SYMBOLIC DISENTANGLED REPRESENTATIONS IN HYPERDIMENSIONAL LATENT SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

The idea of the disentangled representations is to reduce the data to a set of generative factors which generate it. Usually, such representations are vectors in the latent space, in which each coordinate corresponds to one of the generative factors. Then the object represented in this way can be modified by changing the value of a specific coordinate. But first, we need to determine which coordinate handles the desired generative factor, which can be complex with a high vector dimension. In this paper, we propose to represent each generative factor as a vector of the same dimension as the resulting representation. This is possible by using Hyperdimensional Computing principles (also known as Vector Symbolic Architectures), which represent symbols as high-dimensional vectors. They allow us to operate on symbols using vector operations, which leads to a simple and interpretable modification of the object in the latent space. We show it on the objects from dSprites and CLEVR datasets and provide an extensive analysis of learned symbolic disentangled representations in hyperdimensional latent space.

1 INTRODUCTION

Good data representation for machine learning algorithms is one of the key success factors for modern approaches. Initially, the construction of good representations consisted of feature engineering, i.e., the manual selection, creation, and generation of such features that allow the model to solve the main problem successfully. Although feature engineering is still used in some areas, current models rely on learning representations from data (Bengio et al. (2013)). At the same time, a good representation can be considered in several ways: proximity of representations for semantically related objects (Mikolov et al. (2013)), identification of common features in objects (Krizhevsky et al. (2012)), preservation of a complex structure with a decrease in dimension, and disentanglement of representations (Higgins et al. (2017)).

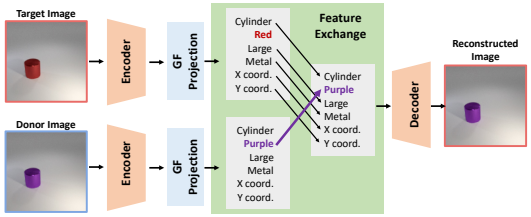

Figure 1: The target image and the donor image are represented by an encoder network and generative factors projection (GF Projection) as a set of high-dimensional vectors, each of which corresponds to one of the generator factors in the data. The same encoder is used for both the target and the donor objects. The donor image differs from the target image in terms of one of the generative factors and coincides with others. Further, the vector corresponding to the selected generation factor is exchanged by the donor image vector in the Feature Exchange module. The decoder reconstructs the target image with a given value of the generative factor.

In this paper, we consider disentangled representations. Following the work in Eastwood & Williams (2018), we define disentangled representations as satisfying three criteria: 1) *disentanglement*, i.e., the representation should factorize (disentangle) the underlying generative factors so that one variable capturing at most one factor; 2) *completeness*, a single variable should capture i.e., each underlying generative factor; 3) *informativeness*, i.e., completeness of information the representation captures about the underlying generative factors. The disentangled representation may potentially improve generalization and explainability in many machine learning tasks: structured scene representation and scene generation (El-Nouby et al. (2019)), reinforcement learning (Keramati et al. (2018)), reasoning (Yang et al. (2020)), and object-centric visual tasks (Groth et al. (2018)).

The main research on disentanglement is focused on obtaining representations expressed by a vector in which each vector's coordinate captures one generative factor (Chen et al. (2016); Higgins et al. (2017)), i.e., by changing this coordinate, we can change the property of the represented object. In this work, we propose a disentangled representation in which a vector of the same dimension captures each generative factor as the resulting representation of the object. We call such a representation a *symbolic disentangled representation*.

Our approach is based on the principles of Hyperdimensional Computing (HD or Vector Symbolic Architecture — VSA) (Kanerva (2009); Kleyko et al. (2021b)), in which symbols are represented by vectors of high and fixed dimensions. In VSA, the seed vectors, from which the description of an object is formed and which characterize its features, are usually obtained by sampling from a predetermined high-dimensional space. They are fixed and are not learned from data. VSA uses that with an extremely high probability all seed vectors from high-dimensional spaces are dissimilar to each other (quasi-orthogonal) Kleyko et al. (2021a), which allows us to reduce the manipulation of the symbols to vector operations.

In our work, the vectors representing the feature value of an object are obtained by applying the attention mechanism (Bahdanau et al. (2014)) over a set of fixed seed vectors (a codebook or an item memory in terms of VSA) and the representation of the object obtained using the autoencoder. This procedure allows the model to learn disentangled representations and edit objects in a controlled manner by manipulating their latent representations for cases where only one object is represented in the scene. We have demonstrated how the proposed model learns symbolic disentangled representations on modified dSprites (Matthey et al. (2017)) and CLEVR datasets (Johnson et al. (2017)).

The contributions of the paper are: 1) a new model is proposed that allows learning symbolic disentangled representations based on a combination of attention over codebook vectors and the principles of Hyperdimensional Computing; 2) learned representations allow us to edit the properties of objects in a controlled and interpretable way by manipulating their representations in the latent space.

2 Object Representation in the Latent Space

In this paper, we apply principles of VSAs to represent an object in latent space. VSA is a computing framework that works with high-dimensional vectors (HD vectors). In most applications (Kleyko et al. (2019); beim Graben et al. (2020); Kovalev et al. (2022)), to represent an object in high-dimensional latent space (the dimension D of the space is typically greater than 1,000), random HD vectors are sampled from that space. But some approaches learn vectors from data (Yilmaz (2015); Osipov et al. (2021)). These vectors are called seed vectors.

HD vectors perform a distributed representation of information across all components. Thus, only the entire vector could be interpreted, not individual components. It is different from the localist representation, which modern disengagement models use, where a single vector component potentially has a meaning. The nature of the vector space might be different that results in binary (Kanerva (2009)), real (Gayler (1998)), or complex (Plate (2003); Komer et al. (2019)) HD vectors. If an object is of a complex structure, then the resulting representation is obtained by performing vector operations on the seed vectors defined for VSAs: addition $+$ (bundling) and multiplication $\odot$ (binding). We describe these operations in more detail in the Appendix B. Using these operations, we can represent any object generated by n underlying generative factors as a set of attribute-value pairs (generative factor-value):

$$O = G_1 \odot V_1 + G_2 \odot V_2 + \cdots + G_n \odot V_n, \tag{1}$$

where G_i is a i-th generative factor and V_i is its value.

Thus, the target (T) object in Fig. 1 could be represented as:

$$\begin{aligned} T = &Shape \odot Cylinder + Color \odot Red \\ &+ Size \odot Large + Material \odot Metal \\ &+ Coord_X \odot X + Coord_Y \odot Y, \end{aligned} \tag{2}$$

where $Shape, Color, Size, Material, Coord_x, Coord_y$ are underlying generative factors and $Cylinder, Red, Large, Metal, X, Y$ are corresponding values.

In this work, to encode generative factors G_i, we use HD vectors that are generated during model initialization by sampling from $\mathcal{N}(0,1)^D$, where the normal distribution $\mathcal{N}(0,1)$ with mean equal to zero and variance equal to one, and D is the dimension of the space. These vectors do not change during training and testing. We sample the number of vectors equal to the number of factors.

Vectors V_i representing the values of generative factors are obtained as follows (Fig. 2). For each generative factor, seed vectors $S_\ell^{G_i}$ are sampled in the number of possible values of this factor G_i, using the procedure described above. These vectors are stored in the codebook (item memory). Thus, the model uses as many codebooks as there are generative factors in the data. Further, the object is mapped in the latent space using the autoencoder. Then, using the Generative Factors Projection (GF Projection), an intermediate value vector V_i' is obtained. This vector is then fed into the Generative Factor Representation (GF Representation) module, which uses the attention mechanism (Bahdanau et al. (2014)) to represent the value vector V_i as a linear combination of seed vectors from the codebook $S_\ell^{G_i}$:

$$a_\ell = softmax(\frac{V_i' K_\ell}{\sqrt{D}})_\ell, \tag{3}$$

where $softmax()_\ell$ — ℓ-th component, D — the dimension of a high-dimensional space, K_ℓ — projection of the ℓ-th seed vector $S_\ell^{G_i}$ from the codebook,

$$V_i = a_1 S_1^{G_i} + a_2 S_2^{G_i} + \cdots + a_\ell S_\ell^{G_i}. \tag{4}$$

The obtained value vectors V_i are bound with generative factor vectors G_i and are summed together in a High-Dimensional Representation (HD Representation) module. The resulting vector O is used when decoding the object into an image.

3 Experiments

In this work, we use paired datasets obtained from the data. The main idea of training and the process of dataset generation are presented in Appendix C and D correspondingly. For all experiments, the training signal is provided from the sum of image reconstruction errors (mean squared) and the loss function for the latent vector (KL divergence multiplied by coefficient 1e-3). The detailed description of the architectures of the used modules is presented in Appendix E.

4 Results

We provide quantitative results that evaluate the quality of the reconstruction of objects and scenes. We use the FID metric (Heusel et al. (2017)) for the CLEVR dataset and Intersection over Union (IoU) for the dSprites dataset.

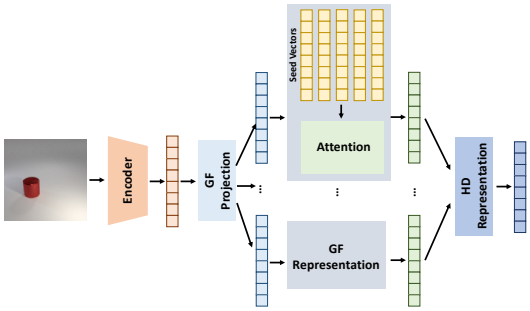

Figure 2: Obtaining an HD representation of an object.

4.1 dSprites paired

The model has reached the IoU value equal to 0.983. After training, we tested the possibility of exchanging features between random images (Fig. 3). The top left corner shows the two original images (ellipse and heart). Just below, the second line shows the result of their reconstruction. On the right, each image shows the result of latent vector decoding after replacing one corresponding feature. The "shape" feature is replaced correctly but with deformation. When "orientation" and "scale" are replaced, the main goal is accomplished, but the shape of the object is slightly distorted in the case of "orientation".

On lines 3 and 4, the ellipse features are replaced one by one by the square features. Here we can see that the "orientation" feature is strongly related to the "shape" feature, unlike the "scale", "pos x", and "pos y" features, with which the images are restored relatively well. We explain this by the

symmetry features of the figures: the square, the oval, and the heart are symmetric when rotated by pi/2, pi, and 2pi, respectively, while the possible angles of rotation are in $[0, 2\pi]$.

In addition, the lower left corner of the image shows that the image with the generative combination excluded from the training sample (square with coordinate $x > 0.5$) is reconstructed correctly. In both examples, the X and Y coordinates are correctly transferred from the donor image.

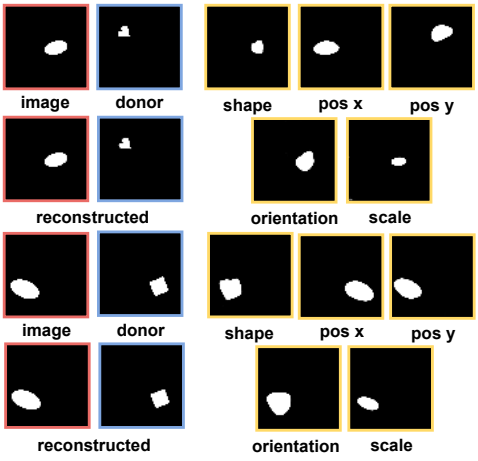

Figure 3: Image reconstruction of objects with modified values of generative factors (yellow) for **dSprites paired** dataset. The target object (red) differs from the donor object (blue) in all factor values.

4.2 CLEVR1 PAIRED

Fig. 4 shows the process of exchanging features between objects from the **CLEVR1 paired** dataset. The feature exchange between random images works well with the "color", "size", and "material" properties. In contrast to the **dSprites paired** dataset, there are problems with reconstruction when changing coordinates. The "shape" feature exchange works well with respect to reconstructed images. The mean FID metric for **CLEVR1 paired** dataset for test set with batch size of 64 is 69.19.

More qualitative results and visualization of the learned symbolic disentangled representations are presented in Appendix G. The ablation studies are presented in Appendix H.

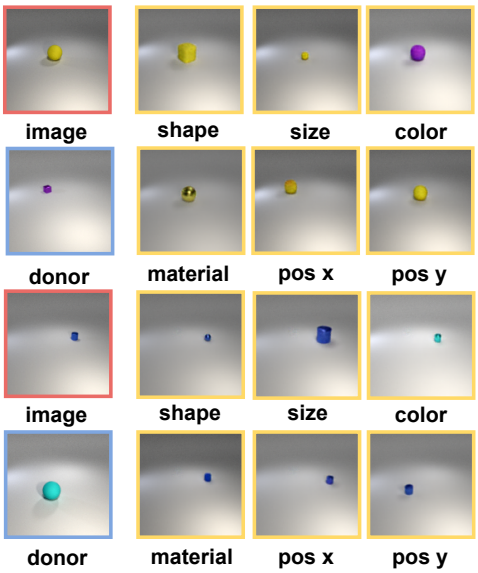

Figure 4: Image reconstruction of objects with modified values of generative factors (yellow) for **CLEVR1 paired** dataset. The target object (red) differs from the donor object (blue) in all factor values.

5 CONCLUSION AND DISCUSSION

In this paper, we proposed a model that learns symbolic disentangled representations using a structured representation of an object in latent space. This representation is based on the principles of VSAs and is a superposition of HD vectors that capture a single generative factor in the data.

Unlike classical approaches based on VAE (Kingma & Welling (2013)) or GAN (Goodfellow et al. (2014)), learned representations are distributed, i.e. individual coordinates are not interpretable (unlike localist representations). Potentially, this allows us to reduce the change in the properties of an object to manipulation with its latent representation using vector operations defined in VSAs.

The limitation of the application of the proposed model for real data is due to the fact that it is necessary to know in advance the number of generative factors. The generalization of the proposed model to the case of real data is an interesting direction for further research.

Also, due to the use of distributed representations, there is difficulty in comparison with existing models for disentangled views that use localist representations. However, establishing a common metric for disentangled representations is still an unsolved problem, and exploring ways to compare models with different representations (localist, distributed, hierarchical, and others) is a challenging task.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2014.

Peter beim Graben, Markus Huber, Werner Meyer, Ronald Römer, Constanze Tschöpe, and Matthias Wolff. Vector symbolic architectures for context-free grammars. *CoRR*, abs/2003.05171, 2020. URL https://arxiv.org/abs/2003.05171.

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. doi: 10.1109/TPAMI.2013.50.

Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in beta-vae, 2018. URL https://arxiv.org/abs/1804.03599.

Ricky T. Q. Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/1ee3dfcd8a0645a25a35977997223d22-Paper.pdf.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *NIPS*. Curran Associates, Inc., 2016.

Yixuan Chen, Yubin Shi, Dongsheng Li, Yujiang Wang, Mingzhi Dong, Yingying Zhao, Robert Dick, Qin Lv, Fan Yang, and Li Shang. Recursive disentanglement network. In *ICLR*, 2022.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=By-7dz-AZ.

Alaaeldin El-Nouby, Shikhar Sharma, Hannes Schulz, Devon Hjelm, Layla El Asri, Samira Ebrahimi Kahou, Yoshua Bengio, and Graham W.Taylor. Tell, draw, and repeat: Generating and modifying images based on continual linguistic instruction. *ICCV*, 2019.

Ross Gayler. Multiplicative binding, representation operators, and analogy. In *Advances in Analogy Research: Integration of Theory and Data from the Cognitive, Computational, and Neural Sciences*, pp. 1–4, 01 1998.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL https://arxiv.org/abs/1406.2661.

Oliver Groth, Fabian B. Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. 2018.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *NIPS*. Curran Associates, Inc., 2017.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2):139–159, Jun 2009. ISSN 1866-9964. doi: 10.1007/s12559-009-9009-8. URL https://doi.org/10.1007/s12559-009-9009-8.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018. URL http://arxiv.org/abs/1812.04948.

Ramtin Keramati, Jay Whang, Patrick Cho, and Emma Brunskill. Fast exploration with simplified models and approximately optimistic planning in model based reinforcement learning. 2018.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *ICML*, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013. URL https://arxiv.org/abs/1312.6114.

Denis Kleyko, Evgeny Osipov, Daswin De Silva, Urban Wiklund, Valeriy Vyatkin, and Damminda Alahakoon. Distributed representation of n-gram statistics for boosting self-organizing maps with hyperdimensional computing. In Nikolaj Bjørner, Irina Virbitskaite, and Andrei Voronkov (eds.), *Perspectives of System Informatics*, pp. 64–79, Cham, 2019. Springer International Publishing. ISBN 978-3-030-37487-7.

Denis Kleyko, Mike Davies, E. Paxon Frady, Pentti Kanerva, Spencer J. Kent, Bruno A. Olshausen, Evgeny Osipov, Jan M. Rabaey, Dmitri A. Rachkovskij, Abbas Rahimi, and Friedrich T. Sommer. Vector Symbolic Architectures as a Computing Framework for Nanoscale Hardware. pp. 1–28, 2021a. URL http://arxiv.org/abs/2106.05268.

Denis Kleyko, Dmitri A Rachkovskij, Evgeny Osipov, and Abbas Rahim. A survey on hyperdimensional computing aka vector symbolic architectures, part ii: Applications, cognitive models, and challenges. *arXiv preprint arXiv:2112.15424*, 2021b.

Brent Komer, Terrence C Stewart, Aaron R Voelker, and Chris Eliasmith. A neural representation of continuous space using fractional binding. In *41st annual meeting of the cognitive science society*. QC: Cognitive Science Society, 2019.

Alexey K. Kovalev, Makhmud Shaban, Evgeny Osipov, and Aleksandr I. Panov. Vector semiotic model for visual question answering. *Cognitive Systems Research*, 71:52–63, 2022. ISSN 1389-0417. doi: https://doi.org/10.1016/j.cogsys.2021.09.001. URL https://www.sciencedirect.com/science/article/pii/S1389041721000632.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. In *6th International Conference on Learning Representations, ICLR*. OpenReview.net, 2018.

Zinan Lin, Kiran Koshy Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr: Disentangling generative adversarial networks with contrastive regularizers. *CoRR*, abs/1906.06034, 2019. URL http://arxiv.org/abs/1906.06034.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017. URL https://arxiv.org/abs/1711.05101.

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

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun (eds.), *ICLR, Workshop Track*, 2013.

Milton L. Montero, Jeffrey S. Bowers, Rui Ponte Costa, Casimir J. H. Ludwig, and Gaurav Malhotra. Lost in latent space: Disentangled models and the challenge of combinatorial generalisation, 2022. URL https://arxiv.org/abs/2204.02283.

Weili Nie, Tero Karras, Animesh Garg, Shoubhik Debnath, Anjul Patney, Ankit B. Patel, and Anima Anandkumar. Semi-supervised stylegan for disentanglement learning. *CoRR*, abs/2003.03461, 2020. URL https://arxiv.org/abs/2003.03461.

Evgeny Osipov, Sachin Kahawala, Dilantha Haputhanthri, Thimal Kempitiya, Daswin De Silva, Damminda Alahakoon, and Denis Kleyko. Hyperseed: Unsupervised learning with vector symbolic architectures. *CoRR*, abs/2110.08343, 2021. URL https://arxiv.org/abs/2110.08343.

T. A. Plate. *Holographic Reduced Representations: Distributed Representation for Cognitive Structures*. Stanford: Center for the Study of Language and Information (CSLI), USA, 2003.

Xuanchi Ren, Tao Yang, Yuwang Wang, and Wenjun Zeng. Learning disentangled representation by exploiting pretrained generative models: A contrastive learning view. In *ICLR*, 2022.

Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates, 2017. URL https://arxiv.org/abs/1708.07120.

Jianwei Yang, Jiayuan Mao, Jiajun Wu, Devi Parikh, David Cox, Joshua B. Tenenbaum, and Chuang Gan. Object-centric diagnosis of visual reasoning. 2020.

Tao Yang, Xuanchi Ren, Yuwang Wang, Wenjun Zeng, and Nanning Zheng. Towards building a group-based unsupervised representation disentanglement framework. In *ICLR*, 2022.

Ozgur Yilmaz. Analogy making and logical inference on images using cellular automata based hyperdimensional computing, 2015.

A APPENDIX — DISENTANGLED REPRESENTATION

Most of the methods for learning disentangled representations are based on either the VAE framework Higgins et al. (2017); Burgess et al. (2018); Kim & Mnih (2018); Chen et al. (2018); Kumar et al. (2018) or Adversarial Generative Networks (GAN) Goodfellow et al. (2014) framework Chen et al. (2016); Lin et al. (2019); Nie et al. (2020).

In these approaches, they achieved the disentanglement by imposing additional restrictions on the loss: introducing a β parameter to balance independence constraints with reconstruction accuracy in β-VAE Higgins et al. (2017); by adding objective for capacity control in Burgess et al. (2018); by factorizing the representations distribution in FactorVAE Kim & Mnih (2018); by decomposing the evidence lower bound in β-TCVAE Chen et al. (2018); by minimising the covariance between the latents in DIP-VAE Kumar et al. (2018); by maximizing the mutual information in InfoGAN Chen et al. (2016); by introducing contrastive regularizer in InfoGAN-CR Lin et al. (2019); by adding a mutual information loss to StyleGAN Karras et al. (2018) in Info-StyleGAN Nie et al. (2020). In our approach, we do not impose additional specific restrictions on the loss, but use a structured representation of the object in the latent space and a special learning procedure.

Some approaches impose additional restrictions based on group theory on existing VAE models Yang et al. (2022) or propagate inductive regulatory bias recursively across the compositional feature space Chen et al. (2022), or provide a framework for learning disentangled representation and discovering the latent space Ren et al. (2022).

In our work, we achieve disentanglement of representations using Hyperdimensional Computing principles and representing generative factors as high-dimensional vectors.

B APPENDIX — VSAS VECTOR OPERATIONS

Here we explain vector operations defined for VSAs using an example of the Multiply-Add-Permute Gayler (1998) implementation of VSAs that works with real vectors and that we use in this paper. The exact realization of vector operations varies for different vector spaces while keeping computational properties.

Two main operations are addition and multiplication. The addition operation or bundling (denoted as $+$) is an element-wise sum: $A = B + C$, where A, B, C are HD vectors. The resultant vector is similar (in the sense of some similarity measure) to summand vectors but quasi-orthogonal (the similarity is approximately equal to zero for a cosine similarity) to others. Semantically, bundling represents a set of vectors and, correspondingly, a set of symbols.

The multiplication operation or binding (denoted as $\odot$) is an element-wise multiplication of two HD vectors: $A = B \odot C$, where A, B, C are HD vectors. It maps vectors B and C to another HD vector A. The resultant vector is dissimilar (quasi-orthogonal) to multiplied and other HD vectors from the vector space. Binding represents an attribute-value pair, an assignment of a value to a corresponding attribute.

C APPENDIX — TRAINING IDEA

It is assumed that the object O depicted on the image I can be represented as a set of N generative factors $G(O) = \{G_1(O), G_2(O), ..., G_N(O)\}$, where $G_i(O) = V_i, V_i \in V_{G_i}$, V_{G_i} is a set of possible values of a generative factor G_i.

Therefore, if we take two objects O_1 and O_2 and encode them into a set of generative factors $G^1 = G(O_1)$ and $G^2 = G(O_2)$, with each value of a generative factor G_i^j represented by a vector $p_{j,i}$ in some latent space, we want that the vectors $p_{1,i} = p_{2,i}$ if $G_i^1 = G_i^2$.

If this condition is met, then:

$$\begin{aligned} G^1 = \{G_i^j | j = 1\} = G^{1\prime} = \\ = \{G_i^j | j = \begin{cases} 2 \text{ if } G_i^1 = G_i^2, \\ 1 \text{ if } G_i^1 \neq G_i^2. \end{cases} \} \end{aligned} \tag{5}$$

If we reconstruct the original object O_1 from the set of values of generative factors $G^{1'}$, the result of reconstructing O'_1 should not differ from the original object O_1. In this way, we can construct a learning model. Due to the fact that for each training example with objects O_1 and O_2, the reconstruction possibility is symmetric for both objects, we can use two reconstructions at once O'_1 compared to O_1, and O'_2 compare to O_2. To make it easier for the model to identify the exchanged feature, it is possible to generate examples with one different generative factor for synthetic data.

D APPENDIX — DATASET GENERATION

In this paper, two datasets generated from dSprites Matthey et al. (2017) and CLEVR Johnson et al. (2017) were used to test the proposed approach.

D.1 DSPRITES PAIRED

dSprites is a dataset that contains 737280 procedurally generated 2D shapes (images of size 64×64) with the following generative factors: shape (square, ellipse, heart), scale (6 values), orientation (40 values in $[0, 2\pi]$), x and y coordinates (32 values for each).

Each training example contains two images x_1, x_2 with generative factors $(G_1^i, ..., G_5^i)$ and a feature exchange vector $e = (e_1, ..., e_5)$, where $e_i = 1 - [G_i^1 = G_i^2]$ ([ ] is the Iverson bracket). Each of the two images can be a donor for the other (Figure 1), and the feature exchange vector is used in the latent representation stage to exchange the corresponding features.

For our training set, we sampled 100,000 pairs of unique images that differ by the value of one generative factor. At the same time, we excluded from the training set images with $shape = square$ and $x > 0.5$, to check our model for compositional generalization as it was done in Montero et al. (2022). The test set was sampled without restrictions and contained 30,000 pairs of unique images that differ by a value of one generative factor and do not match the examples from the training set. The resulting dataset we call **dSprites paired** (Fig. 5a).

D.2 CLEVR1 PAIRED

CLEVR contains images of 3D-rendered objects with the following generative factors: shape, size, material type, color, x and y coordinates, and orientation (because the sphere and the cylinder are symmetric in the XY plane, it is impossible to check the result when exchanging the "orientation" feature, so for all images orientation was set to zero and excluded from the list of generative factors).

We have modified the source generation code from the original dataset Johnson et al. (2017) to create our training and test examples (10,000 and 1000, respectively). As in dSprites, each training example contains two images x_1, x_2 and the feature exchange vector e we call **CLEVR1 paired** (Fig. 5b) because the image contains one object. Camera position and lighting are random but fixed for all images of the same training example.

We used a version of CLEVR with color/shape conditions (CoGenT) to test for compositional generalization. The training set contains cubes that are gray, blue, brown, or yellow, red cylinders, green, purple, or cyan, and spheres that can be any color. In the validation set cubes and cylinders have opposite color palettes, and the test set contains all possible combinations. Additionally, the size of the generated images is 128×128 (default is 320×240), and there is no restriction on overlapping objects.

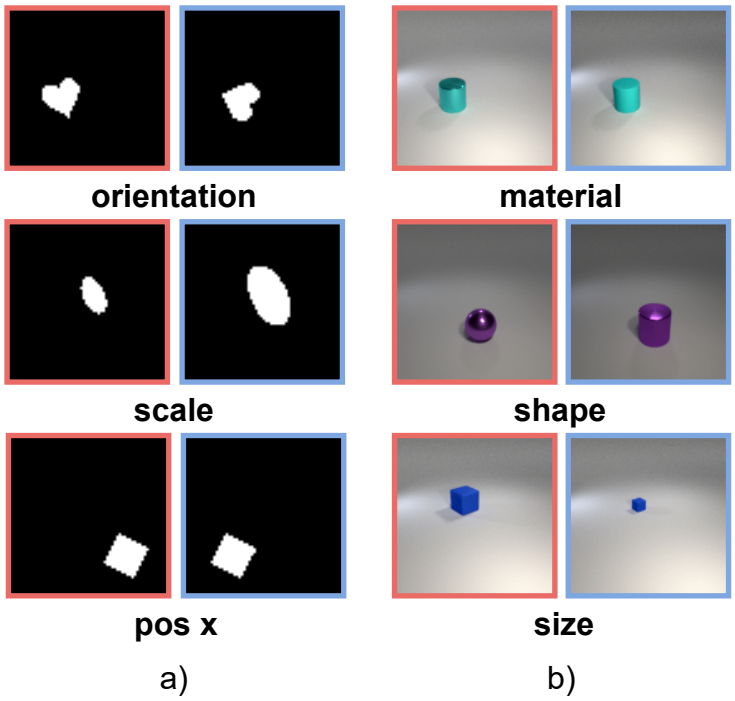

Figure 5: Examples of image pairs from the training sample of datasets: a) **dSprites paired**; b) **CLEVR1 paired**.

E Appendix — Model Architecture

We train the model using the AdamW optimizer (Loshchilov & Hutter (2017)) with a learning rate of $2.5e-5$ with other parameters set to default. We further make use of the learning rate scheduler OneCycleLR (Smith & Topin (2017)) with the percentage of the cycle spent increasing the learning rate set to 0.2. For dSprites the number of epochs was 600 with a batch size of 512 and for CLEVR1, 1000 and 64, respectively.

Table 1: Architecture of the CNN encoder for the experiments on dSprites paired and CLEVR1 datasets. An asterisk means that the layer is only used in the model for CLEVR1 paired.

Layer	Channels	Activation	Params
Conv2D 4×4	64	ReLU	stride: 2, pad: 1
Conv2D 4×4	64	ReLU	stride: 2, pad: 1
Conv2D 4×4	64	ReLU	stride: 2, pad: 1
Conv2D 4×4	64	ReLU	stride: 2, pad: 1
Conv2D 4×4 *	64	ReLU	stride: 2, pad: 1
Linear	1024	ReLU	-
Linear	1024	-	-

Table 2: Architecture of the CNN decoder for the experiments on dSprites paired and CLEVR1 datasets. An asterisk means that the layer is only used in the model for CLEVR1 paired.

Layer	Channels	Activation	Params
Linear	1024	GELU	-
Linear	1024	GELU	-
ConvTranspose2D 4×4	64	GELU	stride: 2, pad: 1
ConvTranspose2D 4×4	64	GELU	stride: 2, pad: 1
ConvTranspose2D 4×4	64	GELU	stride: 2, pad: 1
ConvTranspose2D 4×4 *	64	GELU	stride: 2, pad: 1
ConvTranspose2D 4×4	64	Sigmoid	stride: 2, pad: 1

Table 3: Architecture of the attention layer.

Layer	Channels	Activation	Params
K proj: Linear	1024	-	-
Q proj: Linear	1024	-	-
V proj: -	-	-	-

F APPENDIX — EVALUATION

Popular disentanglement metrics such as BetaVAE score Higgins et al. (2017), DCI disentanglement Eastwood & Williams (2018), MIG Chen et al. (2018), SAP score Kumar et al. (2018), and FactorVAE score Kim & Mnih (2018) are based on the assumption that disentanglement is achieved by having each individual vector coordinate capture one generative factor in the data, i.e. by changing the value in this coordinate it is possible to change the property of the object. Such representations are localist.

Due to the fact that in Hyperdimensional Computing, the representations are distributed, that is, the whole vector captures a certain generative factor, and not any of its individual coordinates, current metrics are not suitable for assessing the disentanglement of the proposed model representations. Therefore, the main results are presented qualitatively.

However, we provide quantitative results that evaluate the quality of the reconstruction of objects and scenes. We use the FID metric Heusel et al. (2017) for the CLEVR dataset and Intersection over Union (IoU) for the dSprites dataset.

G APPENDIX — QUALITATIVE RESULTS AND VISUALIZATIONS

In this section, we provide additional examples of object modification for dSprites (Fig. 6) and CLEVR (Fig. 7) datasets, as well as reconstructed images from individual feature vectors for dSprites (Fig. 8) and CLEVR (Fig. 9) datasets.

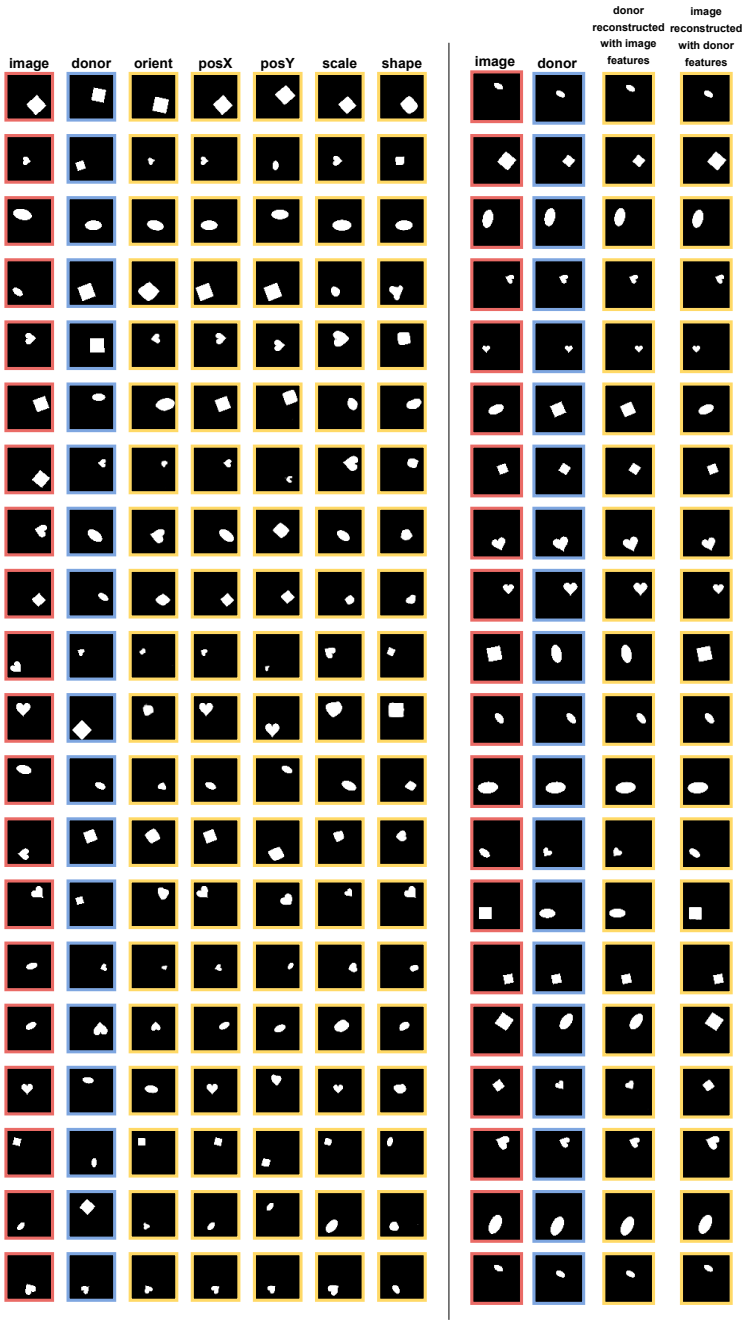

Figure 6: Examples of image reconstruction of objects from the dSprites paired dataset with modified values of generative factors. The target object (red) differs from the donor object (blue) in all factor values. The yellow frames represent the reconstruction of the target image with one of the values of the generative factor replaced by the value of the donor object: shape, position x (pos x), position y (pos y), orientation, and scale.

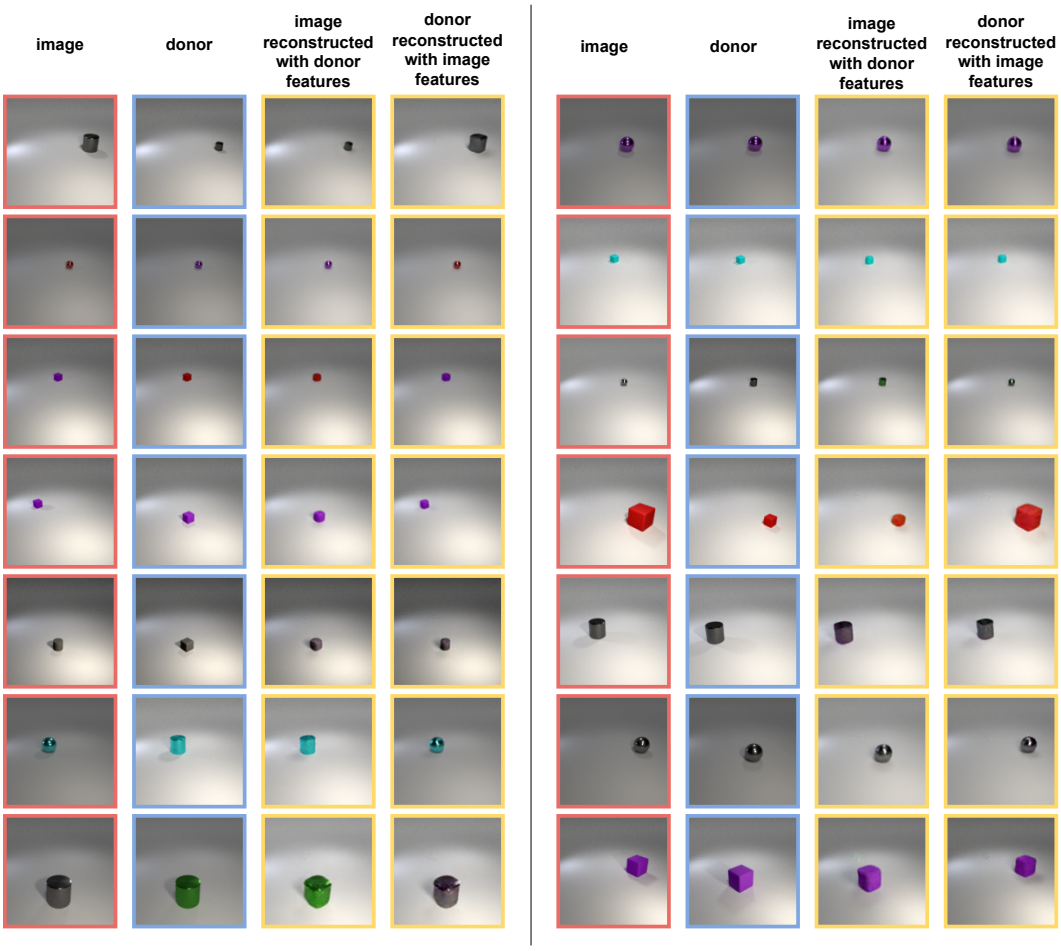

Figure 7: Examples of image reconstruction of objects from the CLEVR1 paired dataset with modified values of generative factors.

H Appendix — Ablation Studies

Since the model uses randomly sampled seed vectors that do not change during training, we tested the stability of the model for the dSprites dataset on 5 seeds. We got a mean of IoU is equal to 0.981 with a variance of 0.002.

An important hyperparameter of the model is the dimension D of the latent space. We checked how changing this value affects the IoU metric for dSprites paired and FID metric for CLEVR1 paired (Table 4). The results were obtained on 3 seeds.

To test the idea of exchanging several features at once, we did an experiment with 1-4 exchanges simultaneously and obtained an IoU metric equal to 0.989.

To test how the attention mechanism works in conjunction with the VSA, we set up the following experiment. For both datasets on the validation split, for each generative factor G_i we checked how the vector taken with the highest attention coefficient corresponds to the vector obtained by unbinding the latent scene vector O with the corresponding placeholder p_i for each of tested latent dimensions (Table 5, Table 6). Visualizations of these tables are shown in Fig. 10 and Fig. 11 correspondingly. The number of training epochs for this experiment is less than for the main experiment: 200 for dSprites paired and 600 for CLEVR1 paired.

Table 4: Metrics for both dSprites paired and CLEVR1 paired datasets calculated with different latent dimension D (mean ± stddev for 3 seeds).

Latent dim.	dSprites paired (IOU)	CLEVR1 paired (FID)
16	0.940±0.013	109.88±8.05
32	0.945±0.009	100.52±0.20
64	0.958±0.006	96.97±1.06
128	0.963±0.003	96.12±0.54
256	0.965±0.002	90.21±1.08
512	0.970±0.002	85.14±2.80
1024	0.976±0.001	80.70±0.68
2048	**0.984±0.001**	**79.07±0.86**

Table 5: Attention-unbinding accuracy metric for dSprites paired.

Latent dim.	shape	scale	orientation	posX	posY
16	0.36±0.08	0.23±0.31	0.09±0.06	0.04±0.03	0.01±0.01
32	0.61±0.01	0.18±0.18	0.11±0.02	0.03±0.02	0.03±0.02
64	0.41±0.12	0.32±0.12	0.17±0.06	0.15±0.04	0.05±0.02
128	0.30±0.09	0.47±0.10	0.36±0.02	0.14±0.02	0.17±0.08
256	0.24±0.19	0.53±0.11	0.37±0.04	0.20±0.04	0.09±0.03
512	0.22±0.09	0.52±0.16	0.36±0.08	0.18±0.06	0.19±0.06
1024	0.52±0.15	**0.70±0.07**	0.26±0.05	0.33±0.14	0.29±0.06
2048	**0.69±0.04**	0.60±0.18	**0.45±0.02**	**0.37±0.08**	**0.39±0.07**

Table 6: Attention-unbinding accuracy metric for CLEVR1 paired.

Latent dim.	shape	size	material	color	posX	posY
16	0.52±0.33	0.51±0.06	0.42±0.41	0.27±0.26	0.04±0.02	0.04±0.05
32	0.44±0.08	0.58±0.13	0.55±0.14	0.19±0.08	0.06±0.02	0.09±0.07
64	0.58±0.11	0.73±0.18	0.65±0.24	0.18±0.01	0.08±0.03	0.10±0.06
128	0.54±0.09	0.91±0.03	0.59±0.04	0.45±0.09	0.17±0.05	0.25±0.09
256	0.63±0.06	0.94±0.04	0.71±0.06	0.43±0.06	0.20±0.10	0.33±0.06
512	**0.79±0.07**	0.96±0.02	0.80±0.04	0.52±0.05	0.28±0.09	0.39±0.12
1024	0.75±0.04	**0.98±0.01**	0.89±0.07	0.64±0.03	0.37±0.10	**0.60±0.03**
2048	0.78±0.05	0.96±0.01	**0.95±0.01**	**0.70±0.02**	**0.65±0.02**	0.55±0.10

I Appendix — Additional Discussion

To train the proposed model, it is necessary to have an assumption about the number of generative factors in the data since this is a model hyperparameter. This is not a limitation when working with synthesized data, such as game and simulation environments, when the number of generative factors is known in advance. However, working with realistic datasets may require additional analysis of the data itself and determining the level of the presentation hierarchy that is worth using. For example, you can describe a person's face in terms of hairstyles and hair colors, nose shapes, eye colors, and so on, or you can go down the hierarchy of representation and, for example, describe the shape of a nose in terms of tangent inclination angles to the line of the nose.

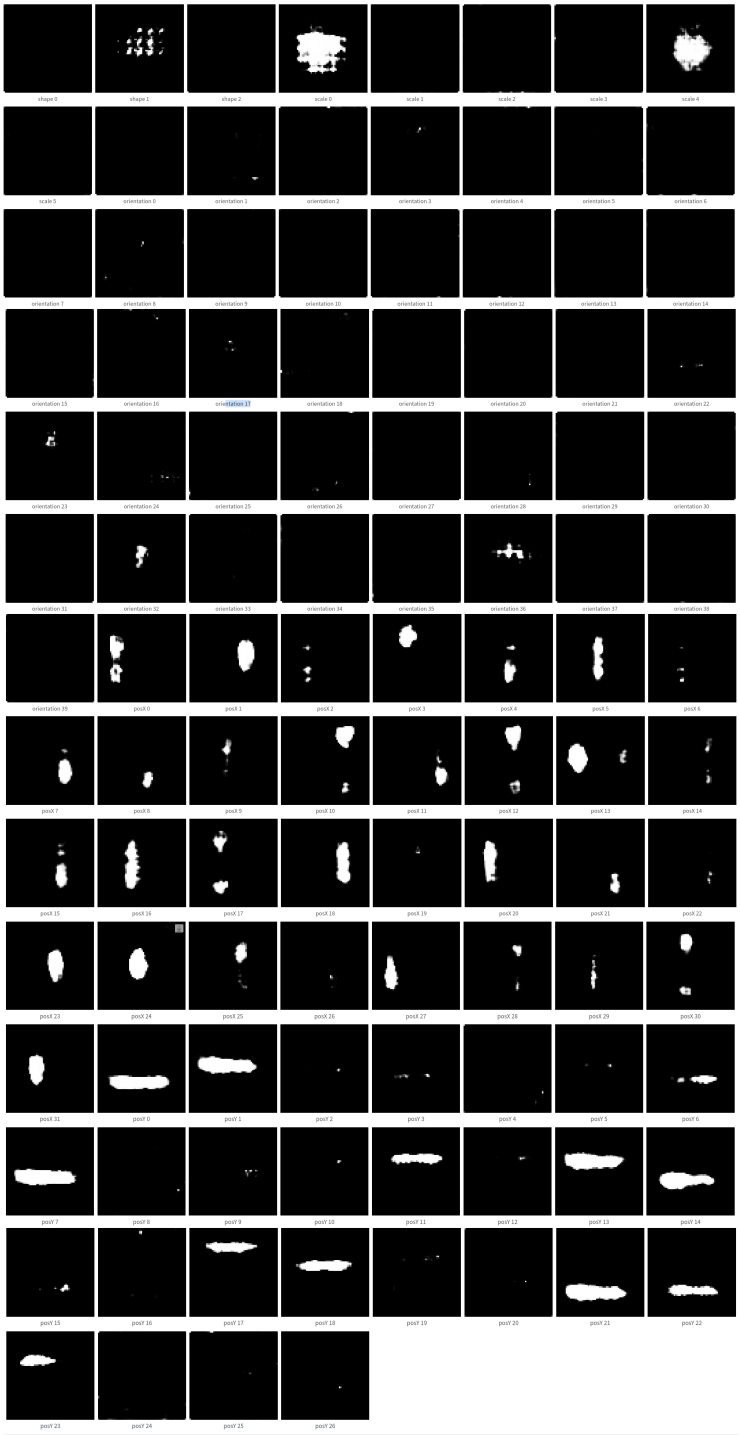

Figure 8: Visualization of image reconstruction from a single feature vector from a codebook (bound to a placeholder value) for the dSprites dataset. It shows that a complete image is not reconstructed from a single vector. This indicates that the vector represents a separate property of the object. This is particularly evident when restoring position. For example, it shows on the bottom lines, where reconstructing Pos Y, the line with the fixed position Y is reconstructed.

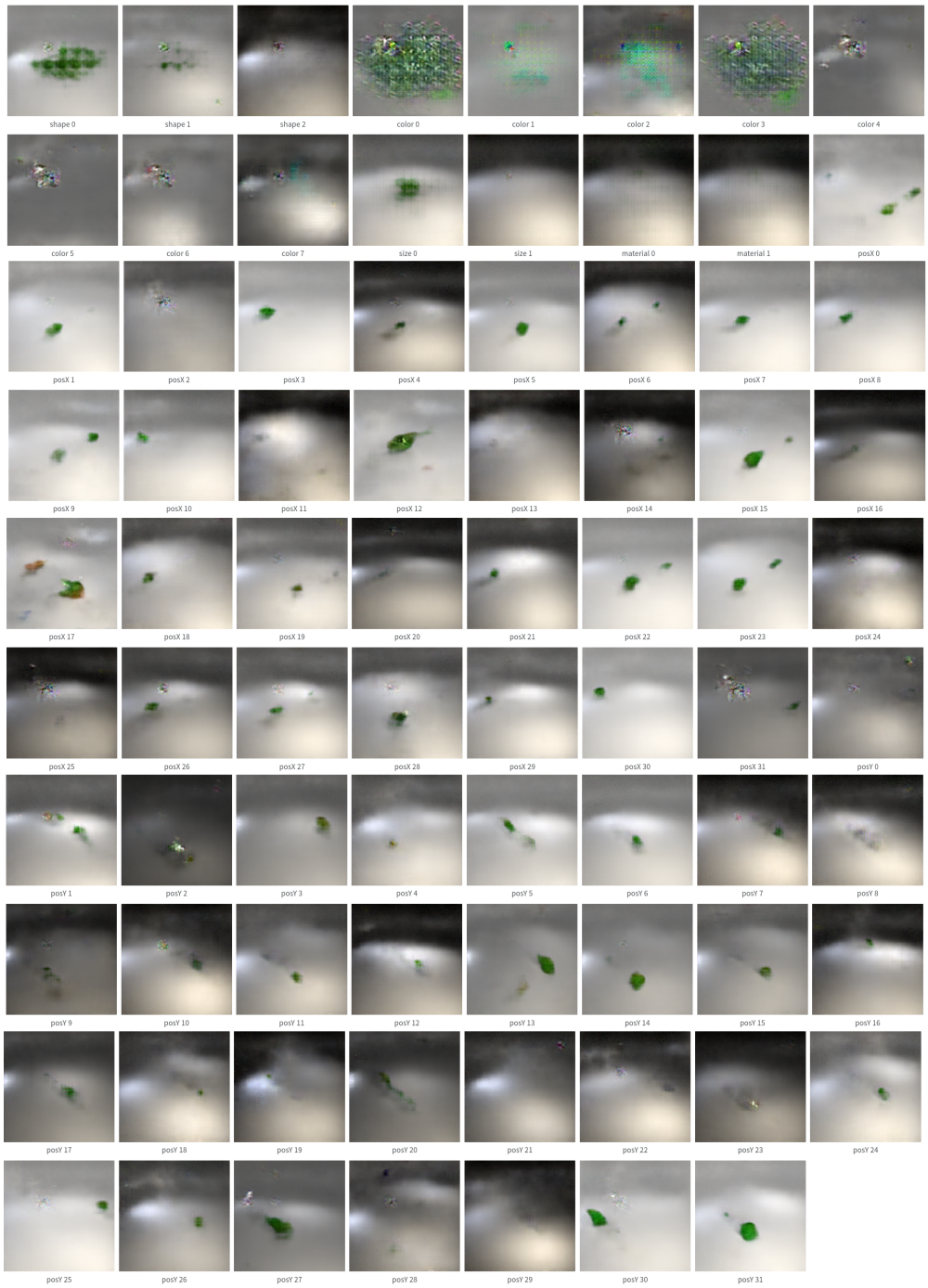

Figure 9: Visualization of image reconstruction from a single feature vector from a codebook (bound to a placeholder value) for the CLEVR dataset. It shows that a complete image is not reconstructed from a single vector. This indicates that the vector represents a separate property of the object.

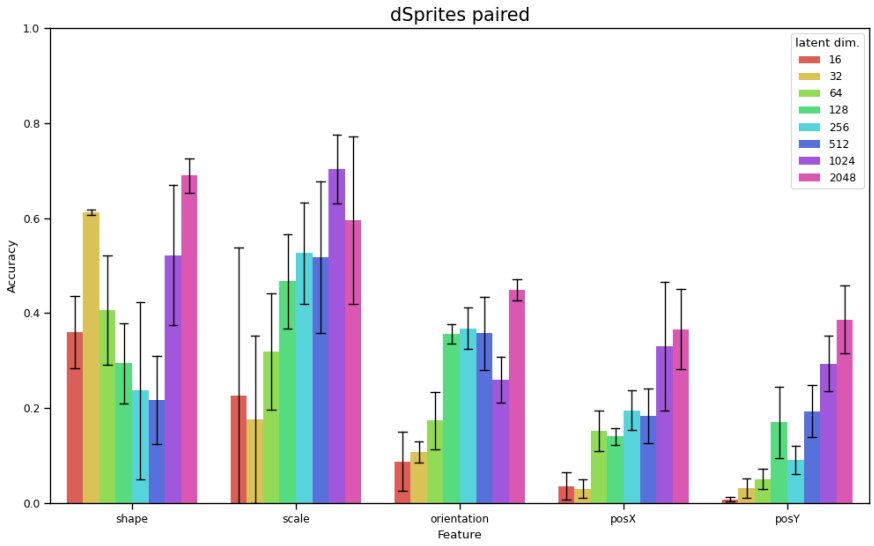

Figure 10: Attention-unbinding accuracy metric for dSprites paired.

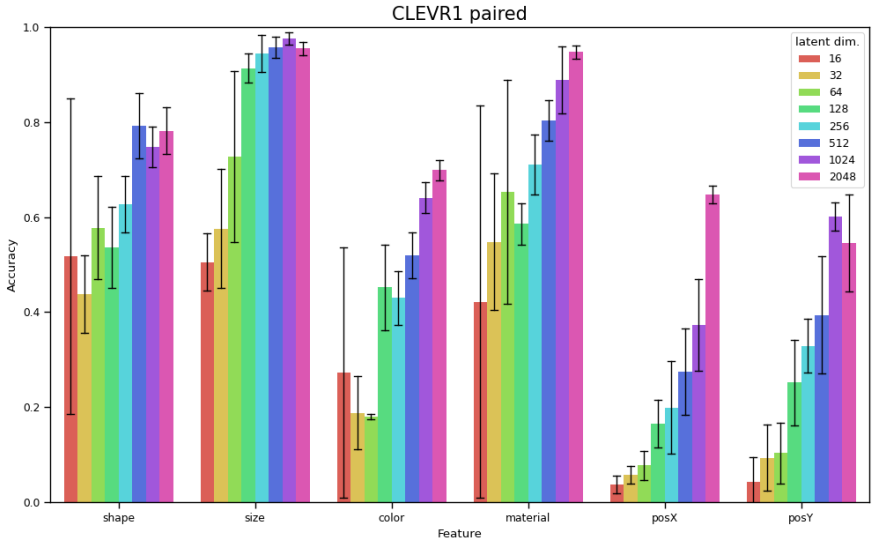

Figure 11: Attention-unbinding accuracy metric for CLEVR1 paired.