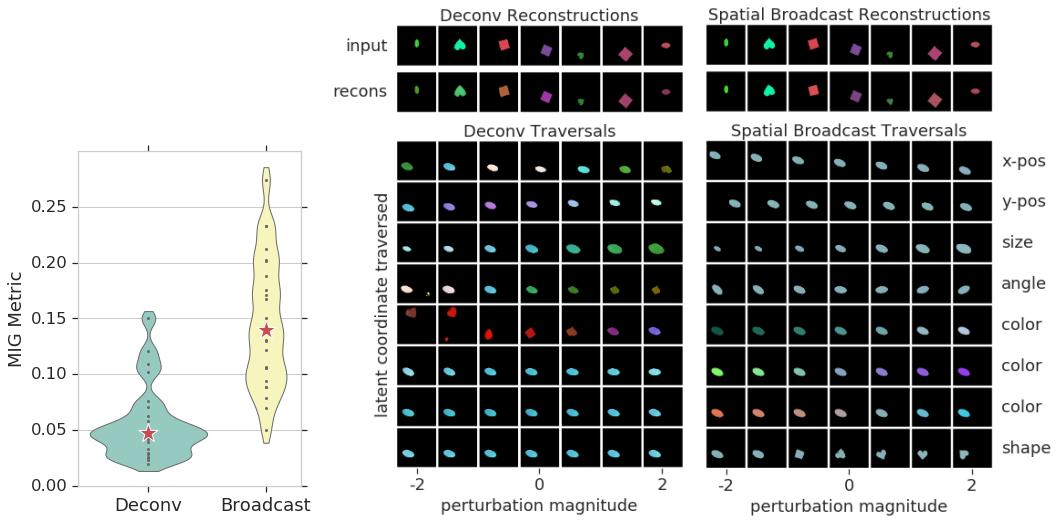


Figure 2: Comparing Deconv to Spatial Broadcast decoder in a VAE. *(left)* MIG results, showing a Spatial Broadcast VAE achieves higher (better) scores than a DeConv VAE. Stars are median MIG values and the seeds used for the traversals on the right. *(middle)* DeConv VAE reconstructions and latent space traversals. Traversals are generated around a seed point in latent space by reconstructing a sweep from -2 to +2 for each coordinate while keeping all other coordinates constant. The traversal shows an entangled representation in this model. *(right)* Spatial Broadcast VAE reconstructions and traversal. The traversal is well-disentangled and aligned with generative factors, as indicated by the labels on the right (which were attributed by visual inspection). While all models were trained with 10 latent coordinates, only the 8 lowest-variance ones are shown in the traversals (the remainder are non-coding coordinates).

then computing the difference between the highest and second-highest elements for each ground truth factor (i.e. the gap), then finally averaging these values over all factors.

Despite significant shortcomings (see Section 4.4), we did find the MIG metric overall can a helpful tool for evaluating representational quality, though should by no means be trusted blindly. We did also use the FactorVAE metric [Kim and Mnih, 2017], yet found it to be less consistent then MIG (we report these results in Appendix F).

In Figure 2 we compare a standard DeConv VAE (a VAE with an MLP + deconvolutional network decoder) to a Spatial Broadcast VAE (a VAE with the Spatial Broadcast decoder). We see that the Spatial Broadcast VAE outperforms the DeConv VAE both in terms of the MIG metric and traversal visualizations. The Spatial Broadcast VAE traversals clearly show all 8 factors represented in separate latent factors, seemingly on par with more complicated state-of-the-art methods on this dataset [Burgess et al., 2018]. The hyperparameters for the models in Figure 2 were chosen over a large sweep to minimize the model's error, not chosen for any disentangling properties explicitly. See Appendix F for more details about hyperparameter choices and sensitivity.

The Spatial Broadcast decoder is complementary to existing disentangling VAE techniques, hence improves not only a vanilla VAE but SOTA models as well. To demonstrate this, we consider two recently developed models, FactorVAE [Kim and Mnih, 2017] and β-VAE [Higgins et al., 2017a].

Figure 3 shows the Spatial Broadcast decoder improving the disentangling of FactorVAE in terms of both the MIG metric and traversal visualizations. See Appendix G for further results to this effect.

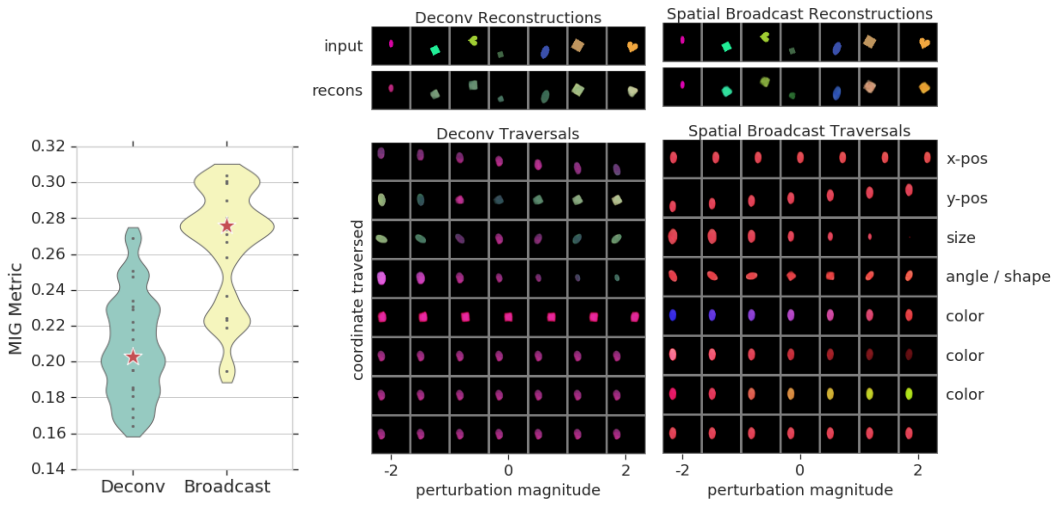


Figure 3: Comparing Deconv to Spatial Broadcast decoder in a FactorVAE. *(left)* MIG results, showing a Spatial Broadcast FactorVAE acheives higher (better) scores than a DeConv FactorVAE. Stars are median MIG values and the seeds used for the traversals on the right. *(middle)* DeConv FactorVAE reconstructions and entangled latent space traversals. *(right)* Spatial Broadcast FactorVAE reconstructions and traversal. The traversal is well-disentangled. As in Figure 2, only the most relevant 8 of each model's 10 latent coordinates are shown in the traversals.

Figure 4 shows performance of a β-VAE with and without the Spatial Broadcast decoder for a range of values of β. Not only does the Spatial Broadcast decoder improve disentangling, it also yields a lower rate-distortion curve, hence learns to more efficiently represent the data than the same model with a DeConv decoder.

4.2 Datasets without positional variation

The colored sprites dataset discussed in Section 4.1 seems particularly well-suited for the Spatial Broadcast decoder because X- and Y-position are factors of variation. However, we also evaluate the architecture on datasets that have no positional factors of variation: Chairs and 3D Object-in-Room datasets [Aubry et al., 2014, Kim and Mnih, 2017]. In the latter, the factors of variation are highly non-local, affecting multiple regions spanning nearly the entire image. We find that on both datasets a Spatial Broadcast VAE learns representations that look very well-disentangled, seemingly as well as SOTA methods on these datasets and without any modification of the standard VAE objective [Kim and Mnih, 2017, Higgins et al., 2017a]. See Figure 5 for results (and supplementary Figure 12 for additional traversals).

While this shows that the Spatial Broadcast decoder does not hurt when used on datasets without positional variation, it may seem unlikely that it would help in this context. However, we show in Appendix G that it actually can help to some extent on such datasets. We attribute this to its using a shallower network and no upsampling deconvolutions (which have been observed to cause optimization difficulties in a variety of settings [Liu et al., 2018, Odena et al., 2016]).

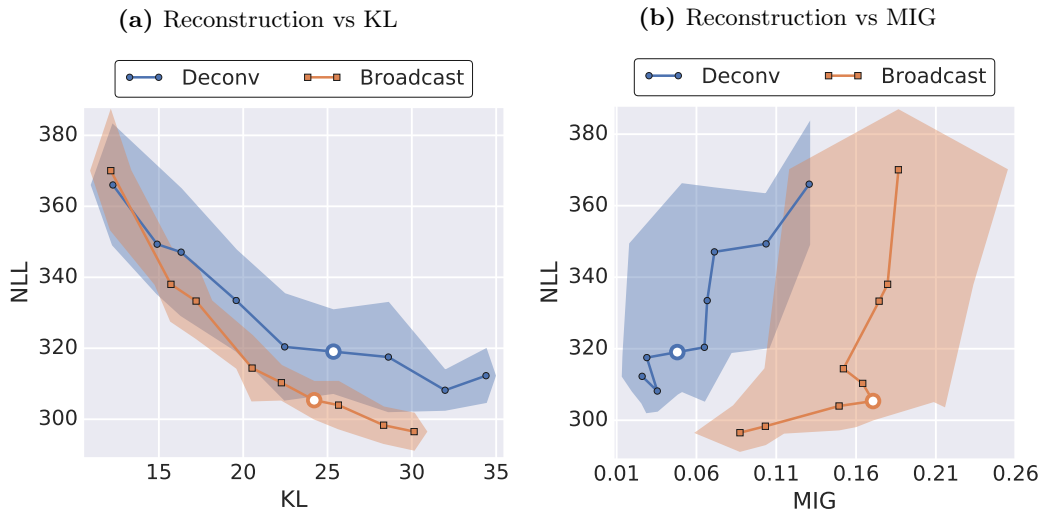


Figure 4: Rate-distortion proxy curves. We swept β log-linearly from 0.4 to 5.4 and for each value trained 10 replicas each of Deconv β-VAE (blue) and Spatial Broadcast β-VAE (orange) on colored sprites. The dots show the mean over these replicas for each β, and the shaded region shows the hull of one standard deviation. White dots indicate $\beta = 1$. **(a)** Reconstruction (Negative Log-Likelihood, NLL) vs KL. $\beta < 1$ yields low NLL and high KL (bottom-right of figure), whereas $\beta > 1$ yields high NLL and low KL (top-left of figure). See Alemi et al. [2017] for details. Spatial Broadcast β-VAE shows a better rate-distortion curve than Deconv β-VAE. **(b)** Reconstruction vs MIG metric. $\beta < 1$ correspond to lower NLL and low MIG regions (bottom-left of figure), and $\beta > 1$ values correspond to high NLL and high MIG scores (towards top-right of figure). Spatial Broadcast β-VAE is better disentangled (higher MIG scores) than Deconv β-VAE.

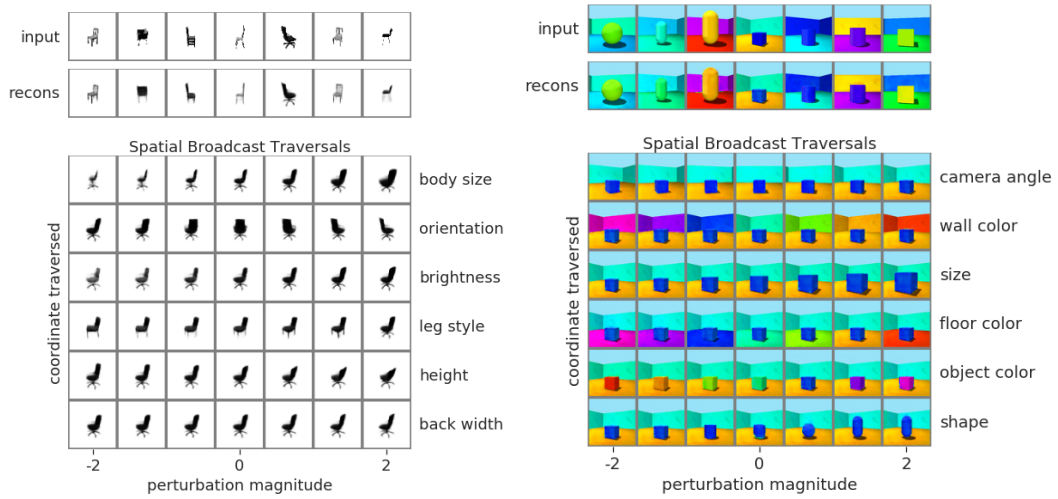


Figure 5: Traversals for datasets with no positional variation. A Spatial Broadcast VAE shows good reconstructions and disentangling on the Chairs dataset [Aubry et al., 2014] and the 3D Object-in-Room dataset [Kim and Mnih, 2017]. As in Figures 2 and 3, the models have 10 latent coordinates, though in these traversals the 4 non-coding ones are omitted.

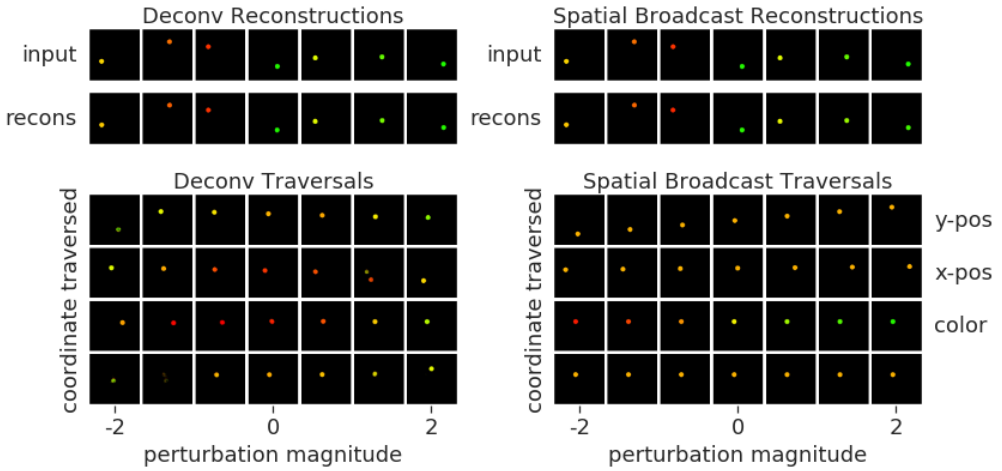


Figure 6: Disentangling small objects. A DeConv VAE *(left)* learns a highly entangled and discontinuous representation of this dataset of small hue-varying circles, while a Spatial Broadcast VAE *(right)* disentangles the dataset well. Here traversals of only the most relevant 4 of the models' 10 latent coordinates are shown (the other latent coordinates are non-coding).

4.3 Datasets with small objects

In exploring datasets with objects varying in position, we often find a (standard) DeConv VAE learns a representation that is discontinuous with respect to object position. This effect is amplified as the size of the object decreases. This makes sense, because the pressure for a VAE to represent position continuously comes from the fact that an object and a position-perturbed version of itself overlap in pixel space (hence it is economical for the VAE to map noise in its latent samples to local translations of an object). However, as an object's size decreases, this pixel overlap decreases, hence the pressure for a VAE to represent position continuously weakens.

In this small-object regime the Spatial Broadcast decoder's architectural bias towards representing positional variation continuously proves extremely useful. We see this in Figure 6 and supplementary Figure 10.

4.4 Latent space geometry visualization

Evaluating the quality of a representation can be challenging and time-consuming. While a number of metrics have been proposed to quantify disentangling, many of them have serious shortcomings and there is as yet no consensus in the literature which to use [Locatello et al., 2018]. We believe it is impossible to quantify how good a representation is with a single scalar, because there is a fundamental trade-off between how much information a representation contains and how well-structured the representation is. This has been noted by others in the disentangling literature [Ridgeway and Mozer, 2018, Eastwood and Williams, 2018]. This disentangling-distortion trade-off is a recapitulation of the rate-distortion trade-off [Alemi et al., 2017] and can be seen first-hand in Figure 4. We would like representations that both reconstruct well and disentangle well, but exactly how to balance these two factors is a matter of subjective preference (and surely depends on the dataset). Any scalar disentangling metric will implicitly favor some arbitrary disentangling-reconstruction potential.

In addition to this unavoidable limitation of disentangling metrics, we found that the

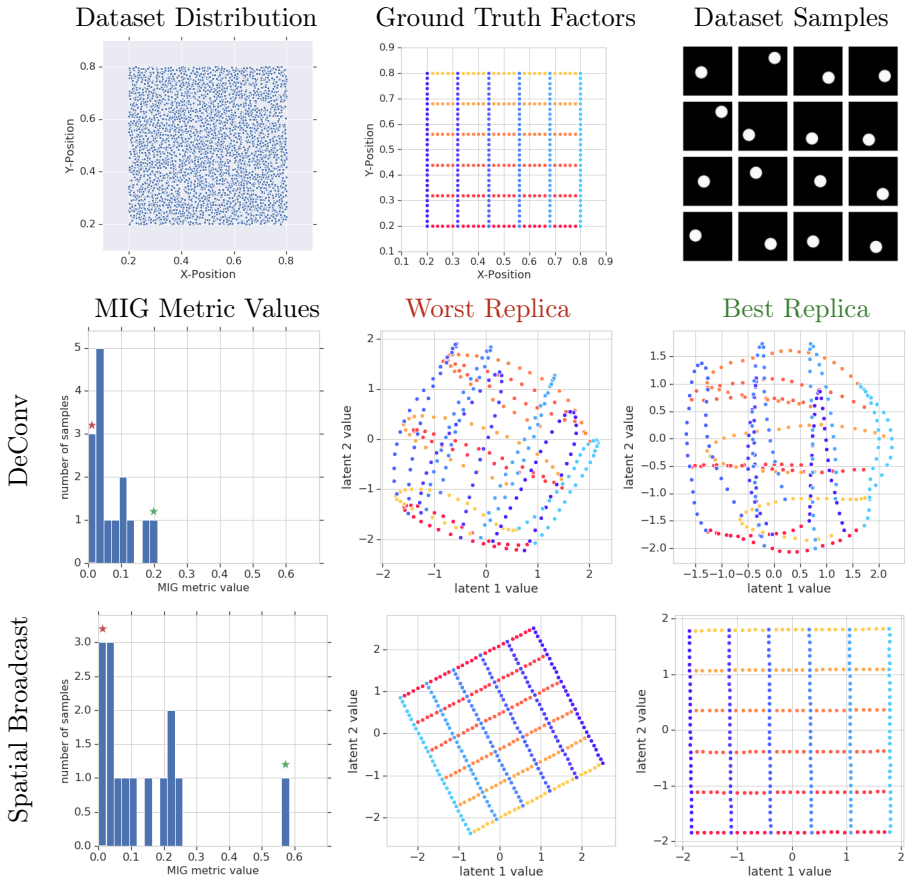


Figure 7: Latent space geometry analysis of x-y dataset. Comparison of DeConv and Spatial Broadcast decoders in a standard VAE on a dataset of circles varying in position. On the top row we see: *(left)* the data generative factor distribution uniform in X- and Y-position, *(middle)* a grid of points in generative factor space spanning the data distribution, and *(right)* 16 sample images from the dataset. The next two rows show a analysis of the DeConv VAE and Spatial Broadcast VAE on this dataset. In each we see a histogram of MIG metric values over 15 independent replicas and a latent space geometry visualization for the replica with the worst MIG and the replica with the best MIG (highlighted in the MIG histograms by the colored stars). These geometry plots show the embedding of the ground truth factor grid in latent space with respect to the two (out of 10) latent components with the smallest mean variance, i.e. the most informative latent components. The Deconv decoder gives rise to highly entangled latent spaces, while the Spatial Broadcast decoder learns embeddings that are extremely well-disentangled (in fact, nearly linear), even in the worst performing replica. Note that the MIG does not capture this contrast because it is very sensitive to rotation in the latent space.

MIG metric (while perhaps more accurate than other existing metrics) does not capture the intuitive notion of disentangling because:

- It depends on a choice of basis for the ground truth factors, and heavily penalizes rotation of the representation with respect to this basis. Yet it is often unclear what the correct basis for the ground truth factors is (e.g. RGB vs HSV vs HSL). For example,

see the bottom row of Figure 7.

- It is invariant to a folding of the representation space, as long as the folds align with the axes of variation. See the middle row of Figure 7 for an example of a double-fold in the latent space which isn't penalized by the MIG metric.

Due to the subjective nature of disentangling and the difficulty in defining appropriate metrics, we put heavy emphasis on latent space visualization as a means for representational analysis. Latent space traversals have been extensively used in the literature and can be quite revealing [Higgins et al., 2017a,b]. However, in our experience, traversals suffer two shortcomings:

- Some latent space entanglement can be difficult for the eye to perceive in traversals. For example, a slight change in brightness in a latent traversal that represents changing position can easily go unnoticed.

- Traversals only represent the latent space geometry around one point in space, and cross-referencing corresponding traversals between multiple points is quite time-consuming.

Consequently, we caution the reader against relying too heavily on traversals when evaluating latent space geometry. We propose an additional method for analyzing latent spaces, which we found very useful in our research. This is possible when there are known generative factors in the dataset and aims to directly view the embedding of generative factor space in the model's latent space: We plot in latent space the locations corresponding to a grid of points in generative factor space. While this is can only visualize the latent embedding of a 2-dimensional subspace of generative factor space, it can be very revealing of the latent space geometry.

We showcase this analysis method in Figure 7 on a dataset of circles varying in X- and Y-position. (See Appendix G for similar analyses on many more datasets.) We compare the representations learned by a DeConv VAE and a Spatial Broadcast VAE. This reveals a stark contrast: The Spatial Broadcast latent geometry is a near-perfect Euclidean transformation, while the Deconv decoder model's representations are very entangled.

The MIG metric does not capture this difference well: It gives a score near zero to a Spatial Broadcast model with a latent space rotated with respect to the generative factors and a greater score to a highly entangled Deconv model. In practice we care primarily about compositionality of (and hence disentangling of) subspaces of generative factor space and not about axis-alignment within these subspaces. For example, rotation within X-Y position subspace is acceptable, whereas rotation in position-color space is not. This naturally poses a great challenge for both designing disentangling metrics and formally defining disentangling [cf. Higgins et al., 2018]. We believe that additional structure over the factors of variation can be inferred from temporal correlations, the structure of a reinforcement learning agent's action space, and other effects not necessarily captured by a static-image dataset. Nonetheless, inductive biases like the Spatial Broadcast decoder can be explored in the context of static images, as they will likely also help representation learning in more complete contexts.

4.5 Datasets with dependent factors

Our primary motivation for unsupervised representation learning is the prospect of improved generalization and transfer, as discussed in Section 1. Consequently, we would like a model to learn compositional representations that can generalize to new feature combinations. We

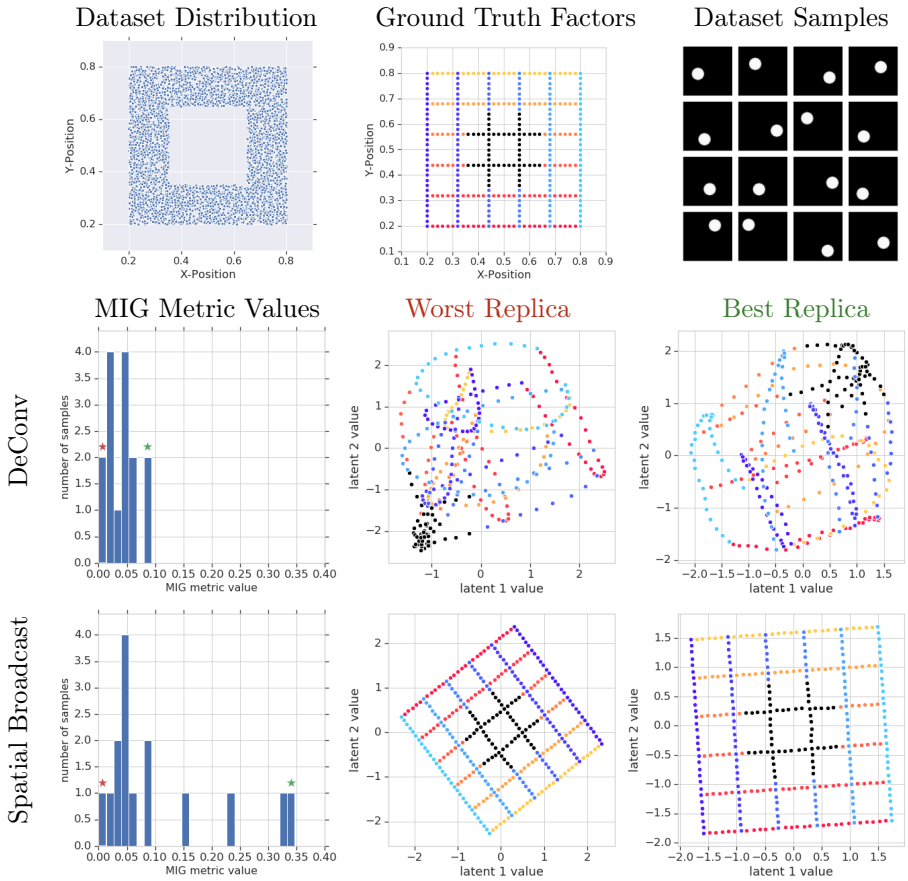


Figure 8: Latent space geometry analysis of x-y dataset with dependent factors. Analogous to Figure 7, except the dataset has a large held-out hole in generative factor space (see dataset distribution in top-left), hence the generative factors are not independent. For the latent geometry visualizations, we do evaluate the representation of images in this held-out hole, which are shown as black dots. This tests for generalization, namely extrapolation in pixel space and interpolation in generative factor space. The Spatial Broadcast VAE generalizes extremely well, again with representations looking nearly linear, even through the held-out hole. In contrast, the DeConv VAE learns highly entangled representations. As in Figure 7, the MIG does not adequately reflect this difference because of its sensitivity to rotation.

explore this in a controlled setting by holding out a region of generative factor space from the training dataset, then evaluating the representation in this held-out region. Figure 8 shows results to this effect, comparing the Spatial Broadcast VAE to a DeConv VAE on a dataset of circles with a held-out region in the middle of X-Y position-space (so the model sees no circles in the middle of the image, indicated by the black dots in the second column). The Spatial Broadcast VAE generalizes almost perfectly in this case, appearing unaffected by the fact that the data generative factors are no longer independent and extrapolating nearly linearly throughout the held-out region. In contrast, the DeConv VAE's latent space is highly entangled, even more so than in the case with independent factors of Figure 7. See Appendix G for more analyses of generalization on other datasets.

From the perspective of density modeling, disentangling of the Spatial Broadcast VAE may seem undesirable in this case because it allocates high probability in latent space to a region of low (here, zero) probability in the dataset. However, from the perspective of representation learning and compositional features, such generalization is a highly desirable property.

5 Conclusion

Here we present and analyze the Spatial Broadcast decoder in the context of Variational Autoencoders. We demonstrate that it improves learned latent representations, most dramatically for datasets with objects varying in position. It also improves generalization in latent space and can be incorporated into SOTA models to boost their performance in terms of both disentangling and reconstruction accuracy. We rigorously analyze the contribution of the Spatial Broadcast decoder on a wide variety of datasets and models using a range of visualizations and metrics. We hope that this analysis has provided the reader with an intuitive understanding of how and why it improves learned representations.

We believe that learning compositional representations is an important ingredient for flexibility and generalization in many contexts, from supervised learning to reinforcement learning, and the Spatial Broadcast decoder is one step towards robust compositional visual representation learning.

Acknowledgments

We thank Irina Higgins, Danilo Rezende, Matt Botvinick, and Yotam Doron for helpful discussions and insights.

References

Alexander A. Alemi, Ben Poole, Ian Fischer, Joshua V. Dillon, Rif A. Saurous, and Kevin Murphy. An information-theoretic analysis of deep latent-variable models. *CoRR*, abs/1711.00464, 2017. URL http://arxiv.org/abs/1711.00464.

M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014.

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

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

Tian Qi Chen, Xuechen Li, Roger B. Grosse, and David K. Duvenaud. Isolating sources of disentanglement in variational autoencoders. *CoRR*, abs/1802.04942, 2018. URL http://arxiv.org/abs/1802.04942.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Garoe Dorta, Sara Vicente, Lourdes Agapito, Neill D.F. Campbell, Simon Prince, and Ivor Simpson. Laplacian pyramid of conditional variational autoencoders. In *Proceedings of the 14th European Conference on Visual Media Production (CVMP 2017)*, CVMP 2017, pages 7:1–7:9, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5329-8. doi: 10.1145/3150165.3150172. URL http://doi.acm.org/10.1145/3150165.3150172.

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

S. M. Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, Andrei A. Rusu, Ivo Danihelka, Karol Gregor, David P. Reichert, Lars Buesing, Theophane Weber, Oriol Vinyals, Dan Rosenbaum, Neil Rabinowitz, Helen King, Chloe Hillier, Matt Botvinick, Daan Wierstra, Koray Kavukcuoglu, and Demis Hassabis. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018. ISSN 0036-8075. doi: 10.1126/science.aar6170. URL http://science.sciencemag.org/content/360/6394/1204.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Learning visual feature spaces for robotic manipulation with deep spatial autoencoders. *CoRR*, abs/1509.06113, 2015. URL http://arxiv.org/abs/1509.06113.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122, 2017. URL http://arxiv.org/abs/1705.03122.

Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. DRAW: A recurrent neural network for image generation. *CoRR*, abs/1502.04623, 2015. URL http://arxiv.org/abs/1502.04623.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. Incorporating copying mechanism in sequence-to-sequence learning. *CoRR*, abs/1603.06393, 2016. URL http://arxiv.org/abs/1603.06393.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2017a.

Irina Higgins, Nicolas Sonnerat, Loic Matthey, Arka Pal, Christopher P Burgess, Matthew Botvinick, Demis Hassabis, and Alexander Lerchner. Scan: learning abstract hierarchical compositional visual concepts. *arXiv preprint arXiv:1707.03389*, 2017b.

Irina Higgins, David Amos, David Pfau, Sébastien Racanière, Loïc Matthey, Danilo J. Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *CoRR*, abs/1812.02230, 2018. URL http://arxiv.org/abs/1812.02230.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *CoRR*, abs/1506.02025, 2015. URL http://arxiv.org/abs/1506.02025.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arxiv*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.

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

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *CoRR*, abs/1504.00702, 2015. URL http://arxiv.org/abs/1504.00702.

Xiaodan Liang, Yunchao Wei, Xiaohui Shen, Jianchao Yang, Liang Lin, and Shuicheng Yan. Proposal-free network for instance-level object segmentation. *CoRR*, abs/1509.02636, 2015. URL http://arxiv.org/abs/1509.02636.

Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *CoRR*, abs/1807.03247, 2018. URL http://arxiv.org/abs/1807.03247.

Francesco Locatello, Stefan Bauer, Mario Lucic, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *arXiv preprint arXiv:1811.12359*, 2018.

Gary Marcus. Deep learning: A critical appraisal. *CoRR*, abs/1801.00631, 2018. URL http://arxiv.org/abs/1801.00631.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset, 2017. URL https://github.com/deepmind/dsprites-dataset/.

Charlie Nash, Ali Eslami, Chris Burgess, Irina Higgins, Daniel Zoran, Theophane Weber, and Peter Battaglia. The multi-entity variational autoencoder. *NIPS Workshops*, 2017.

Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, and Alexander Ku. Image transformer. *CoRR*, abs/1802.05751, 2018. URL http://arxiv.org/abs/1802.05751.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. *CoRR*, abs/1709.07871, 2017. URL http://arxiv.org/abs/1709.07871.

Scott E. Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. *CoRR*, abs/1610.02454, 2016. URL http://arxiv.org/abs/1610.02454.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, 32(2):1278–1286, 2014.

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

Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Deep image prior. *CoRR*, abs/1711.10925, 2017. URL http://arxiv.org/abs/1711.10925.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *Advances in Neural Information Processing Systems 30*, pages 4539–4547. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7040-visual-interaction-networks-learning-a-physics-simulator-from-video.pdf.

Nicholas Watters, Loic Matthey, Sebastian Borgeaud, Rishabh Kabra, and Alexander Lerchner. Spriteworld: A flexible, configurable reinforcement learning environment. https://github.com/deepmind/spriteworld/, 2019. URL https://github.com/deepmind/spriteworld/.

Zbigniew Wojna, Alexander N. Gorban, Dar-Shyang Lee, Kevin Murphy, Qian Yu, Yeqing Li, and Julian Ibarz. Attention-based extraction of structured information from street view imagery. *CoRR*, abs/1704.03549, 2017. URL http://arxiv.org/abs/1704.03549.

Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann LeCun. Stacked what-where auto-encoders. *arXiv*, abs/1506.02351, 2015. URL https://arxiv.org/abs/1506.02351.

Supplementary material

A Experiment Details

For all VAE models we used a Bernoulli decoder distribution, parametrized by its logits. It is with respect to this distribution that the reconstruction error (negative log likelihood) was computed. This could accomodate our datasets, since they were normalized to have pixel values in $[0, 1]$. We also explored using a Gaussian distribution with fixed variance (for which the NLL is equivalent to scaled MSE), and found that this produces qualitatively similar results and in fact improves stability. Hence while a Bernoulli distribution usually works, we suggest the reader wishing to experiment with these models starts with a Gaussian decoder distribution with mean parameterized by the decoder network output and variance constant at 0.3.

In all networks we used ReLU activations, weights initialized by a truncated normal (see [Ioffe and Szegedy, 2015]), and biases initialized to zero. We use no other neural network tricks (no BatchNorm or dropout), and all models were trained with the Adam optimizer [Kingma and Ba, 2015]. See below for learning rate details.

A.1 VAE Hyperparameters

For all VAE models except β-VAE (shown only in Figure 4), we use a standard VAE loss, namely with a KL term coefficient $\beta = 1$. For FactorVAE we also use $\beta = 1$, as in Kim and Mnih [2017].

For the VAE, β-VAE, CoordConv VAE and ablation study we used the network parameters in Table 1. We note that, while the Spatial Broadcast decoder uses fewer parameters than the DeConv decoder, it does require about 50% more memory to store the weights. However, for the 3D Object-in-Room dataset we included three additional deconv layers in the Spatial Broadcast decoder (without these additional layers the decoder was not powerful enough to give good reconstructions on that dataset).

All of these models were trained using a learning rate of $3 \cdot 10^{-4}$ on with batch size 16. All convolutional and deconvolutional layers have "same" padding, i.e. have zero-padded input so that the output shape is `input_shape` $\times$ `stride` in the case of convolution and `input_shape/stride` in the case of deconvolution.

Encoder
FC(2×10) Output LogNormal
FC(256)
Conv(k=4, s=2, c=64)
Conv(k=4, s=2, c=64)
Conv(k=4, s=2, c=64)
Conv(k=4, s=2, c=64)
Input Image [$64 \times 64 \times C$]

DeConv Decoder
Output Logits
DeConv(k=4, s=2, c=C)
DeConv(k=4, s=2, c=64)
DeConv(k=4, s=2, c=64)
DeConv(k=4, s=2, c=64)
DeConv(k=4, s=2, c=64)
reshape($2 \times 2 \times 64$)
FC(256)
Input Vector [10]

Broadcast Decoder
Output Logits
Conv(k=4, s=1, c=C)
Conv(k=4, s=1, c=64)
Conv(k=4, s=1, c=64)
append coord channels
tile($64 \times 64 \times 10$)
Input Vector [10]

Table 1: Network architectures for Vanilla VAE, β-VAE, CoordConv VAE and ablation study. The numbers of layers were selected to minimize the ELBO loss of a VAE on the colored sprites data (see Appendix F). Note that for 3D Object-in-Room, we include three additional convolutional layers in the spatial broadcast decoder. Here C refers to the number of channels of the input image, either 1 or 3 depending on whether the dataset has color.

A.2 FactorVAE

For the FactorVAE model, we used the hyperparameters described in the FactorVAE paper [Kim and Mnih, 2017]. Those network parameters are reiterated in Table 2. Note that the Spatial Broadcast parameters are the same as for the other models in Table 1. For the optimization hyperparameters we used $\gamma = 35$, a learning rate of 10^{-4} for the VAE updates, a learning rate of $2 \cdot 10^{-5}$ for the discriminator updates, and batch size 32. These parameters generally gave stable results.

However, when training the FactorVAE model on colored sprites we encountered instability during training. We subsequently did a number of hyperparameter sweeps attempting to improve stability, but to no avail. Ultimately, we used the hyperparameters in Table 2, though even with limited training steps (see Appendix Section A.4) about 15% of seeds diverged before training completed for both Spatial Broadcast and Deconv decoder.

Encoder
FC(2×10) Output LogNormal
FC(256)
Conv(k=4, s=2, c=64)
Conv(k=4, s=2, c=64)
Conv(k=4, s=2, c=32)
Conv(k=4, s=2, c=32)
Input Image [$64 \times 64 \times C$]

DeConv Decoder
Output Logits
DeConv(k=4, s=2, c=C)
DeConv(k=4, s=2, c=32)
DeConv(k=4, s=2, c=32)
DeConv(k=4, s=2, c=64)
DeConv(k=4, s=2, c=64)
reshape($2 \times 2 \times 64$)
FC(256)
Input Vector [10]

Broadcast Decoder
Output Logits
Conv(k=4, s=1, c=C)
Conv(k=4, s=1, c=64)
Conv(k=4, s=1, c=64)
append coord channels
tile($64 \times 64 \times 10$
Input Vector [10]

Table 2: Network architectures for FactorVAE. The encoder and DeConv decoder architectures are as described in the FactorVAE paper [Kim and Mnih, 2017], and the Spatial Broadcast decoder architecture is the same as for the other models (Table 1.)

A.3 Datasets

All datasets were rendered in images of size 64×64 and normalized to $[0, 1]$.

Colored Sprites:

For this dataset, we use the binary dsprites dataset open-sourced in [Matthey et al., 2017], but multiplied by colors sampled in HSV space uniformly within the region $H \in [0.0, 1.0]$, $S \in [0.3, 0.7]$, $V \in [0.3, 0.7]$. Sans color, there are 737,280 images in this dataset. However, we sample the colors online from a continuous distribution, effectively making the dataset size infinite.

Chairs:

This dataset is open-sourced in [Aubry et al., 2014]. This dataset, unlike all others we use, has only a single channel in its images. It contains 86,366 images.

3D Object-in-Room:

This dataset was used extensively in the FactorVAE paper [Kim and Mnih, 2017]. It consists of an object in a room and has 6 factors of variation: Camera angle, object size, object shape, object color, wall color, and floor color. The colors are sampled uniformly from a continuous set of hues in the range $[0.0, 0.9]$. This dataset contains 480,000 images, procedurally generated as all combinations of 10 floor hues, 10 wall hues, 10 object hues, 8 object sizes, 4 object shapes, and 15 camera angles.

Circles Datasets:

To more thoroughly explore datasets with a variety of distributions, factors of variation, and held-out test sets we created our own datasets using the Spriteworld environment (Watters et al. [2019], available from `https://github.com/deepmind/spriteworld/`). We did not use the Reinforcement Learning aspects of Spriteworld, but used Spriteworld's factor distribution library and renderer, so we created "tasks" with maximum episode length 1 (so the environment resets every step) and used a dummy agent to record image observations. All images were rendered with an anti-aliasing factor of 5. For the results in this paper, we used an earlier version of Spriteworld which used a PyGame renderer instead of the PIL renderer currently in the library, but this choice of renderer does not affect our results. We used this Spriteworld-based dataset generator for the datasets in Section 4.4. In these datasets we control subsets of the following factors of variation: X-position, Y-position, Size, Color. We generated five datasets in this way, which we call `X-Y`, `X-H`, `R-G`, `X-Y-H Small`, and `X-Y-H Tiny`, and can be seen in Figures (Fig 13), (Fig 16), (Fig 19), (Fig 6), and (Fig 10) respectively.

Table 3 shows the values of these factors for each dataset. Note that for some datasets we define the color distribution in RGB space, and for others we define it in HSV space.

To create the datasets with dependent factors, we hold out one quarter of the dataset (the intersection of half of the ranges of each of the two factors), either centered within the data distribution or in the corner.

For each dataset we generate 500,000 randomly sampled training images.

	X	Y	Size	(H, S, V)	(R, G, B)
X-Y	[0.2, 0.8]	[0.2, 0.8]	0.2	N/A	(1.0, 1.0, 1.0)
X-H	[0.2, 0.8]	0.5	0.3	([0.2, 0.8], 1.0, 1.0)	N/A
R-G	0.5	0.5	0.5	N/A	([0.4, 0.8], [0.4, 0.8], 1.0)
X-Y-H Small	[0.2, 0.8]	[0.2, 0.8]	0.1	([0.2, 0.8], 1.0, 1.0)	N/A
X-Y-H Tiny	[0.2, 0.8]	[0.2, 0.8]	0.075	([0.2, 0.8], 1.0, 1.0)	N/A

Table 3: Circles datasets details. Each row represents a different dataset, and each column shows a generative factor's range for the datasets.

A.4 Training Steps

The number of training steps for each model on each dataset can be found in Table 4. In general, for each dataset we used enough training steps so that all models converged. Note that while the training iterations is different for FactorVAE than for the other models on colored sprites (due to instability of FactorVAE), this has no bearing on our results because we do not compare across models. We only compare across decoder architectures, and we always used the same training steps for both DeConv and Spatial Broadcast decoders within each model.

B Ablation Study

One aspect of the Spatial Broadcast decoder is the concatenation of constant coordinate channels to its tiled input latent vector. While our justification of its performance emphasizes

	VAE	FactorVAE
Colored Sprites	$1.5 \cdot 10^6$	$1 \cdot 10^6$
Chairs	$7 \cdot 10^5$	N/A
3D Objects	$2.5 \cdot 10^6$	N/A
Circles	$5 \cdot 10^5$	$5 \cdot 10^5$

Table 4: Number of training steps for each model on each dataset. Here the VAE column includes β-VAE, Deconv VAE, Spatial Broadcast VAE, CoordConv VAE, and the ablation study.

the simplicity of computation it affords, it may seem possible that the coordinate channels are only used to provide positional information and the simplicity of this positional information (linear meshgrid) is irrelevant. Here we perform an ablation study to demonstrate that this is not the case; the organization of the coordinate channels is important. For this experiment, we randomly permute the coordinate channels through space. Specifically, we take the $[h \times w \times 2]$-shape coordinate channels and randomly permute the $h \cdot w$ entries. We keep each (i, j) pair together to ensure that after the shuffling each location does still have a unique pair of coordinates in the coordinate channels. Importantly, we only shuffle the coordinate channels once, then keep them constant throughout training.

Figure 9 shows reconstructions and traversals for two replicas (with different shuffled coordinate channels). Both disentangling and reconstruction accuracy are significantly reduced.

C Disentangling Tiny Objects

In the dataset of small colored circles shown in Figure 6 the circle diameter is 0.1 times the frame-width. We also generated a dataset with circles 0.075 times the frame-width, and Figure 10 shows similar results on this dataset (albeit more difficult for the eye to make out). We were surprised to see disentangling of such tiny objects and have not explored the lower object size limit for disentangling with the Spatial Broadcast decoder.

D CoordConv VAE

CoordConv VAE [Liu et al., 2018] has been proposed as a decoder architecture to improve the continuity of VAE representations. CoordConv VAE appends coordinate channels to every feature layer of the standard deconvolutional decoder, yet does not spatially tile the latent vector, hence retains upsampling deconvolutions.

Figure 11 shows analysis of this model on the colored sprites dataset. While the latent space does appear to be continuous with respect to object position, it is quite entangled (far more so than a Spatial Broadcast VAE). This is not very surprising, since CoordConv VAE uses upscale deconvolutions to go all they way from spatial shape 1×1 to spatial shape 64×64, while in Table 10 we see that introducing upscaling hurts disentangling in a Spatial Broadcast VAE.

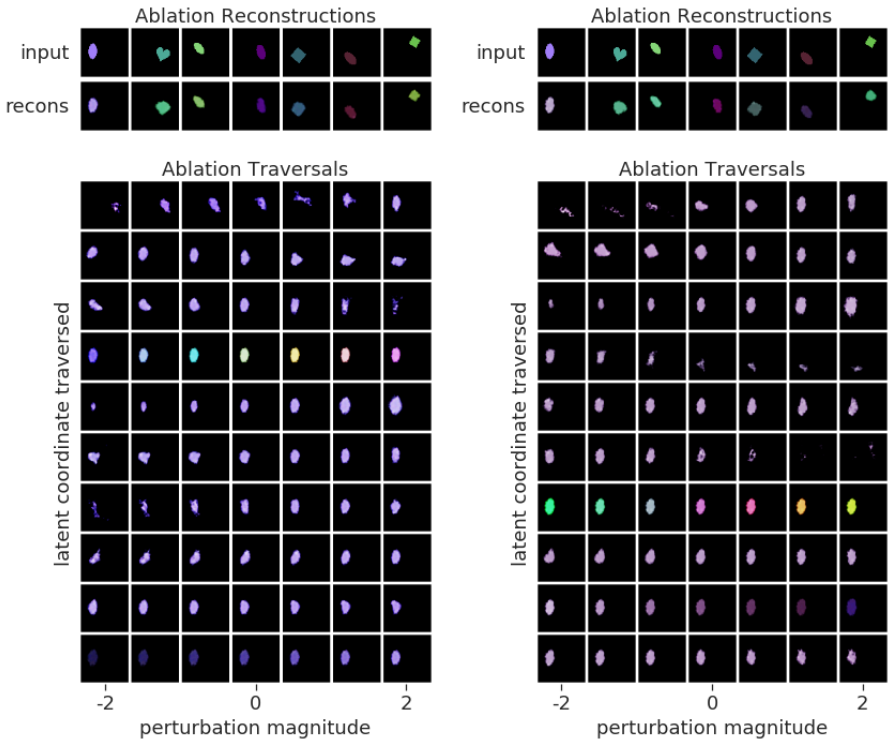


Figure 9: Ablation Study Here we see traversals from a Spatial Broadcast VAE with a random (though constant throughout training) shuffling of the coordinate channels. Different models (with different shufflings) are show on left and the right. We see that the this shuffling negatively impacts both reconstructions and traversals, hence the linear organization of the coordinate channels is important for the model's performance. Quantitatively, this shuffling causes the MIG to drop from 0.147 to 0.104 ± 0.025 (though still better than a DeConv VAE's 0.052) and the ELBO loss to drop from 329 to 362.8 ± 2.90 (worse than the DeConv VAE's 347). The shuffled-coordinate model has KL loss 26.56 ± 0.47, number of relevant latents 8.99 ± 0.37 and negative log likelihood 336.3 ± 2.98.

E Extra traversals for datasets without positional variation

As we acknowledge in the main text, a single latent traversal plot only shows local disentangling at one point in latent space. Hence to support our claim in Section 4.2 that the Spatial Broadcast VAE disentangled the Chairs and 3D objects datasets, we show in Figure 12 traversals about a second seed in each dataset for the same models as in Figure 5.

F Architecture Hyperparameters

In order to remain objective when selecting model hyperparameters for the Spatial Broadcast and Deconv decoders, we chose hyperparameters based on minimizing the ELBO loss, not considering any information about disentangling. After finding reasonable encoder hyperparameters, we performed large-scale (25 replicas each) sweeps over a few decoder hyperparameters for both the DeConv and Spatial Broadcast decoder on the colored sprites

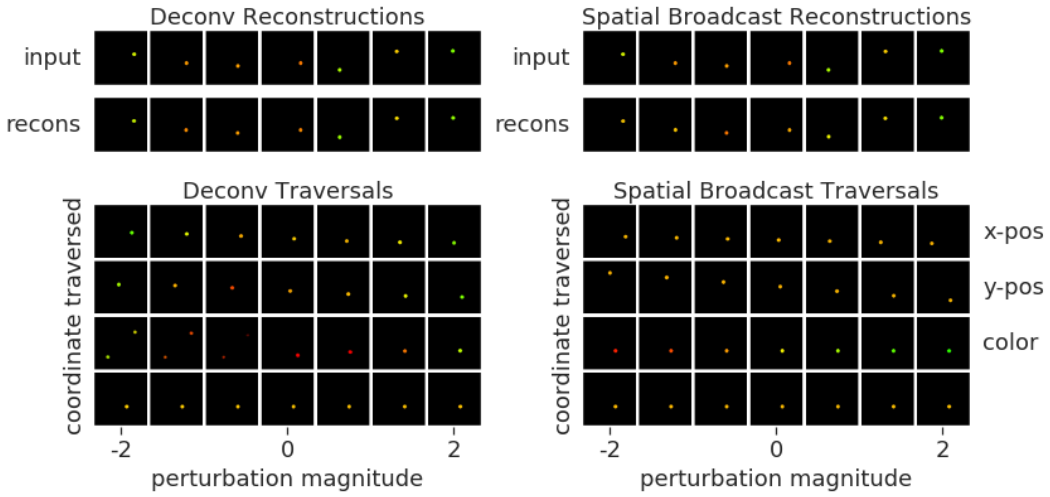


Figure 10: Disentangling tiny objects. A DeConv VAE *(left)* learns a highly entangled and discontinuous representation of this dataset of tiny hue-varying circles. In contrast, a Spatial Broadcast VAE *(right)* disentangles this dataset. The circles are 0.075 times the frame-width, while those in Figure 6 are 0.1 times the frame-width.

dataset. These sweeps are revealing of hyperparameter sensitivity, so we report the following quantities for them:

- ELBO. This is the evidence lower bound (total VAE loss). It is the sum of the negative log likelihood (NLL) and KL-divergence.
- NLL. This is the negative log likelihood of an image with respect to the model's reconstructed distribution of that image. It is a measure of reconstruction accuracy.
- KL. This is the KL divergence of the VAE's latent distribution with its Gaussian prior. It measures how much information is being encoded in the latent space.
- Latents Used. This is the mean number of latent coordinates with standard deviation less than 0.5. Typically, a VAE will have some unused latent coordinates (with standard deviation near 1) and some used latent coordinates. The threshold 0.5 is arbitrary, but this quantity does provide a rough idea of how many factors of variation the model may be representing.
- MIG. The MIG metric.
- Factor VAE. This is the metric described in the FactorVAE paper [Kim and Mnih, 2017]. We found this metric to be less consistent than the MIG metric (and equally flawed with respect to rotated coordinates), but it qualitatively agrees with the MIG metric most of the time.

F.1 ConvNet Depth

Table 5 shows results of sweeping over ConvNet depth in the Spatial Broadcast decoder. This reveals a consistent trend: As the ConvNet deepens, the model moves towards lower

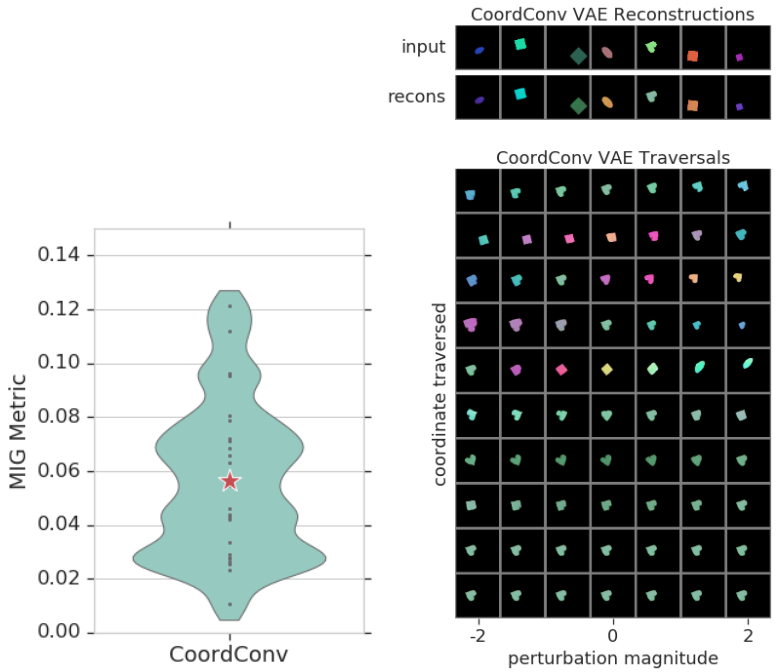


Figure 11: CoordConv VAE on colored sprites. *(left)* MIG results. Comparing to Figure 2, we see that this achieves far lower scores than a Spatial Broadcast VAE, though about the same as (or maybe slightly better than) a DeConv VAE. *(right)* Latent space traversals are entangled. Note that in contrast to traversal plots in the main text, we show the effect of traversing all 10 latent components (sorted by smallest to largest mean variance), including the non-coding ones (in the bottom rows).

rate/higher distortion. Consequently, latent space information and reconstruction accuracy drop. Traversals with deeper nets show the model dropping factors of variation (the dataset has 8 factors of variation).

Table 6 shows a noisier but similar trend when increasing DeConvNet depth in the DeConv decoder.

ConvNet	ELBO	NLL	KL	Latents Used	MIG	Factor VAE
2-Layer	339 (±2.3)	312 (±2.7)	27.5 (±0.54)	8.33 (±0.37)	0.076 (±0.038)	0.187 (±0.027)
3-Layer	**329 (±4.1)**	305 (±4.6)	24.4 (±0.54)	7.22 (±0.34)	0.147 (±0.057)	0.208 (±0.084)
4-Layer	341 (±6.8)	318 (±7.7)	22.6 (±0.97)	5.93 (±0.51)	0.157 (±0.045)	0.226 (±0.046)
5-Layer	340 (±8.8)	317 (±9.7)	22.7 (±0.99)	5.70 (±0.37)	0.173 (±0.059)	0.218 (±0.030)

Table 5: Effect of ConvNet depth on Spatial Broadcast VAE performance. These results use the colored sprites dataset. Traversals with deeper nets show the model dropping factors of variation (usually color first, then angle, shape, size, and position in that order). The increasing metric values with deeper ConvNets belies the fact that the model is encoding fewer of the 8 factors of variation in the dataset.

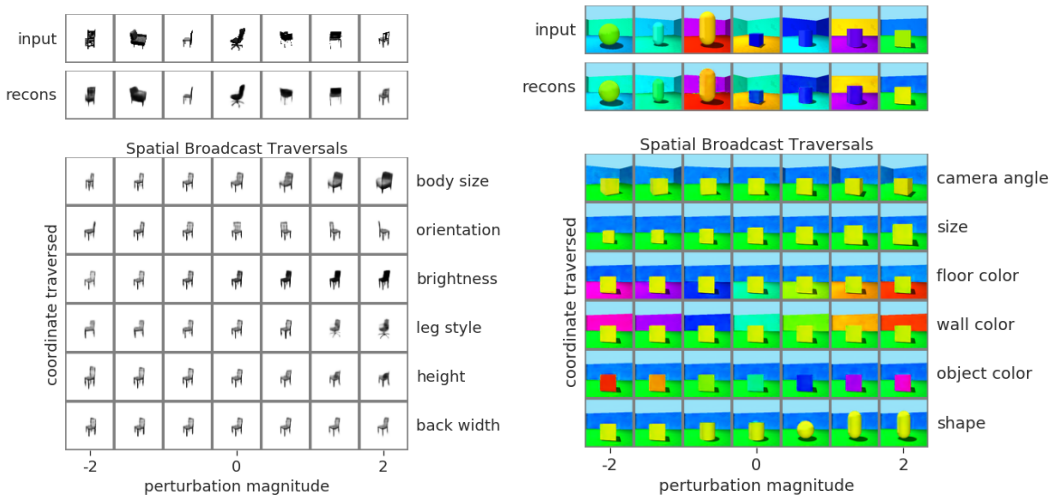


Figure 12: Traversals for datasets with no positional variation. Same as Figure 5 but with a different traversal origin in latent space. Cross-referencing with 5, we see that the Spatial Broadcast VAE does seem to globally disentangle these datasets.

ConvNet	ELBO	NLL	KL	Latents Used	MIG	Factor VAE
3-Layer	372 (±8.6)	346 (±8.9)	26.8 (±0.40)	9.20 (±0.04)	0.031 (±0.018)	0.144 (±0.031)
4-Layer	349 (±9.4)	322 (±10.0)	27.1 (±0.88)	8.90 (±0.24)	0.025 (±0.015)	0.139 (±0.009)
5-Layer	**340 (±9.8)**	314 (±10.4)	26.0 (±1.00)	7.95 (±0.64)	0.056 (±0.32)	0.184 (±0.053)
6-Layer	349 (±15.0)	326 (±16.1)	23.3 (±1.42)	6.36 (±0.81)	0.056 (±0.019)	0.199 (±0.029)

Table 6: Effect of ConvNet depth on DeConv VAE performance. These results use the colored sprites dataset. Similarly to Table 5, deeper convnets cause the model to represent fewer factors of variation.

F.2 MLP Depth

The Spatial Broadcast decoder as presented in this work is fully convolutional. It contains no MLP. However, motivated by the need for more depth on the 3D Object-in-Room dataset, we did explore applying an MLP to the input vector prior to the broadcast operation. We found that including this MLP had a qualitatively similar effect as increasing the number of convolutional layers on the colored sprited dataset, decreasing latent capacity and giving poorer reconstructions. These results are shown in Table 7.

However, on the 3D Object-in-Room dataset adding the MLP did improve the model when using ConvNet depth 3 (the same as for colored sprites). Results of a sweep over depth of a pre-broadcast MLP are shown in Table 9. As mentioned in Section 4.2, we were able to achieve the same effect by instead increasing the ConvNet depth to 6, but for those interested in computational efficiency using a pre-broadcast MLP may be a better choice for datasets of this sort.

In the DeConv decoder, increasing the MLP layers again has a broadly similar effect as increasing the ConvNet layers, as shown in Table 8.

MLP	ELBO	NLL	KL	Latents Used	MIG	Factor VAE
0-Layer	**329 (±4)**	305 (±5)	24.5 (±0.54)	7.21 (±0.34)	0.147 (±0.057)	0.208 (±0.084)
1-Layer	330 (±6)	307 (±6)	23.9 (±0.72)	6.68 (±0.46)	0.164 (±0.043)	0.200 (±0.034)
2-Layer	349 (±15)	327 (±17)	21.5 (±2.13)	6.03 (±0.66)	0.210 (±0.048)	0.232 (±0.045)
3-Layer	392 (±23)	399 (±114)	15.7 (±2.93)	4.17 (±1.23)	0.160 (±0.034)	0.275 (±0.064)

Table 7: Effect of a pre-broadcast MLP on the Spatial Broadcast VAE's performance. These results use the colored sprites dataset, on which an MLP seems to hurt performance (though as noted in the text this is not always the case on the 3D Object-in-Room dataset).

MLP	ELBO	NLL	KL	Latents Used	MIG	Factor VAE
1-Layer	**347 (±13)**	321 (±14)	25.8 (±1.3)	7.97 (±0.72)	0.052 (±0.020)	0.174 (±0.043)
2-Layer	352 (±17)	328 (±18)	23.7 (±1.7)	6.68 (±0.78)	0.051 (±0.024)	0.196 (±0.024)
3-Layer	365 (±19)	345 (±21)	19.7 (±2.4)	5.24 (±0.73)	0.144 (±0.062)	0.243 (±0.043)

Table 8: Effect of MLP depth on the DeConv VAE's performance. Increasing MLP depth seems to have a broadly similar effect as increasing DeConvNet depth, causing the model to represent fewer factors of variation in the data.

MLP	ELBO	NLL	KL	Latents Used	MIG	Factor VAE
0-Layer	4039 (±3.4)	4010 (±3.3)	29.3 (±0.46)	8.73 (±0.41)	0.541 (±0.091)	0.931 (±0.043)
1-Layer	4022 (±3.7)	4003 (±3.7)	19.3 (±0.35)	6.30 (±0.49)	0.538 (±0.105)	0.946 (±0.043)
2-Layer	4018 (±3.3)	3999 (±3.3)	18.5 (±0.30)	5.94 (±0.41)	0.574 (±0.096)	0.978 (±0.027)
3-Layer	4020 (±3.0)	4002 (±2.9)	18.3 (±0.38)	5.73 (±0.31)	0.659 (±0.123)	0.979 (±0.037)

Table 9: Effect of a pre-broadcast MLP on the Spatial Broadcast VAE's performance, 3D Object-in-Room dataset. This table is analogous to Table 7, except using the 3D Object-in-Room dataset. Here the model seems to over-representation the dataset generative factors (or which there are 6 for this dataset) without a pre-broadcast MLP (top row). However, adding a pre-broadcast MLP with 2 or 3 layers gives rise to accurate reconstructions with the appropriate number of used latents and good disentangling. Adding a pre-broadcast MLP like this is an alternative to increasing the ConvNet depth in the model (shown in Figure 5).

F.3 Decoder Upscale Factor

We acknowledge that there is a continuum of models between the Spatial Broadcast decoder and the Deconv decoder. One could interpolate from one to the other by incrementally replacing the convolutional layers in the Spatial Broadcast decoder's network by deconvolutional layers with stride 2 (and simultaneously decreasing the height and width of the tiling operation). Table 10 shows a few steps of such a progression, where (starting from the bottom) 1, 2, and all 3 of the convolutional layers in the Spatial Broadcast decoder are replaced by a deconvolutional layer with stride 2. We see that this hurts disentangling without affecting the other metrics, further evidence that upscaling deconvolutional layers are bad for representation learning.

MLP	ELBO	NLL	KL	Latents Used	MIG	Factor VAE
0 Upscales	329 (±4.1)	305 (±4.6)	24.4 (±0.54)	7.22 (±0.34)	0.147 (±0.057)	0.208 (±0.084)
1 Upscale	327 (±4.4)	302 (±4.9)	24.4 (±0.55)	7.29 (±0.26)	0.149 (±0.048)	0.194 (±0.026)
2 Upscales	329 (±4.3)	304 (±4.8)	24.2 (±0.60)	7.14 (±0.42)	0.122 (±0.045)	0.235 (±0.070)
3 Upscales	330 (±2.4)	305 (±2.7)	24.2 (±0.24)	7.39 (±0.08)	0.110 (±0.032)	0.182 (±0.028)

Table 10: Effect of upscale deconvolution on the Spatial Broadcast VAE's performance. These results use the colored sprites dataset. The columns show the effect of repeatedly replacing the convolutional, stride-1 layers in the decoder by deconvolutional, stride-2 layers (starting at the bottom-most layer). This incrementally reduces performance without affecting the other statistics much, testament to the negative impact of upscaling deconvolutional layer on VAE representations.

G Latent Space Geometry Analysis for Circle Datasets

We showed visualization of latent space geometry on the circles datasets in Figures 7 (with independent factors of variation) and 8 (with dependent factors of variation). These figures showcased the improvement that the Spatial Broadcast decoder lends. However, we also conducted the same style experiments on many more datasets and on FactorVAE models. In this section we will present these additional results.

We consider three generative factor pairs: (X-Position, Y-Position), (X-Position, Hue), and (Redness, Greenness). Broadly, the following figures show that the Spatial Broadcast decoder nearly always helps disentangling. It helps most dramatically on the most positional variation (X-Position, Y-Position) and least significantly when there is no positional variation (Redness, Greenness).

Note, however, that even with no position variation, the Spatial Broadcast decoder does seem to improve latent space geometry in the generalization experiments (Figures 20 and 21). We believe this may be due in part to the fact that the Spatial Broadcast decoder is shallower than the DeConv decoder.

Finally, we explore one completely different dataset with dependent factors: A dataset where half the images have no object (are entirely black). This we do to simulate conditions like that in a multi-entity VAE such as [Nash et al., 2017] when the dataset has a variable number of entities. These conditions pose a challenge for disentangling, because the VAE objective will wish to allocate a large (low-KL) region of latent space to representing a blank image when there is a large proportion of blank images in the dataset. However, we do see a stark improvement by using the Spatial Broadcast decoder in this case.

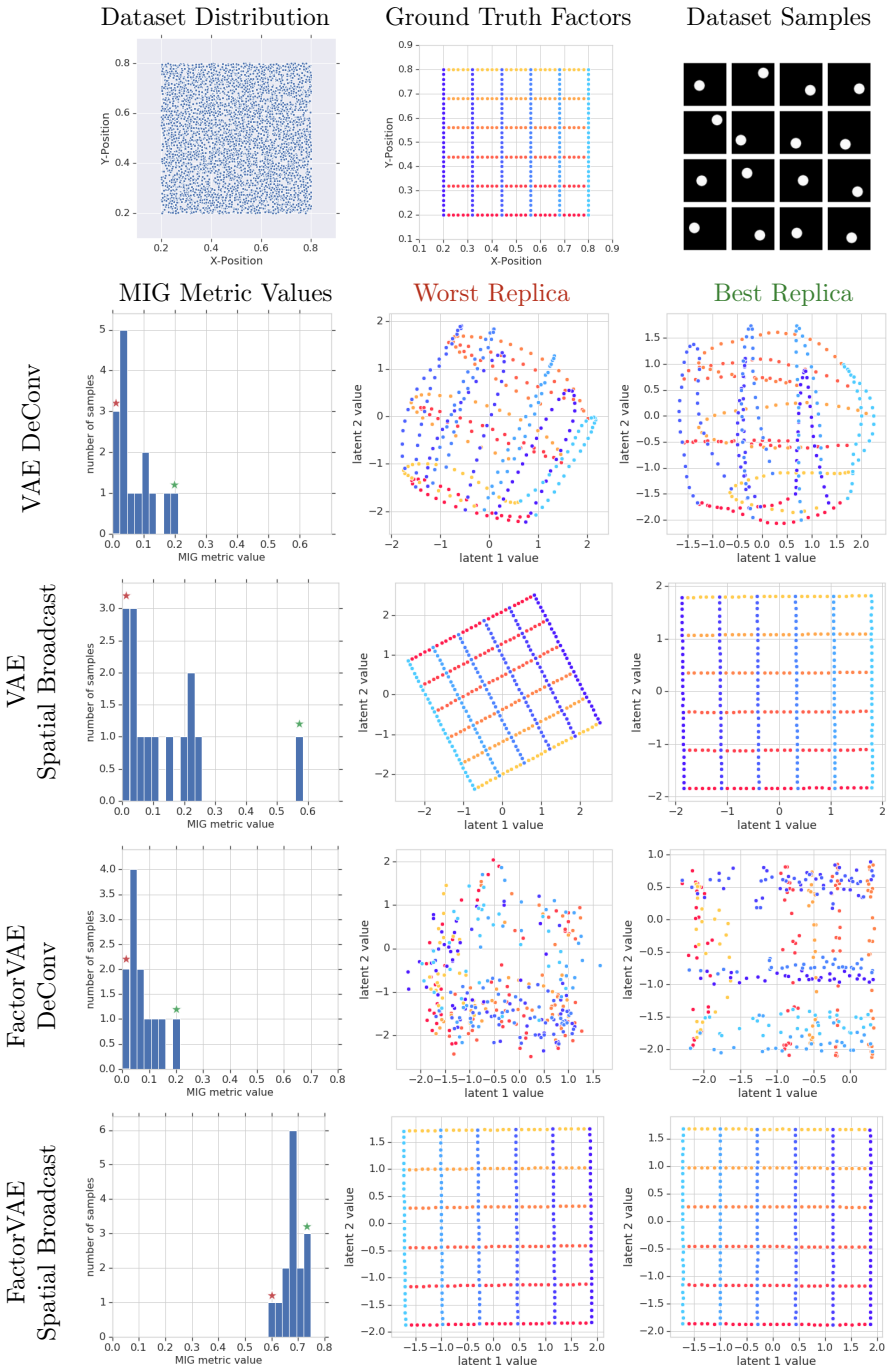


Figure 13: Latent space analysis for independent X-Y dataset. This is the same as in Figure 7, except with the additional FactorVAE results. We see that the Spatial Broadcast decoder improves FactorVAE as well as a VAE (and in the FactorVAE case seems to always be axis-aligned). Note that DeConv FactorVAE has a surprisingly messy latent space — we found that using a fixed-variance Normal (instead of Bernoulli) decoder distribution improved this significantly, though still not to the level of the Spatial Broadcast FactorVAE. We also noticed in small-scale experiments that including shape or size variability in the dataset helped FactorVAE disentangle as well. However, FactorVAE does seem to be generally quite sensitive to hyperparameters [Locatello et al., 2018], as adversarial models often are.

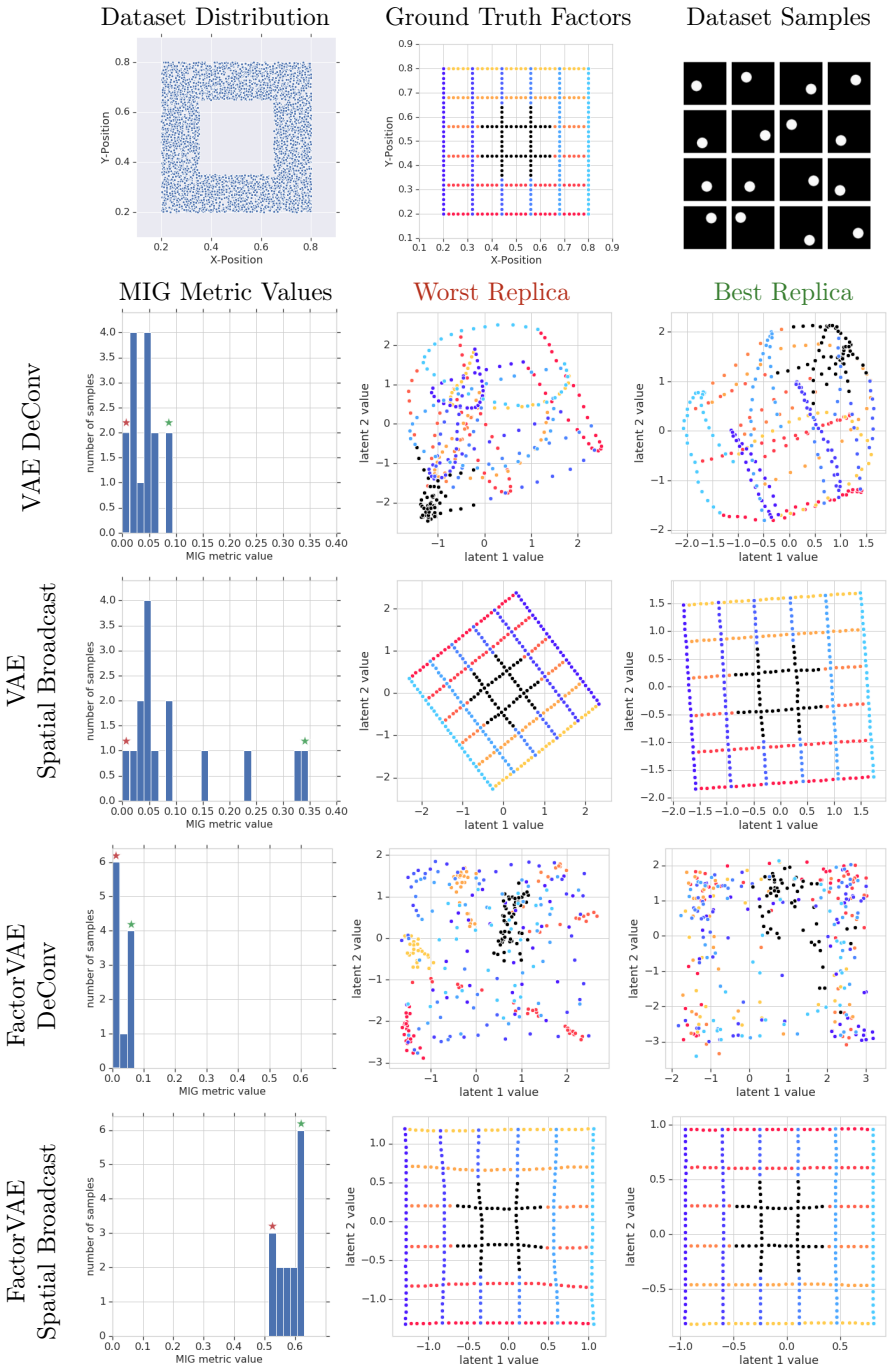


Figure 14: Latent space analysis for dependent X-Y dataset. This is the same as in Figure 8, though with the additional FactorVAE results. Again, the Spatial Broadcast decoder dramatically improves the representation — its representation looks nearly linear with respect to the ground truth factors, even through the extrapolation region.

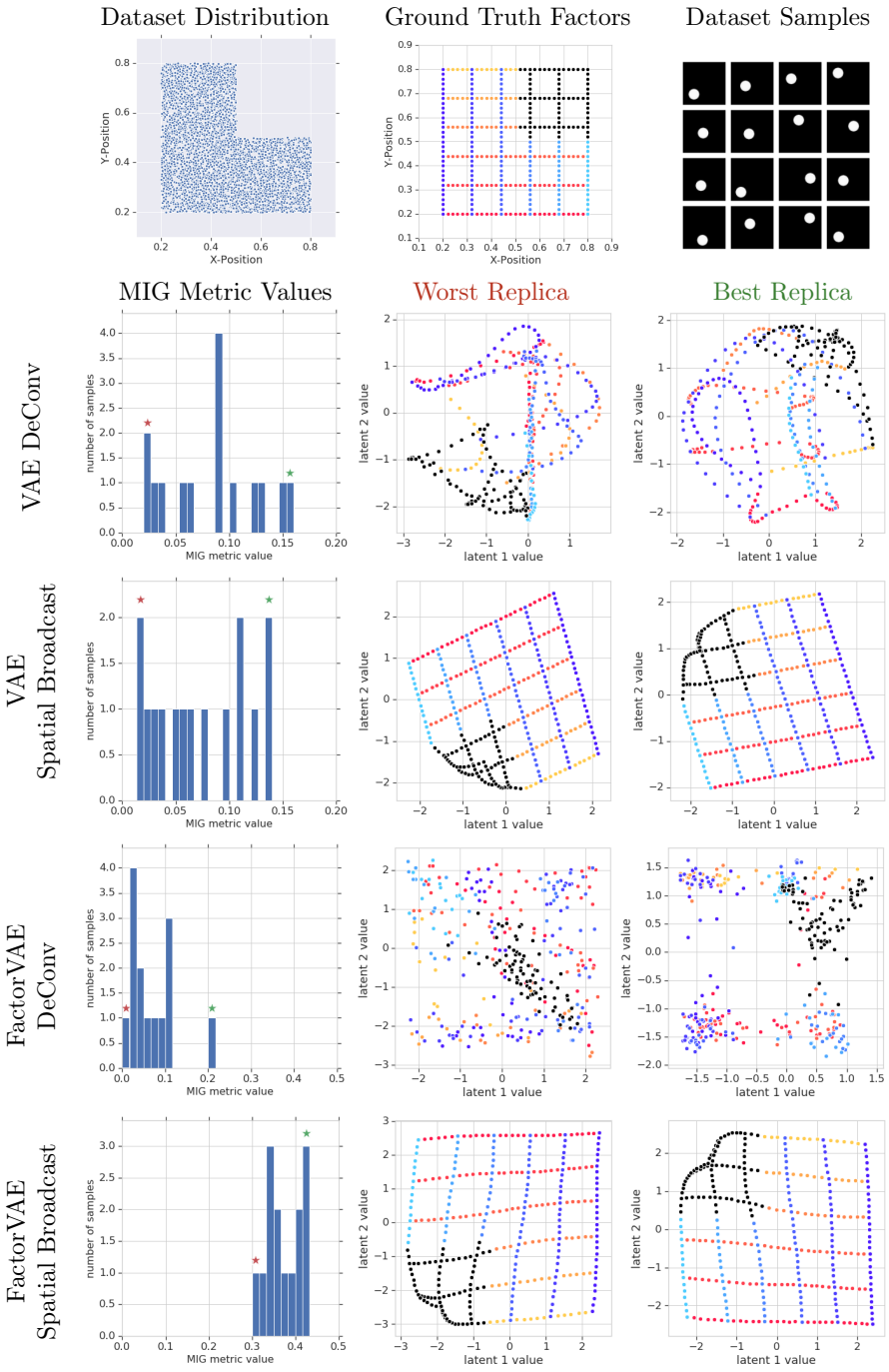


Figure 15: Latent space analysis for dependent X-Y dataset. This is similar to Figure 14, except the "hole" in the dataset is in the corner of generative factor space rather than the middle. Hence this tests not only extrapolation in pixel space, but also extrapolation in generative factor space. As usual, the Spatial Broadcast decoder helps a lot, though in this case the extrapolation is naturally more difficult than in Figure 14.

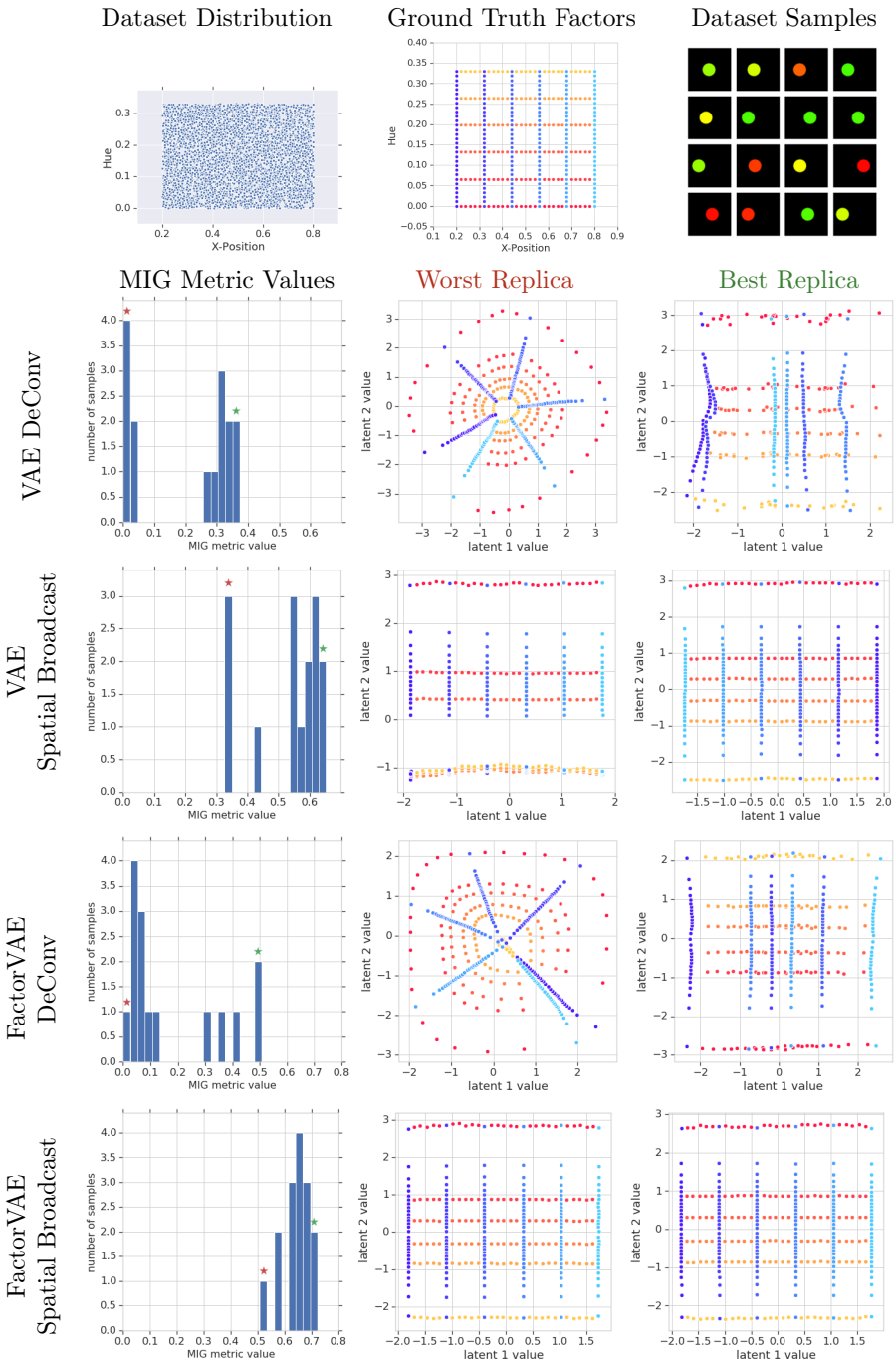


Figure 16: Latent space analysis for independent X-H dataset. In this dataset the circle varies in X-Position and Hue. As expected given this has less positional variation than the X-Y datasets, we see the relative improvement of the Spatial Broadcast decoder to be lower, though still quite significant. Interestingly, the representation with the Spatial Broadcast decoder is always axis-aligned and nearly linear in the positional direction, though non-linear in the hue direction. While this is not what the VAE objective is pressuring the model to do (the VAE objective would like to balance mean and variance in its latent space), we attribute this to the fact that a linear representation is much easier to compute from the coordinate channels with ReLU layers than a non-linear effect, particularly with only three convolutional ReLU layers. In a sense, the inductive bias of the architecture is overriding the inductive bias of the objective function.

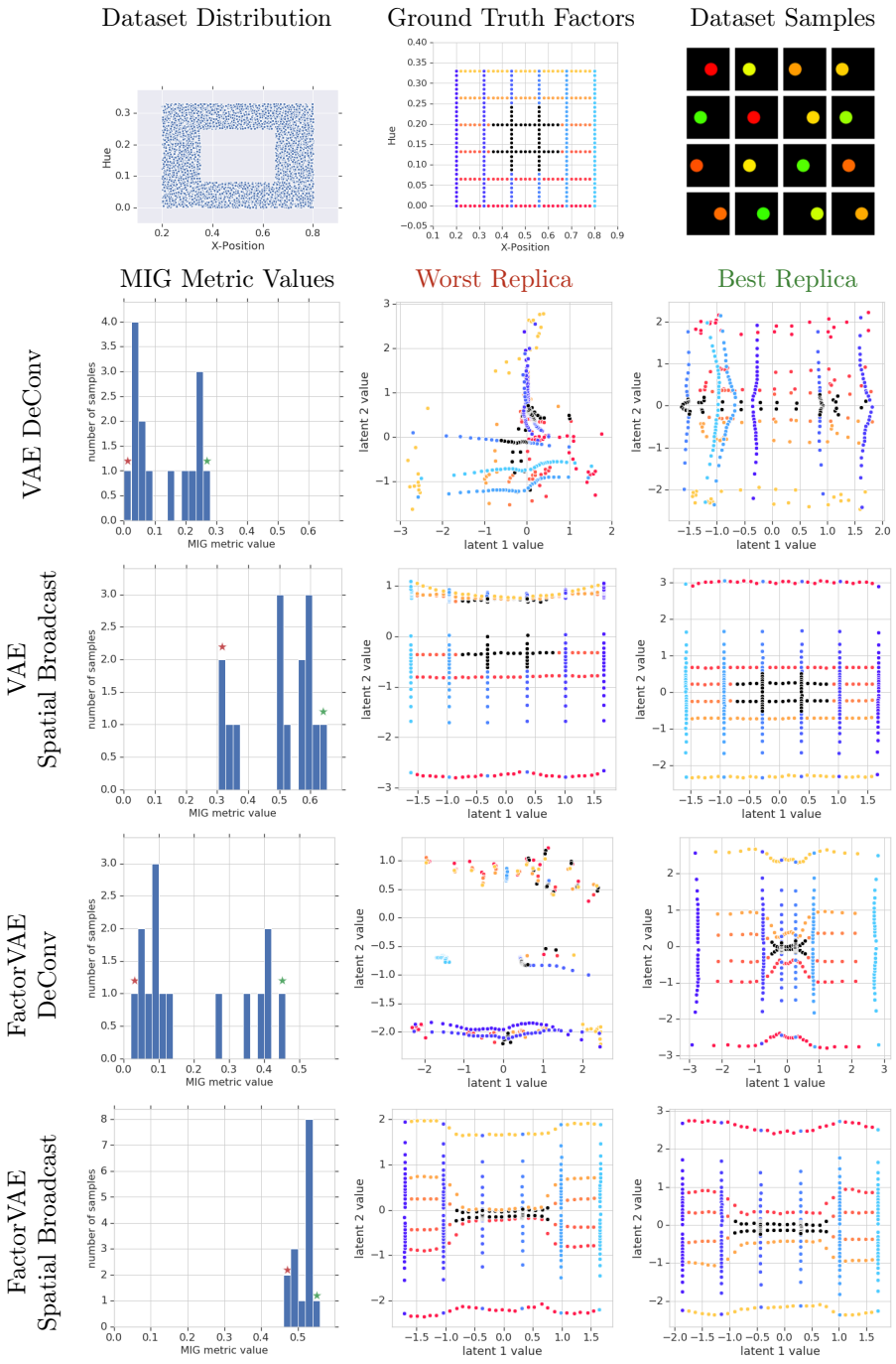


Figure 17: Latent space analysis for dependent X-H dataset. This is the same as Figure 16 except the dataset has a held-out "hole" in the middle, hence tests the model's generalization ability. This generalization is extrapolation in pixel space yet interpolation in generative factor space. This poses a serious challenge for the DeConv decoder, and again the Spatial Broadcast decoder helps a lot. Interestingly, note the severe contraction by FactorVAE of the "hole" in latent space. The independence pressure in FactorVAE strongly tries to eliminate unused regions of latent space.

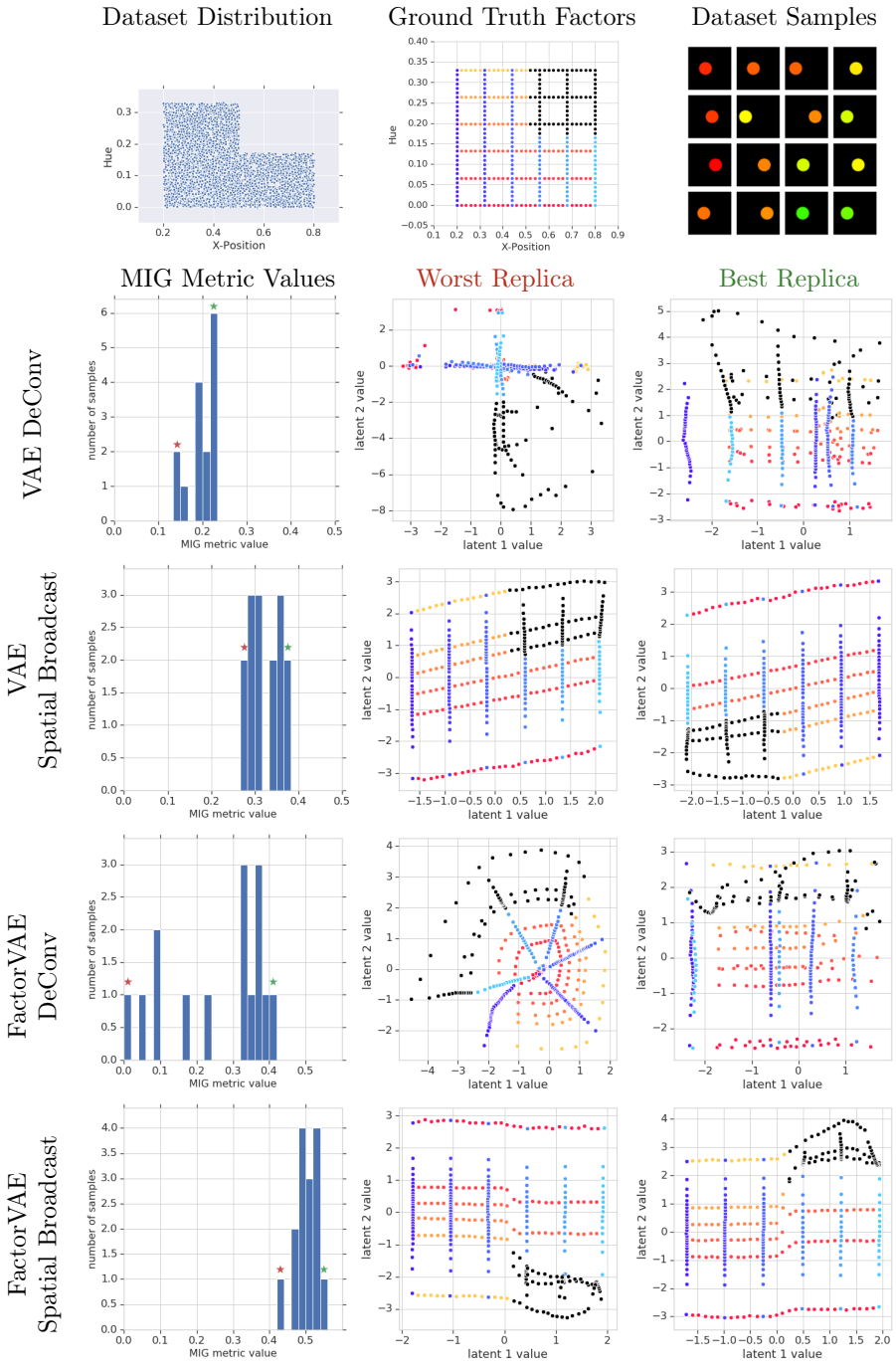


Figure 18: Latent space analysis for dependent X-H dataset. This is the same as Figure 17 except the held-out "hole" is in the corner of generative factor space, testing extrapolation in both pixel space and generative factor space. The Spatial Broadcast decoder again yields significant improvements, and as in Figure 17 we see FactorVAE clearly sacrificing latent space geometry to remove the "hole" from the latent space prior distribution (see bottom row).

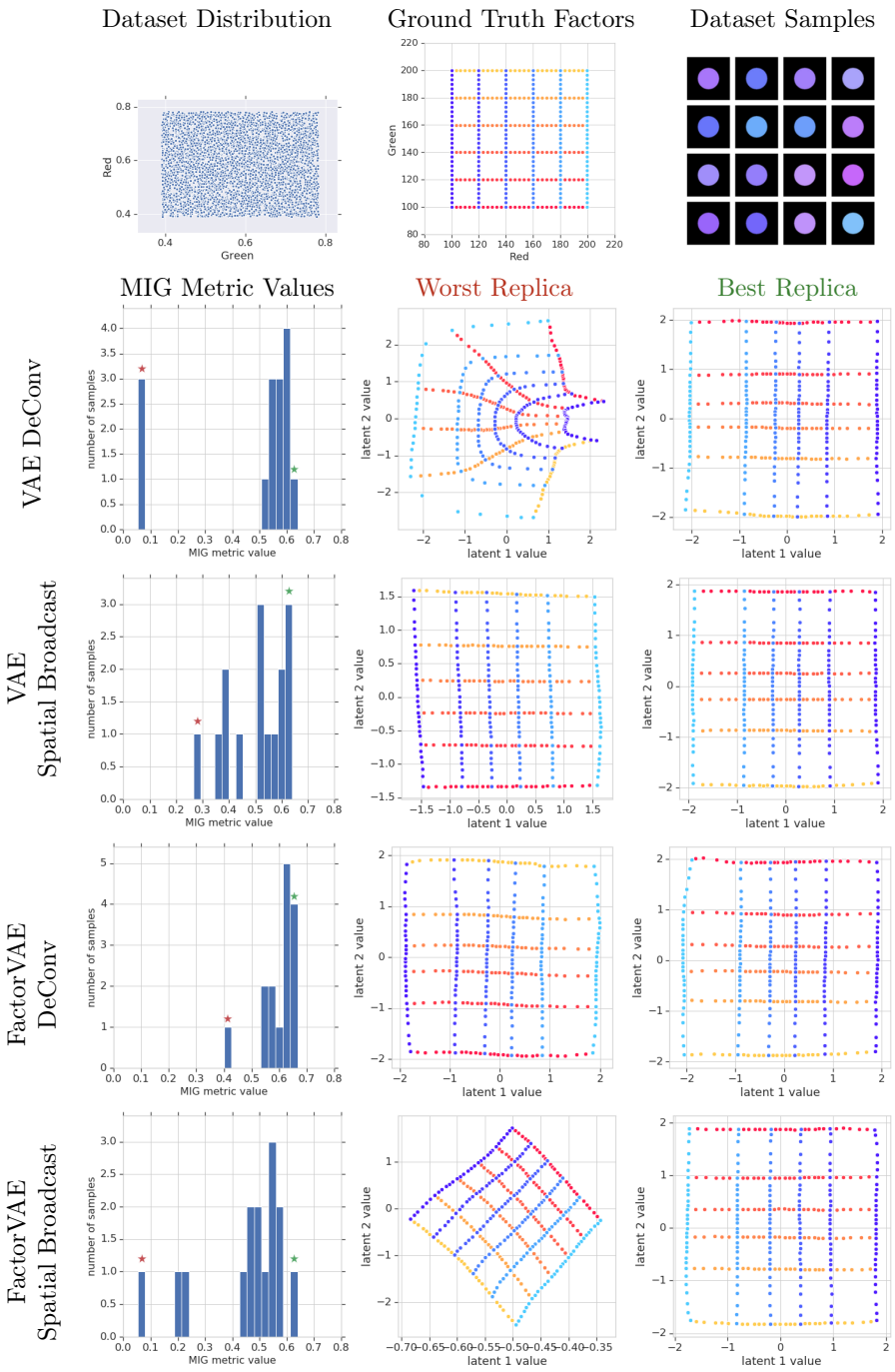


Figure 19: Latent space analysis for independent R-G dataset. In this dataset the circles vary only in their Redness and Greenness, not in their position. Here the benefit of the Broadcast decoder is less clear. It doesn't seem to hurt, and may help avoid a small fraction of poor seeds (3 out of 15 with MIG near zero) in a standard VAE.

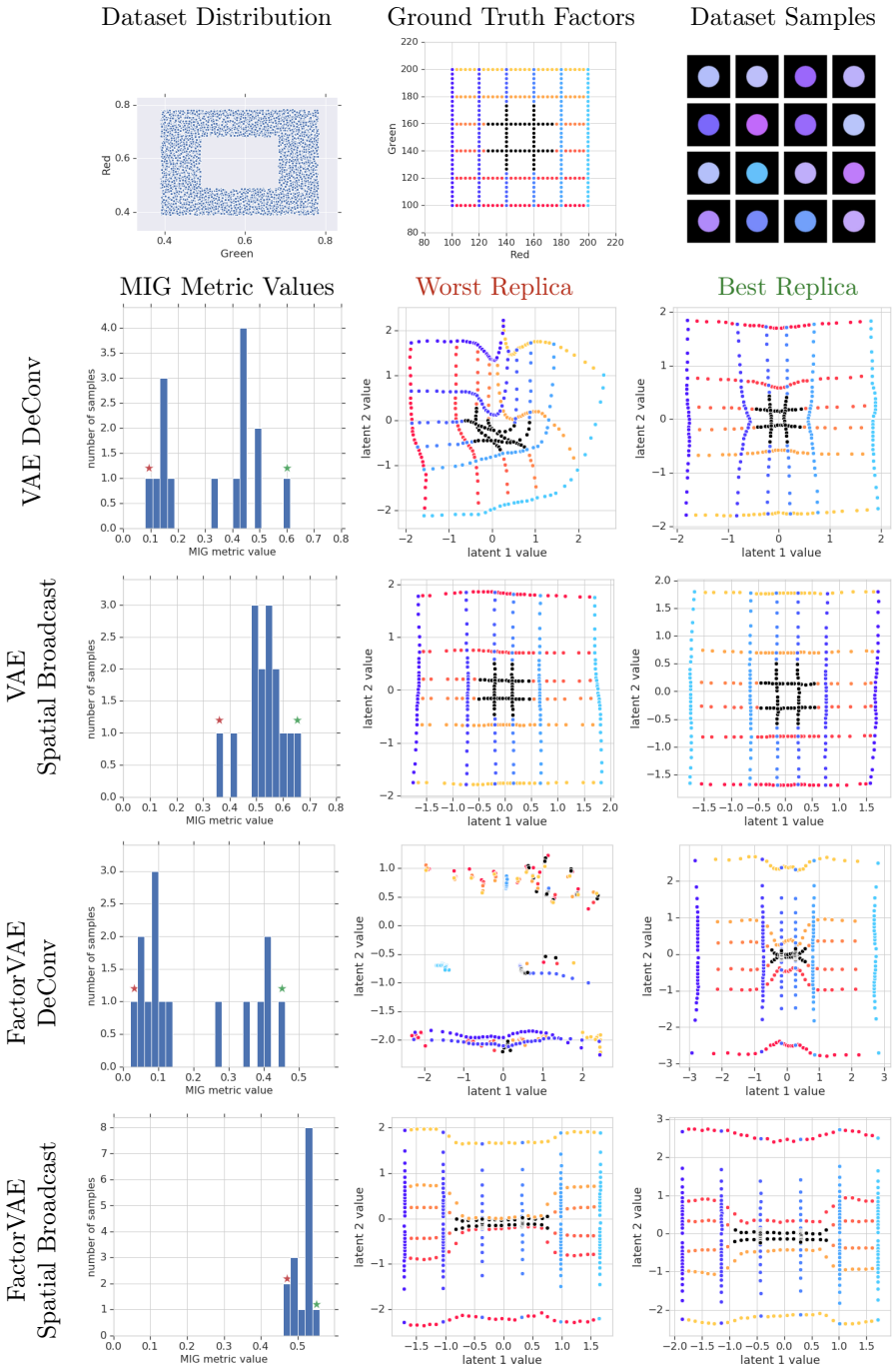


Figure 20: Latent space analysis for dependent R-G dataset. This is the same as Figure 19 except with a held-out "hole" in the center of generative factor space, testing extrapolation in pixel space (interpolation in generative factor space). Here the benefit of the Spatial Broadcast decoder is more clear than in Figure 19. We attribute it's benefit here in part to it being a shallower network.

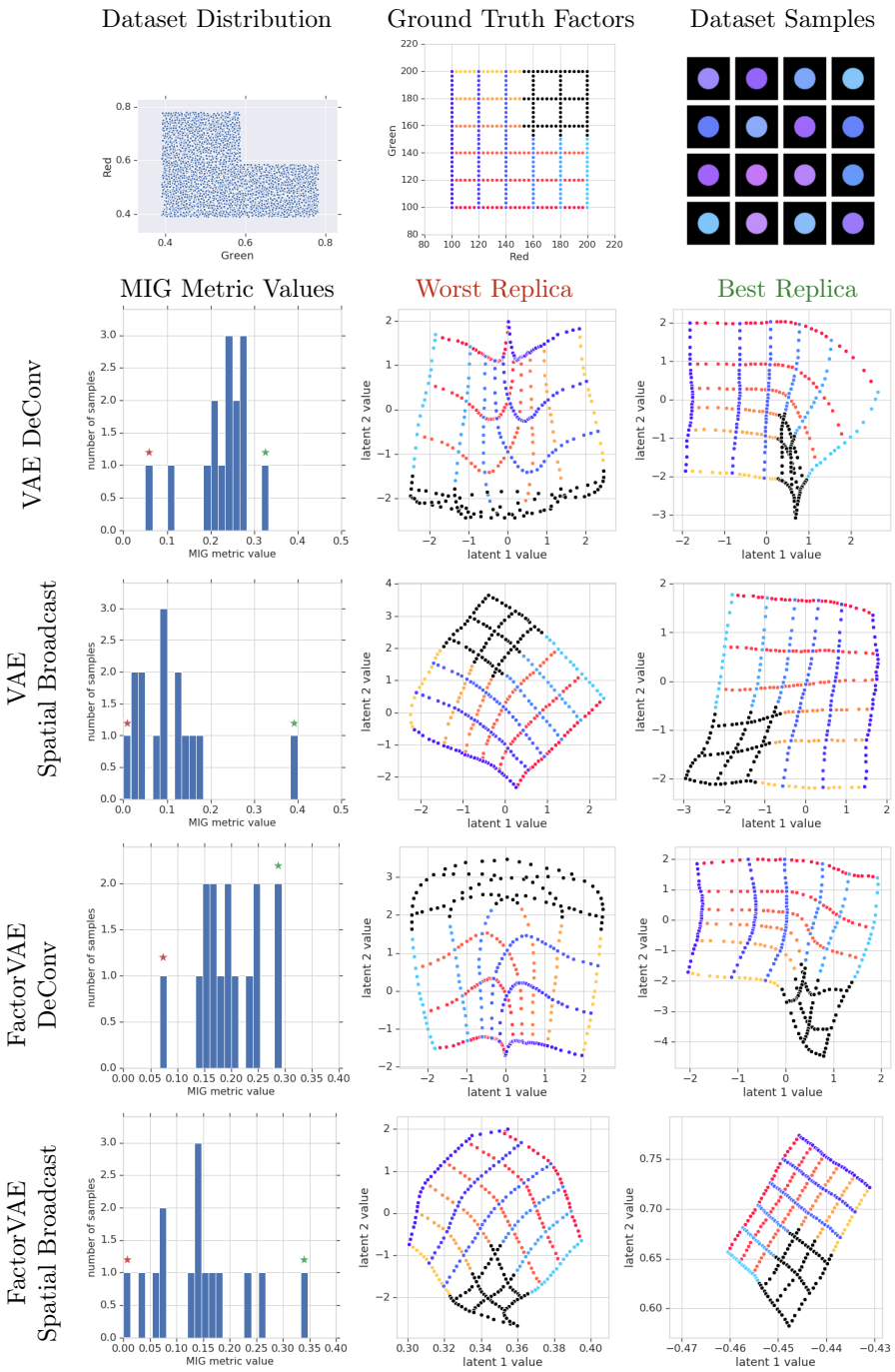


Figure 21: Latent space analysis for dependent R-G dataset. This is the same as Figure 20 except the "hole" is in the corner of generative factor space, testing extrapolation in both pixel space and generative factor space. While the Spatial Broadcast decoder seems to give rise to slightly lower MIG scores, this appears to be from rotation in the latent space. It looks like if anything the Spatial Broadcast decoder reduces warping of the latent space geometry, which is what we really care about in the representations.

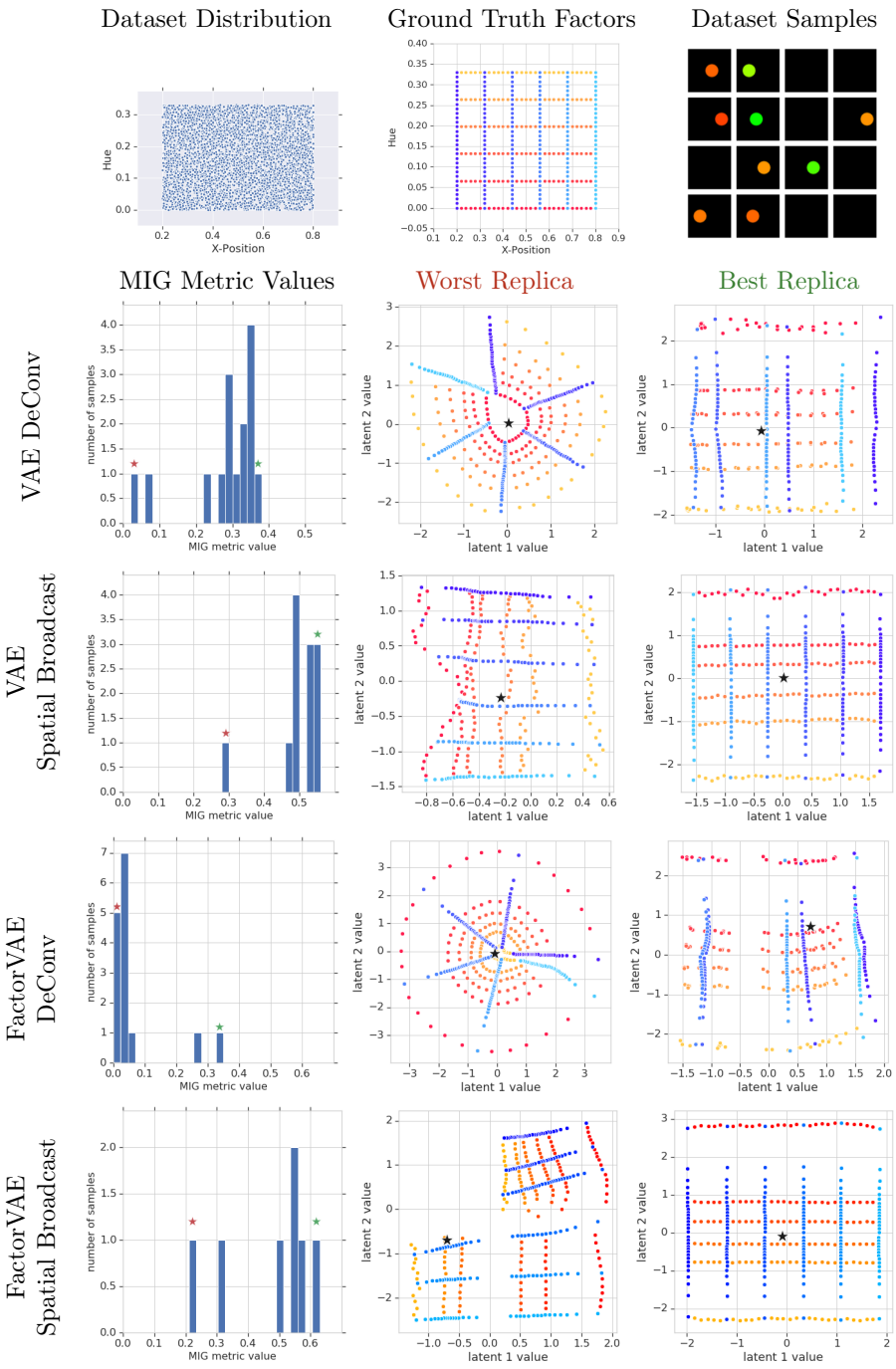


Figure 22: Latent space analysis for X-H dataset with blank images. This is the same as Figure 16 except the dataset consists half of images with no objects (entirely black images, as can be seen in the dataset samples in the top-right). This simulates the data distribution produced by the VAE of a multi-entity VAE [Nash et al., 2017] on a dataset with a variable number of objects. Again the Spatial Broadcast decoder improves latent space geometry, according to both the MIG metric and the traversals. In the latent geometry plots, the black star represents the encoding of a blank image. We noticed that the Spatial Broadcast VAE always allocates a third relevant latent to indicate the (binary) presence or absence of an object, a very natural representation of this dataset.