DISENTANGLEMENT WITH HOLOGRAPHIC REDUCED REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

The process of disentanglement, that focuses on separating the factors of variation in a dataset using neural networks, is a long standing challenge in machine learning. Prior solutions to this problem included the design of variational autoencoders and generative adversarial networks that use concepts from variational inference and information-theoretic constraints in their loss functions, respectively. In contrast to these prior works that rely on continuous representations in their models, we propose a new design that naturally views disentangled representations as symbolic structures due to the compositional nature of the relationship between the concepts that make up a sample from a distribution. However, learning discrete symbolic structures using traditional neural network architectures while maintaining differentiability is challenging, often requiring complex architectures to accomplish. To this end, in this paper, we propose an unsupervised learning algorithm that makes use of holographic reduced representations (HRR) for disentanglement using neural networks. We find that the unbinding operation, defined over HRR vectors, yields a suitable inductive bias for the separation of factors, and yields competitive results compared to other baselines, as measured using latent component swaps and disentanglement metrics. We complement these empirical findings with an information-theoretic analysis of the HRR unbinding channel. We prove that unbinding induces approximately independent symbol-value pairs (also called slots) and derive a per-slot capacity bound that quantifies how many distinct symbolic concepts the representation can reliably encode, providing a quantitative account of the inductive bias toward disentanglement. The disentangled representations produced in this process differ from other autoencoder based models in that the individual latent units are vectors themselves which are summed together, differing from the paradigm of latent units behaving as scalar dimensions of low dimensional vectors. We show that this type of HRR representation is more robust to noise than other disentangled representations and can maintain good reconstruction performance across a range of SNRs, while simultaneously gracefully degrading in the recoverable semantic content. Our results thus show that the use of vector symbolic architectures (VSA), such as HRRs, may hold a promising potential for representation learning, paving the way toward exploring new ways in which the symbolic benefits of VSAs can be used to represent data with neural networks.

1 INTRODUCTION

The pursuit of learning efficient representations for data has been of interest to the deep learning field for many years, and led to the emergence of the field of representation learning (Bengio et al., 2013a). Understanding representations of data within neural networks can shed light into how to construct better inductive biases to train them, and provide new insights into how neural networks interact with the data they use. Disentangled representation learning is a popular approach to the more general representation learning problem whose focus is on how to disentangle the generative factors that make up a dataset. Perhaps the earliest work on this was a more heavily regularized variational autoencoder (VAE) (Kingma & Welling, 2022), called the β-VAE (Higgins et al., 2017). Following the β-VAE, a line of work emerged that attempted to further improve on its disentanglement abilities leading to works such as the total correlation (TC) VAE (Chen et al., 2018), and the FactorVAE (Kim & Mnih, 2018), each of which build on the joint reconstruction and Kullback Leibler

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(KL) divergence objectives of the original VAE. Alongside these models, a parallel line of work proposed more comprehensive metrics for evaluating disentanglement (Ridgeway & Mozer, 2018; Eastwood & Williams, 2018). Most prior art in disentanglement has followed the VAE framework, with the exception of InfoGAN, an information-theoretic extension of generative adversarial networks (GANs) (Goodfellow et al., 2014; Chen et al., 2016). Recently, there has also been evidence to show that disentanglement can arise using discrete representations (Hsu et al., 2023), building on previous evidence that neural networks can learn discrete representations, in general (van den Oord et al., 2017). However, disentanglement does not arise on its own. Without inductive biases on both the data and the model, unsupervised disentanglement is impossible (Locatello et al., 2020). This points to the inductive bias of each method as the main driver of its disentanglement quality. The same study argues that progress depends on making the inductive bias explicit and on demonstrating benefits beyond disentanglement scores (Locatello et al., 2020), which motivates both the structure we impose on the latent and the compositional generalization study in Section 4.3. Recent methods also combine several biases at once, as in Tripod (Hsu et al., 2024), which extends latent quantization with two further regularizers and which we compare against in Appendix I. Vector symbolic architectures (VSAs) offer a structurally different inductive bias, namely a *compositional vector space*, but their potential for unsupervised disentanglement remains largely unexplored. A prior work explored this intersection and was able to disentangle generative factors using VSAs (Korchemnyi et al., 2024). However, the training procedure required carefully constructed data pairs that differ in only one generative factor at a time, limiting the solution of Korchemnyi et al. (2024) to datasets that span a sufficient number of factor combinations.

A central challenge to using VSAs in representation learning is the difficulty of mapping high-dimensional data to the structured vector spaces that VSAs provide. Mapping data to vectors using VSAs is traditionally done using hand designed methods that attempt to capture the structures apparent in the raw data (Kleyko et al., 2022; Smets et al., 2024; Imani et al., 2018; Renner et al., 2024). For example, one method of mapping images to a VSA vector involves representing the image as a sum of many vectors, each representing pixel intensity and position (Renner et al., 2024). While such methods are interpretable, they can only encode structure that the designer anticipates in advance, leaving little room for the model to discover new representations from data. On the other hand, neural networks have been shown to be able to learn strong representations of data that can be used for downstream tasks such as classification (Chen et al., 2020; Radford et al., 2016; Caron et al., 2021; Assran et al., 2023; Grill et al., 2020; He et al., 2022; Mikolov et al., 2013; Radford et al., 2021). However, using neural networks to learn VSA representations directly from data remains a relatively unexplored field. Some works have trained neural networks to map data into VSAs (Hersche et al., 2023; Ganesan et al., 2021; Alam et al., 2023b;a), but these typically rely on labeled training data (Ganesan et al., 2021), generalize poorly, and leave little room for the model to discover its own structure. The closest method to ours is Soft TPR (Sun et al., 2024), which learns compositional representations with a soft tensor product binding but trains with weak supervision on paired observations, and the HLB architecture (Alam et al., 2024) offers a real-valued alternative binding. We compare against both in Appendix I. Despite the rich literature on disentanglement and the long history of VSAs as a representation framework, there is no existing method that learns HRR-structured latent representations from data without supervision, and no existing theory that explains why an HRR structure should produce disentanglement. This paper closes this gap.

The main contribution of this paper is to demonstrate, both empirically and theoretically, that holographic reduced representations (HRRs) provide an effective inductive bias for unsupervised disentanglement, achieving competitive performance on standard benchmarks while admitting a quantitative information-theoretic account of why the bias works. In particular, our key contributions include:

- We show that training a neural network with a reconstruction objective, paired with a latent reconstruction process and an HRR structural regularizer, *produces disentangled representations of the generative factors of a dataset* without the use of any labels, unlike prior work on learning VSA representations from data (Korchemnyi et al., 2024).
- We derive a theoretical upper bound on the mutual information between the input and the HRR-structured latent, quantifying how many distinct symbolic concepts the representation can reliably encode. This bound provides a principled basis for choosing the architectural parameters HRR vector dimension, number of symbols, and the latent codebook size.

- Experimental results show that the resulting representations achieve the best aggregate scores on all six standard disentanglement metrics (InfoMEC and DCI) across four datasets, with relative improvements of roughly 6 to 8% over the next-best baseline on InfoM, InfoC, and DCI D and C, while exhibiting noise robustness close to VQ-VAE and considerably better than other disentanglement methods. HRR is the only model that achieves both superior disentanglement and strong noise robustness on the studied benchmarks. Qualitative analyses of latent component swapping further confirm that the learned representation aligns with individual generative factors.

2 PRELIMINARIES OF HRR VECTORS

HRRs are a VSA that represents discrete entities, called *symbols*, as high-dimensional random vectors, and represents structured combinations of those entities through algebraic operations on the vectors. Based on Plate (1995), an HRR vector $\mathbf{x} \in \mathbb{R}^d$ is typically initialized with entries sampled independently as $x_i \sim \mathcal{N}(0, 1/d)$, so that distinct randomly initialized vectors are approximately orthogonal for large d.

HRRs define two main operations, *binding* and *bundling*. Binding, denoted $\otimes$, associates two vectors through circular convolution, producing a representation that is approximately dissimilar to either input. In practice, this operation can be computed efficiently as multiplication in the frequency domain. Bundling, denoted $\oplus$, superimposes vectors through addition and is commonly used to represent sets or compositions of symbols. A useful property of HRRs is approximate unbinding. Given a bound representation $\mathbf{z} = \mathbf{x} \otimes \mathbf{y}$, one can recover a noisy estimate of $\mathbf{x}$ by applying an approximate inverse of $\mathbf{y} : \hat{\mathbf{x}} = \mathbf{y}^\dagger \otimes \mathbf{z} \approx \mathbf{x}$. The approximation error is often called *unbinding noise*. Here, "inverse" follows the terminology of the HRR literature rather than denoting an exact algebraic inverse; we defer further details to the appendix and refer readers to prior work on HRR inverses (Ganesan et al., 2021), and the original HRR paper (Plate, 1995).

3 HRRS FOR DISENTANGLEMENT

Disentangled representations aim to capture the structure within a dataset. Although continuous representations, and more recently discrete representations (Hsu et al., 2023; 2024), have been shown to separate some of this structure, we will show that symbolic representations may be better suited for this task. Symbolic structures are useful for representing data with compositional structure, where entities, attributes, and relations combine in meaningful ways. For example, consider an image of a dog in a backyard. Although the image is represented at the pixel level, we naturally interpret the scene in terms of entities, attributes, and relations. There is a dog, the dog is in a backyard, and the dog has attributes such as breed, age, color, and size. This interpretation can be organized as a simple symbolic structure, where the root node captures the main entity, *dog*, and its associated attributes are represented as role-filler pairs such as *breed: Chihuahua* or *size: small*. In a VSA, this kind of structure can be encoded by binding each role vector to its corresponding filler vector and bundling the resulting pairs into a single distributed representation. While this example uses only one level of role-filler binding, more complex hierarchies can be expressed through recursive composition (Plate, 1995). The flat latent representations used in prior continuous and discrete disentanglement work (Hsu et al., 2023; 2024) do not naturally express this kind of compositional structure. Here, we present a proof of concept that a neural network can be trained to produce such structured vector representations directly from data.

HRRs allow us to impose such a structure on a neural network's latent space using a hetero-associative memory, where pairs of vectors are stored such that one element of the pair can be retrieved using the other as a cue. We define an HRR structured representation as a vector in the form $\mathbf{z} = (\mathbf{s}_1 \otimes \mathbf{v}_1) \oplus \cdots \oplus (\mathbf{s}_m \otimes \mathbf{v}_m)$. Each $\mathbf{s}_i$ and $\mathbf{v}_i$ is an HRR vector, where $\otimes$ and $\oplus$ are the binding and bundling operations over HRRs. We refer to the $\mathbf{s}_i$ as symbol vectors, which act as the cues to the memory and to $\mathbf{v}_i$ as value vectors, which act as the items stored in the memory. We refer to the number of symbol-value pairs m as the symbolic length of the representation. The encoder is trained to produce $\mathbf{z}$ from the data $\mathbf{x}$. Ideally, the model learns to associate each $\mathbf{s}_i$ with the most important generative factors of the data. For example, on images of chairs, $\mathbf{s}_1$ could represent chair type (sofa, stool, recliner, etc.), $\mathbf{s}_2$ chair pose, and $\mathbf{s}_3$ chair size; the corresponding $\mathbf{v}_i$ would

then encode the specific value of each factor (e.g., $\mathbf{v}_1$ for "sofa"). Each bound pair $\mathbf{s}_i \otimes \mathbf{v}_i$ thus represents the realization of one generative factor, and the bundled sum $\mathbf{z} = \bigoplus_{i=1}^{m} \mathbf{s}_i \otimes \mathbf{v}_i$ forms a neuro-symbolic representation of $\mathbf{x}$. The symbolic length m also controls how finely the model can separate generative factors. A larger m provides more symbol-value pairs in which to place distinct factors, while a smaller m forces the model to group factors together.

3.1 Model architecture

Fundamentally, our model is a CNN based autoencoder that learns to map the data, $\mathbf{x}$, directly to the latent representation, $\mathbf{z}$, with an HRR unbinding and vector quantization step. Along with the autoencoder we initialize a set of m symbol vectors, $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m\}$, and a codebook of $k \gg m$ value vectors, $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k\}$. The vectors in $\mathcal{S}$ are randomly initialized from the HRR distribution described in Section 2 and remain frozen throughout training. The codebook $\mathcal{C}$ is also initialized randomly but not from the HRR distribution as the codebook is optimized throughout the training process.

The encoder first predicts the latent representation, $\mathbf{z}$, from the data. From the latent representation, we retrieve the vectors in the hetero-associative memory through the unbinding operation defined in Section 2 by binding the latent once with the inverse of each of the vectors in $\mathcal{S}$. This retrieval produces the noisy value vectors $\{\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \ldots, \tilde{\mathbf{v}}_m\}$, which each get processed by a small feedforward neural network, referred to as a denoising network, to produce $\{\bar{\mathbf{v}}_1, \bar{\mathbf{v}}_2, \ldots, \bar{\mathbf{v}}_m\}$. From here, we do vector quantization as described in van den Oord et al. (2017) to map each $\bar{\mathbf{v}}_i$ to its nearest Euclidean neighbor in the codebook $\mathcal{C}$, resulting in the final set of value vectors $\{\hat{\mathbf{v}}_1, \ldots, \hat{\mathbf{v}}_m\}$, where each $\hat{\mathbf{v}}_i \in \mathcal{C}$. We will write $\bar{\mathbf{V}} \in \mathbb{R}^{m \times d}$ and $\mathbf{C} \in \mathbb{R}^{k \times d}$ for the matrices whose rows are the denoised retrievals and the codebook entries, respectively. We bind each $\hat{\mathbf{v}}_i$ with the corresponding symbol vector that it originally corresponded with to construct $\hat{\mathbf{z}} = (\mathbf{s}_1 \otimes \hat{\mathbf{v}}_1) \oplus \cdots \oplus (\mathbf{s}_m \otimes \hat{\mathbf{v}}_m)$. The decoder then uses $\hat{\mathbf{z}}$ to reconstruct the original input $\mathbf{x}$.

3.2 Regularization and optimization

For a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with rows $\mathbf{x}_i$, we define the structural regularizer

$$\mathcal{R}(\mathbf{X};\, \tau_n, \tau_v) = \left(\frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i\|^2 - \tau_n \right)^2 + \left(\mathrm{mean}(\mathbf{X})\right)^2 + \left(\mathrm{Var}(\mathbf{X}) - \tau_v\right)^2, \tag{1}$$

where mean and Var are computed globally over all entries. This regularizer pushes a matrix toward zero mean, target per-row squared norm τ_n, and target per-entry variance τ_v, matching the statistical properties expected of HRR vectors and their bundled compositions (Appendix C). We apply $\mathcal{R}$ to three different matrices, with targets chosen so that each matches the relevant HRR distribution, as shown in Table 1.

Table 1: Specifications of the three structural regularizers.

Term	$\mathbf{X}$	n	τ_n	τ_v
$\mathcal{L}_{\text{latent}}$	$\mathbf{z}^\top$	1	m	m/d
$\mathcal{L}_{\text{value}}$	$\bar{\mathbf{V}}$	m	1	$1/d$
$\mathcal{L}_{\text{codebook}}$	$\mathbf{C}$	k	1	$1/d$

The targets $\tau_v = 1/d$ for $\bar{\mathbf{V}}$ and $\mathbf{C}$ reproduce the per-component variance of HRR vectors (Section 2); the targets for $\mathbf{z}$ ($\tau_n = m$, $\tau_v = m/d$) match the expected squared norm and per-component variance of a sum of m bound HRR pairs (Appendix C). Because the vector quantization step is non-differentiable, we use the straight-through estimator (Bengio et al., 2013b) to propagate gradients through it.

Loss function The model is trained with an unsupervised auto-encoding objective, using the AdamW optimization algorithm (Loshchilov & Hutter, 2019). The total loss combines a binary cross-entropy reconstruction loss $\mathcal{L}_{\text{recon}}$, the vector quantization (VQ) and commitment losses from

the original VQ-VAE (van den Oord et al., 2017), and the three structural regularizers defined above. Following van den Oord et al. (2017), we define

$$\mathcal{L}_{\text{VQ}} = \|\text{sg}[\bar{\mathbf{v}}] - \mathbf{c}\|^2,\ \mathcal{L}_{\text{commit}} = \|\bar{\mathbf{v}} - \text{sg}[\mathbf{c}]\|^2,$$

where $\text{sg}[\cdot]$ is the stop gradient operator, $\bar{\mathbf{v}}$ is the denoised retrieval, and $\mathbf{c}$ is the nearest Euclidean neighbor in the codebook $\mathcal{C}$. $\mathcal{L}_{\text{VQ}}$ allows gradients to reach the codebook, while $\mathcal{L}_{\text{commit}}$ allows gradients to reach the encoder. Both terms ensure the codebook and the encoder work together to converge on mutually optimal codebook entries and latent predictions, respectively. The total objective is

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{recon}} + \lambda_{\text{vq}}\mathcal{L}_{\text{VQ}} + \lambda_c\mathcal{L}_{\text{commit}}}_{\text{reconstruction}} + \underbrace{\lambda_{\text{lat}}\mathcal{L}_{\text{latent}} + \lambda_{\text{val}}\mathcal{L}_{\text{value}} + \lambda_{\text{cb}}\mathcal{L}_{\text{codebook}}}_{\text{regularization}}, \tag{2}$$

where all $\lambda \geq 0$ are scalar weights. The hyperparameters used for all runs are listed in Table 6.

3.3 Capacity of the HRR latent channel

Having specified the model and its training objective, we now turn to an information-theoretic analysis of the HRR latent channel, which clarifies how much information the symbolic latent $\hat{\mathbf{z}}$ can carry about the input $\mathbf{x}$ and motivates the design choices made in our experiments. In what follows, we refer to the i-th symbol-value pair $(\mathbf{s}_i, \mathbf{v}_i)$ as the i-th slot. The unbinding analysis in Appendix C shows that, when retrieving the value vector bound to slot i, the result decomposes as

$$\tilde{\mathbf{v}}_i \ = \ (1+\xi)\,\mathbf{v}_i \ + \ \boldsymbol{\eta}_i \ + \sum_{k\neq i} \boldsymbol{\lambda}_i^{(k)}, \tag{3}$$

with intrinsic noise $\boldsymbol{\eta}_i$ and cross-talk $\boldsymbol{\lambda}_i^{(k)}$ from the other $m-1$ bound pairs. Treating $\xi \to 0$ in expectation, the per-component signal power is $1/d$ and the per-component noise power is m/d, giving an SNR of $1/m$ per dimension. Each retrieval thus passes through d parallel AWGN sub-channels, each with the same SNR. We use this to bound the total information the symbolic latent $\hat{\mathbf{z}}$ can carry about the input $\mathbf{x}$.

Proposition 1. *Let $\hat{\mathbf{z}} = \sum_{i=1}^{m} \mathbf{s}_i \otimes \hat{\mathbf{v}}_i$ be the symbolic latent, where each $\hat{\mathbf{v}}_i$ is the quantized value selected from a codebook of size k. Under the unbinding noise model of Appendix C, the mutual information between input and latent satisfies*

$$\mathbb{I}(\mathbf{x};\,\hat{\mathbf{z}}) \ \leq \ \min\Big(\, m \cdot \tfrac{d}{2}\log\!\big(1+\tfrac{1}{m}\big),\ m\log k \,\Big). \tag{4}$$

The proof can be found in Appendix D.6.

Slot–dimension tradeoff The continuous part of the bound in (4) is $\frac{md}{2}\log(1+1/m)$, or equivalently $\frac{d}{2}\log(1+1/m)$ per slot. Thus, increasing the number of slots m provides more slots, but also lowers the effective per-slot SNR because each retrieved slot receives cross-talk from the other slots. For large m, $\log(1+1/m) \approx 1/m$, so the total continuous capacity approaches $\frac{d}{2}$. In this regime, adding more slots does not substantially increase the total amount of information that can be carried by the HRR latent, rather, it primarily divides the available capacity across more noisier retrieval channels.

This formalizes the empirical observation that unbinding performance degrades as m increases at fixed d (Appendix F). It also motivates using a modest latent-to-source overcomplete ratio in our experiments, as the number of slots should be large enough to cover the generative factors, but not so large that each slot's effective retrieval channel becomes dominated by cross-talk noise. This motivates the approximately $1.5\times$ latent-to-source ratio used in Section 4.1.

Codebook sizing The second term, $m\log k$, is the maximum discrete rate available after quantization. When $\log k < \frac{d}{2}\log(1+1/m)$, the codebook resolution is the active bottleneck, and increasing k can increase the representational rate. Once $\log k \geq \frac{d}{2}\log(1+1/m)$, the HRR retrieval channel

becomes the bottleneck, so further increasing k cannot increase the leading-order capacity bound. The codebook perplexity curves in Figure 4 are consistent with this interpretation. Harder datasets such as Isaac3D and MPI3D-C drive higher codebook perplexity, suggesting that these datasets require more discrete states per slot and are therefore more likely to operate near the codebook-resolution constraint.

3.4 Approximate slot independence

Proposition 2 (Approximate slot independence)**.** *Let $q(\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_m \mid \mathbf{x})$ be the joint distribution of unbound retrievals under the HRR noise model of Appendix C, and let $q_i(\tilde{\mathbf{v}}_i \mid \mathbf{x})$ be its marginals. With posterior variance floor $t^2 > 0$,*

$$D_{\mathrm{KL}}\big(q(\tilde{\mathbf{V}}_{1:m} \mid \mathbf{x}) \,\big\|\, \textstyle\prod_{i=1}^{m} q_i(\tilde{\mathbf{v}}_i \mid \mathbf{x})\big) \;=\; \mathcal{O}\Big(\tfrac{m^2}{t^4 d}\Big)\,, \tag{5}$$

which vanishes as $d \to \infty$ at fixed m and fixed $t > 0$.

The proof can be found in Appendix D.1. Together, Propositions 1 and 2 give a partial answer to the question of *why* the HRR structure favors disentangled representations. Two mechanisms act on the encoder. First, slots are statistically near-independent under unbinding (Proposition 2), so the encoder can optimize each slot largely independently of the others, rather than coping with cross-slot noise correlations. Second, attaining the capacity bound of Proposition 1 requires non-redundant slot usage by placing distinct generative factors in distinct slots. Quantization then discretizes each slot's content and pushes the encoder toward a finite, factor-aligned set of codes.

4 Experimental results and analysis

In the experiments, we will quantify the disentanglement performance of our proposed HRR model compared to previous models proposed for disentanglement, show that different slots in our representation correspond to different generative factors, and show the robustness to noise of our HRR representation. All together, these results demonstrate that HRR-inspired representations can achieve good disentanglement quality while offering a robustness to noise not seen in prior disentanglement works.

4.1 Disentanglement performance

We measure our model's disentanglement performance using the InfoMEC metrics, proposed in a prior work (Hsu et al., 2023), because it uses information-theoretic metrics, mutual information, to measure disentanglement quality as opposed to using feature importance from learned classifiers and regression models, as done in DCI, an older disentanglement metric (Eastwood & Williams, 2018). However we still report the DCI scores to complement our InfoMEC scores. For consistency we use the same baselines that were proposed (Hsu et al., 2023). Specifically, our baselines include, the β-VAE (Higgins et al., 2017), the TC-VAE (Chen et al., 2018), the VQ-VAE (van den Oord et al., 2017), and QLAE (Hsu et al., 2023). Consistent with Hsu et al. (2023), we use Shapes3D (Burgess & Kim, 2018), MPI3D (Gondal et al., 2019) (specifically the complex real world shape variant), and NVIDIA's Falcor3D and Isaac3D datasets (NVl, 2025). Consistent with prior works (Hsu et al., 2023), for each baseline we sweep a single hyperparameter for each (model, dataset) pair and use the best performing hyperparameter over two seeds. For our HRR model we tuned the regularization weights and gradient clipping with a random search at a single seed. All baselines and our HRR model initialize the latent units to be $1.5\times$ the ground truth number of generative factors, which determined m in our experiments. For more details on the sweep, we refer to Table 7 of Appendix E. We note the best performing hyperparameter and re-run each model on each dataset with its best hyperparameter 5 times from which we construct a confidence interval. As is consistent with other disentanglement works, we train on the entire dataset without holding out a validation or test set (Locatello et al., 2020). The purpose of this is that, in disentanglement, we are mostly concerned with how the model can learn to disentangle the different latent factors, not combating overfitting.

For each run, we sample a random subset of the dataset (without removing it from the training data) that we use to evaluate the disentanglement metrics. To ensure fairness, each model uses the same encoder and decoder architecture, and the same optimization parameters, consistent with the work of

Table 2: Disentanglement scores over $n = 5$ seeds. InfoMEC cells show (InfoM InfoE InfoC); bold marks the best per column using the CI-overlap criterion, and the highest mean in the Aggregated column. Aggregated is the mean across all datasets.

Model	Aggregated	Shapes3D	Falcor3D	Isaac3D	MPI3D-C
	InfoMEC := (InfoM InfoE InfoC) ↑				
β-VAE	(0.54 0.57 0.45)	(0.59 0.81 0.47)	(**0.64** 0.68 **0.59**)	(0.54 0.56 0.46)	(0.37 0.23 0.29)
β-TCVAE	(0.56 0.63 0.51)	(0.61 0.83 0.51)	(**0.59** 0.65 **0.55**)	(0.63 0.62 **0.57**)	(0.41 0.42 0.39)
VQ-VAE	(0.57 0.88 0.47)	(0.58 **1.00** 0.44)	(**0.50 0.86** 0.40)	(0.70 **0.86** 0.52)	(0.49 0.82 0.53)
QLAE	(0.65 0.84 0.49)	(0.77 **1.00** 0.52)	(**0.61** 0.79 **0.46**)	(0.66 0.81 0.49)	(**0.56** 0.78 0.48)
HRR (ours)	(**0.69 0.89 0.55**)	(**0.85 1.00 0.61**)	(**0.64** 0.81 **0.47**)	(**0.72 0.89 0.60**)	(0.53 **0.85 0.54**)
	DCI := (D I C) ↑				
β-VAE	(0.36 0.81 0.33)	(0.64 **1.00 0.55**)	(0.40 0.83 **0.39**)	(0.31 0.84 0.30)	(0.08 0.57 0.10)
β-TCVAE	(0.38 0.85 0.35)	(0.59 **0.98 0.49**)	(0.30 0.79 **0.29**)	(0.35 **0.87** 0.34)	(0.27 0.75 0.26)
VQ-VAE	(0.63 0.86 0.35)	(0.74 **0.98** 0.35)	(**0.64** 0.81 **0.34**)	(**0.64 0.91 0.36**)	(**0.49** 0.73 0.34)
QLAE	(0.59 0.88 0.31)	(0.76 **0.99** 0.35)	(0.50 0.84 **0.28**)	(0.59 **0.92** 0.31)	(**0.51 0.78** 0.29)
HRR (ours)	(**0.67 0.89 0.37**)	(**0.87 1.00 0.40**)	(0.58 **0.86 0.37**)	(**0.68 0.93 0.38**)	(**0.55 0.78 0.35**)

Locatello et al. (2020). More details, such as model architecture and optimization hyperparameters can be found in Table 4 and Table 5 in Appendix E. Details on the bolding criteria for the results can also be found in Appendix E.

As seen in Table 2, our HRR based disentanglement method outperforms most baselines across the InfoM metric, with a higher mean from QLAE on the MPI3D-C dataset. Our model is also able to match or exceed baselines on InfoE and InfoC scores on all datasets except on Falcor3D for the InfoE and InfoC metrics, where the VQ-VAE and β-VAE reach the highest scores, respectively. However, in aggregate, our method outperforms other baselines on InfoM and InfoC and meets VQ-VAE on InfoE, providing a 6.2% improvement in InfoM, and 7.8% improvement in InfoC, over the second highest scores in the respective categories, computed from the values in Table 2. The pattern across metrics is consistent with the analysis of Section 3.3. HRR's gains concentrate on InfoM (modularity) and InfoC (compactness), the two metrics that measure whether each factor occupies its own slot, which is consistent with the approximate slot independence of Proposition 2. On InfoE (explicitness) HRR matches VQ-VAE in aggregate, and on Falcor3D VQ-VAE and β-VAE lead on InfoE and InfoC respectively, which indicates that explicitness depends more on the decoder and the codebook size than on the slot structure.

4.2 Latent robustness to noise when decoding

Another quantity that we measure is the robustness to noise of the different latent representations. This provides insight into how much a latent representation could be corrupted by external noise while still being able to be accurately decoded. To this end, we measure the reconstruction peak-signal-to-noise-ratio (PSNR) of representations under zero mean Gaussian noise, at varying signal-to-noise ratios (SNR). We apply noise to the latent representation from +20 dB SNR down to -20 dB SNR and measure the PSNR between the resulting reconstruction and the input. As seen in Fig. 1, the proposed HRR model and VQ-VAE models both outperform the other baselines, with VQ-VAE reaching the highest PSNR at most noise levels. This is likely because VQ-VAE's optimization allows for codebook entries to spread apart in Euclidean space. A noise perturbation must therefore be large enough to push a quantized vector closer to a different codebook entry before the discrete code changes, leaving the recovered code unchanged for moderate noise levels. Fig. 8 in Appendix K shows the reconstructions under each noise level to provide a qualitative assessment of the noise robustness.

4.3 Compositional generalization

We evaluate compositional generalization with the recombination-to-range protocol of Montero et al. (2021), which withholds all combinations of a subset of factors during training while every factor value still appears, so each model is tested on seen values in unseen combinations. We use a broad condition on Shapes3D and MPI3D-C that withholds 12.5% of each dataset and a narrow condition on MPI3D-C that withholds a single pairing of two factors (6.25%). Appendix J gives the exact

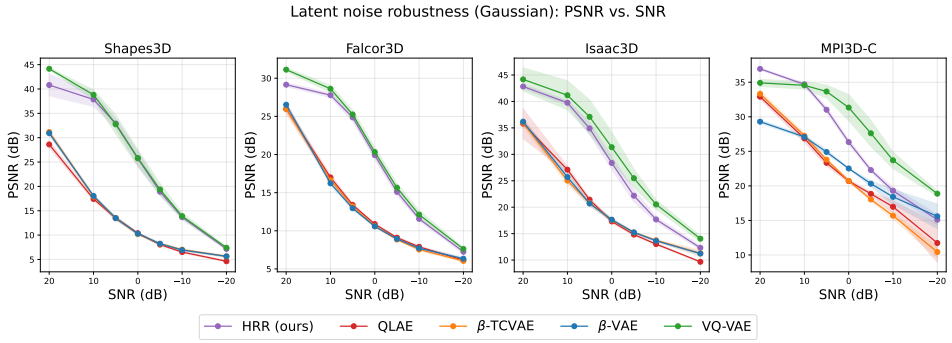


Figure 1: Average performance of each model, as measured by PSNR, under varying levels of Gaussian noise intensity. Shaded regions represent a 95% confidence interval.

Table 3: Held-out linear probe score at each model's bottleneck, pooled over factors and averaged over five seeds, with the change from the shared encoder output in parentheses. Confidence intervals are in Table 29. **Bold** marks the highest probe score per condition.

Model	Broad, Shapes3D	Broad, MPI3D-C	Narrow, MPI3D-C
HRR (ours)	**0.830** (+0.020)	**0.720** (+0.020)	**0.774** (-0.106)
VQ-VAE	0.675 (-0.125)	0.631 (-0.098)	0.745 (-0.195)
QLAE	0.679 (-0.042)	0.585 (-0.113)	0.634 (-0.281)
β-TCVAE	0.532 (-0.279)	0.464 (-0.132)	0.514 (-0.284)

partitions. After training, following Xu et al. (2022), we fit linear probes on 500 random training images and compute the probe score on each factor over five seeds, and we call the in-distribution score minus the held-out score the compositional deficit. We probe two points in each model, the output of the shared convolutional encoder, which has the same size for every model, and the model's own latent representation, which we call its bottleneck, so the change between them measures how much predictive information the projection to the latent loses.

As seen in Table 3, our HRR based model scores highest on held-out factor combinations at the bottleneck in every condition, with a 22.2% higher probe score than the next best model on Shapes3D, 14.1% higher on MPI3D-C under the broad condition, and 3.9% higher under the narrow condition. The per-family probe scores and the compositional deficits behind these aggregates are given in Appendix J.

HRR's advantage on held-out combinations comes from its bottleneck, the latent representation each model learns between the shared encoder and its decoder. All four models share the same convolutional encoder, and the change in parentheses in Table 3 is the difference between the probe score at the bottleneck and the probe score at that shared encoder output. HRR's held-out probe score rises by 0.020 in both broad conditions and falls by 0.106 under the narrow condition, while every baseline loses between 0.042 and 0.284. VQ-VAE, the dimensionality-matched control, has 576 or 640 dimensions against HRR's 512 (Table 28) and still performs behind HRR in every condition, so dimensionality alone does not account for the difference in performance.

4.4 Latent component swapping

Because of the vector quantization process that assigns value vectors to each slot in our latent variables, traditional latent traversals (where scalar dimensions are varied through an interval) cannot be used as a fair qualitative method of assessment for our method. Instead, we opt to show the results of what happens when you swap latent units between two representations from the dataset, i.e, inject a latent factor from one representation into the other. For truly disentangled representations, it would be reasonable to expect a degree of modularity (in the qualitative sense, not the quantitative one used as a metric) between the representations, such that the latent units are compositional.

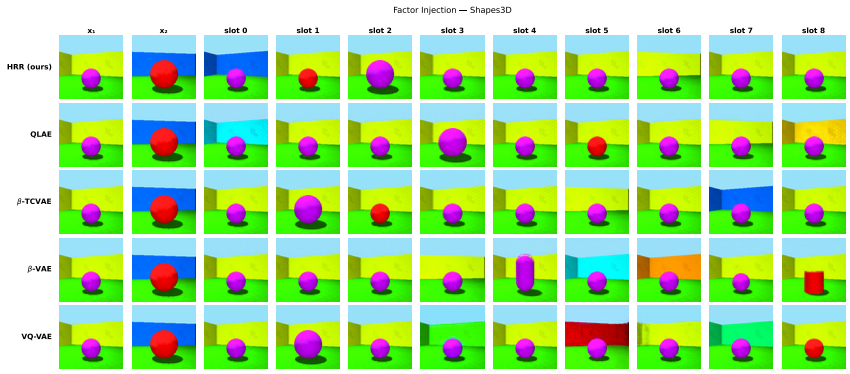


Figure 2: Latent component swaps for the seed with the highest InfoM score. Column slot i replaces the i-th slot of x_1 with that of x_2 and decodes the result, with the slot restored afterwards, so each decoded image should change at most one aspect of x_1.

Fig. 2 shows latent component swaps for HRR and other baselines. We make the following observations here. First, HRR's swaps consistently modify a single generative factor at a time while leaving others intact. Where QLAE's swaps occasionally produce visible artifacts in unrelated factors, HRR maintains the unswapped attributes more cleanly. Second, HRR preserves global scene semantics across nearly all swap positions, while QLAE occasionally alters the orientation of objects during a swap. The concentration of changes within a single generative factor follows from the approximate slot independence established in Proposition 2. If the per-slot retrievals were tightly coupled, swapping one slot would propagate changes across others, but HRR's slot-by-slot swapping remains contained. The qualitative gap with QLAE is informative because both methods quantize their latents but only HRR imposes a symbolic binding structure, suggesting that the binding-and-bundling architecture may be important to enable single-factor edits. Appendix M extends the swaps experiment to cumulative slot replacement, which walks from one image to another, one factor at a time.

5 Discussion

In this paper, we have proposed using a compositional latent structure with HRRs for disentanglement and shown that the resulting representations match or exceed strong baselines in aggregate on the InfoMEC and DCI metrics across four standard datasets while being more robust to additive latent noise than every disentanglement-focused baseline at nearly every noise level. To the best of our knowledge, this is the first work to use HRRs for disentanglement without supervision, and the first to train a neural network to produce HRR-structured latents directly from data. Sections 3.3 and 3.4 provide a partial mechanism for why the HRR structure favors disentanglement. Approximate slot independence under the unbinding channel (Proposition 2) lets the encoder optimize each slot near-independently, while the per-slot capacity bound (Proposition 1) penalizes redundant slot usage. Together with the quantization inductive bias identified in Hsu et al. (2023), these two pressures favor placing distinct generative factors in distinct slots. These results are a good foundation for explaining the disentanglement that emerges from the proposed HRR-based model. Tighter constants, a formal connection to identifiability, and an empirical study of the d/m ratio are natural next steps that would further validate these results. Beyond disentanglement, our results indicate that VSA-inspired representations are worth investigating more broadly in the deep learning field. The robustness to latent noise observed here may be particularly relevant for emerging use cases, such as physical-AI. In physical settings, representations must tolerate the noise that is fundamental in sensors and wireless channels. Robustness will be an important factor in ensuring representations remain useful despite corruption. Finally, compositional generalization, which prior work has noted as a weakness of standard disentanglement methods (Montero et al., 2021), is tested in Section 4.3. We find that our HRR based method retains more probe score on withheld factor combinations than the bottlenecks of the baselines.

AI USE STATEMENT

In this work, we used generative AI tools to provide ingredients for proving mathematical claims and to assist in the writing of proofs, to design or provide feedback on experiments, to implement methods, and to interpret results. We have not used generative AI tools to generate synthetic data sets, to develop theoretical models or conceptual frameworks, to formulate mathematical claims, or to propose or refine hypotheses, and translation, dataset cleaning and reformatting, and qualitative or thematic data analysis are not applicable to this work. We have reviewed all AI-assisted work. LLM-generated code was tested for correctness by the authors, and every LLM-recommended reference was read by the authors before it was cited. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix E describes the architecture, optimization settings, hyperparameters, sweep protocol, evaluation metrics and compute for every reported run. Appendix A describes the denoiser and codebook generator, and Appendix B gives the training procedure as pseudocode. Appendix D contains complete proofs of the theoretical results, with their assumptions. The supplementary material contains anonymized code and configuration files for training and evaluating the five models of Table 2 on the four datasets.

REFERENCES

NVlabs/High-res-disentanglement-datasets. NVIDIA Research Projects, November 2025.

Mohammad Mahmudul Alam, Edward Raff, Stella Biderman, Tim Oates, and James Holt. Recasting Self-Attention with Holographic Reduced Representations. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 490–507. PMLR, July 2023a.

Mohammad Mahmudul Alam, Edward Raff, and Tim Oates. Towards generalization in subitizing with neuro-symbolic loss using holographic reduced representations. In *Neuro-symbolic learning and reasoning in the era of large language models*, 2023b. URL https://openreview.net/forum?id=AOAP8sLYdt.

Mohammad Mahmudul Alam, Alexander Oberle, Edward Raff, Stella Biderman, Tim Oates, and James Holt. A walsh hadamard derived linear vector symbolic architecture. In *The thirty-eighth annual conference on neural information processing systems*, 2024. URL https://openreview.net/forum?id=p3hNrpeWMe.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, June 2023. doi: 10.1109/CVPR52729.2023.01499.

Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, August 2013a. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2013.50.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013b.

Chris Burgess and Hyunjik Kim. 3D shapes dataset, 2018.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, October 2021. doi: 10.1109/ICCV48922.2021.00951.

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

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 1597–1607. PMLR, November 2020.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

Ashwinkumar Ganesan, Hang Gao, Sunil Gandhi, Edward Raff, Tim Oates, James Holt, and Mark McLean. Learning with Holographic Reduced Representations. In *Advances in Neural Information Processing Systems*, volume 34, pp. 25606–25620. Curran Associates, Inc., 2021.

Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché Buc, E. Fox, and R. Garnett (eds.), *Advances in neural information processing systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/d97d404b6119214e4a7018391195240a-Paper.pdf.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284. Curran Associates, Inc., 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, June 2022. doi: 10.1109/CVPR52688.2022.01553.

Michael Hersche, Mustafa Zeqiri, Luca Benini, Abu Sebastian, and Abbas Rahimi. A neuro-vector-symbolic architecture for solving Raven's progressive matrices. *Nature Machine Intelligence*, 5(4): 363–375, March 2023. ISSN 2522-5839. doi: 10.1038/s42256-023-00630-8.

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.

Kyle Hsu, Will Dorrell, James C. R. Whittington, Jiajun Wu, and Chelsea Finn. Disentanglement via latent quantization. In *Thirty-Seventh Conference on Neural Information Processing Systems*, 2023.

Kyle Hsu, Jubayer Ibn Hamid, Kaylee Burns, Chelsea Finn, and Jiajun Wu. Tripod: Three complementary inductive biases for disentangled representation learning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, Vienna, Austria, 2024. JMLR.org.

Mohsen Imani, Chenyu Huang, Deqian Kong, and Tajana Rosing. Hierarchical hyperdimensional computing for energy efficient classification. In *Proceedings of the 55th Annual Design Automation Conference*, Dac '18, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 978-1-4503-5700-5. doi: 10.1145/3195970.3196060.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2649–2658. PMLR, July 2018.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2022.

Denis Kleyko, Dmitri A. Rachkovskij, Evgeny Osipov, and Abbas Rahimi. A Survey on Hyperdimensional Computing aka Vector Symbolic Architectures, Part I: Models and Data Transformations. *ACM Comput. Surv.*, 55(6):130:1–130:40, December 2022. ISSN 0360-0300. doi: 10.1145/3538531.

Alexandr Korchemnyi, Alexey K. Kovalev, and Aleksandr I. Panov. Symbolic disentangled representations for images, 2024.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. A sober look at the unsupervised learning of disentangled representations and their evaluation. *Journal of Machine Learning Research*, 21(209):1–62, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, September 2013. URL http://arxiv.org/abs/1301.3781. arXiv:1301.3781 [cs].

Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

T.A. Plate. Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3): 623–641, May 1995. ISSN 1941-0093. doi: 10.1109/72.377968.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 8748–8763. PMLR, July 2021.

Alpha Renner, Lazar Supic, Andreea Danielescu, Giacomo Indiveri, Bruno A. Olshausen, Yulia Sandamirskaya, Friedrich T. Sommer, and E. Paxon Frady. Neuromorphic visual scene understanding with resonator networks. *Nature Machine Intelligence*, 6(6):641–652, June 2024. ISSN 2522-5839. doi: 10.1038/s42256-024-00848-0. URL https://www.nature.com/articles/s42256-024-00848-0.

Karl Ridgeway and Michael C Mozer. Learning Deep Disentangled Embeddings With the F-Statistic Loss. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Gautam Singh, Yeongbin Kim, and Sungjin Ahn. Neural systematic binder. In *The Eleventh International Conference on Learning Representations*, 2023.

Laura Smets, Werner Van Leekwijck, Ing Jyh Tsang, and Steven Latré. An encoding framework for binarized images using hyperdimensional computing. *Frontiers in Big Data*, 7, June 2024. ISSN 2624-909X. doi: 10.3389/fdata.2024.1371518.

Bethia Sun, Maurice Pagnucco, and Yang Song. Fully Distributed, Flexible Compositional Visual Representations via Soft Tensor Products. In *Advances in Neural Information Processing Systems*, volume 37, pp. 139470–139560. Curran Associates, Inc., 2024. doi: 10.52202/079017-4427. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/fbf1fe4819b359f5715c4989b9df3725-Abstract-Conference.html.

Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Zhenlin Xu, Marc Niethammer, and Colin Raffel. Compositional generalization in unsupervised compositional representation learning: A study on disentanglement and emergent language. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

A ADDITIONAL DETAILS ON MODEL DESIGN

The denoising network described in Section 3.1 is implemented as a single hidden layer feed forward neural network, with a hidden layer size that is $2\times$ the size of the input and that uses leaky relu activation. The codebook design was inspired from a prior work in slot attention (Singh et al., 2023). Specifically, we maintain k "seed" vectors of dimension d, same as $\mathbf{z}$, and generate the codebook by processing each seed vector through a single hidden layer feed forward neural network with a hidden layer size that is the same size as the input and also uses leaky relu activation. We take the output of this neural network to be the approximate-HRR codebook vectors which we regularize.

A step that we took to improve training stability was the inclusion of multiplication constant at the output of the encoder and input to the decoder. Given the structural inductive biases imposed on the encoder, directly producing a zero-mean latent with m/d per-component variance would push the model towards having small weights, potentially leading to training instability and vanishing gradients. Thus, we multiply the output of the encoder by $\sqrt{m/d}$ to ensure a higher variance output by the encoder. Similarly, since the reconstructed latent will also have approximately m/d variance, we multiply the input to the decoder by $\sqrt{d/m}$ to transform it to have approximately unit variance for the decoder's input. We treat the output of the codebook generator the same way as the output of the encoder, and apply a $\sqrt{1/d}$ scaling to the produced codebook vectors. Although we observe in Figure 3 of Appendix C.2 that the empirical variance during training is not exactly as assumed, we found that including the normalization constant improved training stability, but other choices may also have the same effect.

B ALGORITHM DETAILS

This section covers more details on our training algorithm that we presented in Section 3. Algorithm 1 describes the latent reconstruction process that occurs at the bottleneck of the autoencoder. Algorithm 2 references Algorithm 1 in the broader training loop.

Algorithm 1 HRR-VQ: Slot Retrieval, Denoising, and Quantization

Require: Normalized latent $\mathbf{z} \in \mathbb{R}^d$, frozen symbol vectors $\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_m\}$, codebook $\mathbf{C} \in \mathbb{R}^{k\times d}$ with rows $\mathbf{c}_1, \ldots, \mathbf{c}_k$, denoising network $h : \mathbb{R}^d \to \mathbb{R}^d$

Ensure: Quantized latent $\hat{\mathbf{z}}$, quantized value vectors $\{\hat{\mathbf{v}}_i\}$, denoised retrievals $\{\bar{\mathbf{v}}_i\}$, assignment indices $\{j_i\}$

for $i = 1, \ldots, m$ **do**
  $\tilde{\mathbf{v}}_i \leftarrow \mathbf{s}_i^{-1} \otimes \mathbf{z}$ ▷ HRR unbind: retrieve noisy value vector for slot i
end for
for $i = 1, \ldots, m$ **do**
  $\bar{\mathbf{v}}_i \leftarrow h(\tilde{\mathbf{v}}_i)$ ▷ Denoising network
  $j_i \leftarrow \arg\min_j \|\bar{\mathbf{v}}_i - \mathbf{c}_j\|^2$ ▷ Nearest-codebook-vector lookup
  $\hat{\mathbf{v}}_i \leftarrow \mathbf{c}_{j_i}$ ▷ Quantized value vector
  $\mathbf{v}_i^{\mathrm{st}} \leftarrow \bar{\mathbf{v}}_i + \mathrm{sg}(\hat{\mathbf{v}}_i - \bar{\mathbf{v}}_i)$ ▷ Straight-through estimator
end for
$\hat{\mathbf{z}} \leftarrow \sum_{i=1}^{m} \mathbf{s}_i \otimes \mathbf{v}_i^{\mathrm{st}}$ ▷ Rebind and bundle all slots
return $\hat{\mathbf{z}},\ \{\hat{\mathbf{v}}_i\},\ \{\bar{\mathbf{v}}_i\},\ \{j_i\}$

Algorithm 2 Pseudocode for optimizing the HRR Autoencoder

Require: Dataset $\mathcal{D}$, batch size b, AdamW hyperparameters $(\alpha, \beta_1, \beta_2, \mathrm{wd})$, loss weights $\boldsymbol{\lambda} \coloneqq (\lambda_r,\ \lambda_{\mathrm{vq}},\ \beta,\ \lambda_z,\ \lambda_c)$, number of symbols m, codebook size k, latent dimension d

initialize CNN encoder $f : \mathcal{X} \to \mathbb{R}^d$, CNN decoder $g : \mathbb{R}^d \to \mathcal{X}$, denoising network $h : \mathbb{R}^d \to \mathbb{R}^d$

initialize seed matrix $\boldsymbol{\Theta} \in \mathbb{R}^{k \times d}$, codebook generator $\varphi : \mathbb{R}^d \to \mathbb{R}^d$

sample and freeze $\mathcal{S} = \{\mathbf{s}_i\}_{i=1}^m, \quad \mathbf{s}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0},\ \frac{1}{d}\mathbf{I}_d)$ ▷ Random HRR symbol vectors; never updated

while $(f, g, h, \boldsymbol{\Theta}, \varphi)$ has not converged **do**

  for $n = 1, \ldots, b$ **do**

    $\mathbf{x} \sim \mathcal{D}$

    $\mathbf{z} \leftarrow f(\mathbf{x}) \cdot \sqrt{m/d}$ ▷ Encode and normalize latent variance

    $\mathbf{C} \leftarrow \varphi(\boldsymbol{\Theta}) \cdot \sqrt{1/d}$ ▷ Generate codebook from seed matrix

    $\hat{\mathbf{z}},\ \{\hat{\mathbf{v}}_i\},\ \{\bar{\mathbf{v}}_i\},\ \{j_i\} \leftarrow \text{HRR-VQ}(\mathbf{z},\ \mathcal{S},\ \mathbf{C},\ h)$ ▷ Algorithm 1

    $\hat{\mathbf{x}} \leftarrow g(\hat{\mathbf{z}} \cdot \sqrt{d/m})$ ▷ Denormalize and decode

    $\mathcal{L}_{\mathrm{recon}} \leftarrow \lambda_r \cdot \mathrm{BCE}(\hat{\mathbf{x}},\ \mathbf{x})$

    $\mathcal{L}_{\mathrm{vq}} \leftarrow \lambda_{\mathrm{vq}} \cdot \frac{1}{m} \sum_i \left\| \hat{\mathbf{v}}_i - \mathrm{sg}(\bar{\mathbf{v}}_i) \right\|^2$ ▷ Codebook toward denoised retrievals

    $\mathcal{L}_{\mathrm{commit}} \leftarrow \beta \cdot \frac{1}{m} \sum_i \left\| \mathrm{sg}(\hat{\mathbf{v}}_i) - \bar{\mathbf{v}}_i \right\|^2$ ▷ Encoder commits to codebook

    $R_z \leftarrow \mathbb{E}[\mathbf{z}]^2 + \left(\mathrm{Var}(\mathbf{z}) - \frac{m}{d}\right)^2 + \left(\|\mathbf{z}\|^2 - m\right)^2$

    $R_v \leftarrow \mathbb{E}[\bar{\mathbf{v}}]^2 + \left(\mathrm{Var}(\bar{\mathbf{v}}) - \frac{1}{d}\right)^2 + \left(\frac{1}{m} \sum_i \|\bar{\mathbf{v}}_i\|^2 - 1\right)^2$

    $R_c \leftarrow \mathbb{E}[\mathbf{C}]^2 + \left(\mathrm{Var}(\mathbf{C}) - \frac{1}{d}\right)^2 + \left(\frac{1}{k} \sum_j \|\mathbf{c}_j\|^2 - 1\right)^2$

    $\mathcal{L}_{\mathrm{reg}} \leftarrow \lambda_z (R_z + R_v) + \lambda_c \cdot R_c$

    $\mathcal{L}^{(n)} \leftarrow \mathcal{L}_{\mathrm{recon}} + \mathcal{L}_{\mathrm{vq}} + \mathcal{L}_{\mathrm{commit}} + \mathcal{L}_{\mathrm{reg}}$

  end for

  $(f, g, h) \leftarrow \mathrm{AdamW}\left(\nabla_{f,g,h} \frac{1}{b} \sum_n \mathcal{L}^{(n)},\ (f, g, h),\ \alpha,\ \beta_1,\ \beta_2,\ \mathrm{wd}\right)$

  $(\boldsymbol{\Theta}, \varphi) \leftarrow \mathrm{AdamW}\left(\nabla_{\boldsymbol{\Theta},\varphi} \frac{1}{b} \sum_n \mathcal{L}^{(n)},\ (\boldsymbol{\Theta}, \varphi),\ 2\alpha,\ \beta_1,\ \beta_2,\ \mathrm{wd}\right)$ ▷ Codebook LR $= 2\alpha$

end while

C PROPERTIES AND STRUCTURE OF THE HRR

In the following sections, we show that retrieval through the unbinding operation over HRRs introduces two terms that describe noise from different sources within the representation, and show that the particular variance used to initialize the HRR is not an essential component to preserving per-component SNR.

C.1 DISTRIBUTION OF A RETRIEVAL

It is possible to derive the distribution of a retrieval from the unbinding operation. This can be done through the circular correlation of unbinding. A version of this example was originally shown in Plate (1995) and also revisited in Ganesan et al. (2021). Under the circular correlation definition of binding (circular convolution with the inverse would be equivalent),

$$\mathbf{c}^\dagger \otimes (\mathbf{c} \otimes \mathbf{x}) = \begin{bmatrix} x_0(c_0^2 + c_1^2 + c_2^2) + x_1 c_0 c_2 + x_2 c_0 c_1 + x_1 c_0 c_1 \\ +x_2 c_1 c_2 + x_1 c_1 c_2 + x_2 c_0 c_2 \\ x_1(c_0^2 + c_1^2 + c_2^2) + x_0 c_0 c_1 + x_2 c_0 c_2 + x_0 c_0 c_2 \\ +x_2 c_1 c_2 + x_0 c_1 c_2 + x_2 c_0 c_1 \\ x_2(c_0^2 + c_1^2 + c_2^2) + x_0 c_0 c_1 + x_1 c_1 c_2 + x_0 c_1 c_2 \\ +x_1 c_0 c_2 + x_0 c_0 c_2 + x_1 c_0 c_1 \end{bmatrix}$$

$$= \begin{bmatrix} x_0(1+\xi) + \eta_0 \\ x_1(1+\xi) + \eta_1 \\ x_2(1+\xi) + \eta_2 \end{bmatrix} = (1+\xi)\,\mathbf{x} + \boldsymbol{\eta}.$$

Here, $\xi = (c_0^2 + \cdots + c_d^2) - 1$. Through the central limit theorem and by assuming independence between c_i and x_i, we can show that $\xi \sim \mathcal{N}(0, 2/d)$, and $\eta_i = \mathcal{N}(0, (d-1)/d^2)$. For the average case, the $1 + \xi$ term can be taken to 1, since $\mathbb{E}\left[\|\mathbf{c}\|^2\right] = 1$, which leaves the final result to be our intended retrieval target plus a noise term.

These results can be extended to our case for retrieval from a sum of bound vectors. We add another pair to the quantity from the previous example, $\mathbf{c}^\dagger \otimes [(\mathbf{c} \otimes \mathbf{x}) + (\mathbf{d} \otimes \mathbf{y})]$. For compactness, denote $w_i = (\mathbf{c} \otimes \mathbf{x})_i$, and $v_i = (\mathbf{d} \otimes \mathbf{y})_i$. The retrieval can be expressed as below.

$$\mathbf{c}^\dagger \otimes [(\mathbf{c} \otimes \mathbf{x}) + (\mathbf{d} \otimes \mathbf{y})] = \begin{bmatrix} c_0(w_0 + v_0) + c_1(w_1 + v_1) + c_2(w_2 + v_2) \\ \cdots \\ \cdots \end{bmatrix}$$

$$= \begin{bmatrix} c_0(c_0 x_0 + c_1 x_2 + c_2 x_1 + d_0 y_0 + d_1 y_2 + d_2 y_1) \\ +c_1(c_0 x_1 + c_1 x_0 + c_2 x_2 + d_0 y_1 + d_1 y_0 + d_2 y_2 \\ +c_2(c_0 x_2 + c_1 x_1 + c_2 x_0 + d_0 y_2 + d_1 y_1 + d_2 y_0 \\ \cdots \\ \cdots \end{bmatrix}$$

$$= \begin{bmatrix} (c_0^2 + c_1^2 + c_2^2)\, x_0 \\ +c_0 c_1 x_2 + c_0 c_2 x_1 + c_0 d_0 y_0 + c_0 d_1 y_2 + c_0 d_2 y_1 \\ +c_0 c_1 x_1 + c_1 c_2 x_2 + c_1 d_0 y_1 + c_1 d_1 y_0 + c_1 d_2 y_2 \\ +c_0 c_2 x_2 + c_1 c_2 x_1 + c_2 d_0 y_2 + c_2 d_1 y_1 + c_2 d_2 y_0 \\ \cdots \\ \cdots \end{bmatrix}$$

$$= \begin{bmatrix} x_0(1+\xi) + \eta_0 + \lambda_0 \\ x_1(1+\xi) + \eta_1 + \lambda_1 \\ x_2(1+\xi) + \eta_2 + \lambda_2 \end{bmatrix}$$

$$= (1+\xi)\mathbf{x} + \boldsymbol{\eta} + \boldsymbol{\lambda}.$$

The final expression, for the $m = 2$ case, results in the same expression for the $m = 1$ case, with an extra term for the cross talk noise due to unbinding. Each λ_i has d^2 terms, each like $c_j d_k y_l$. Assuming

independence between terms, $c_j d_k y_l \sim \mathcal{N}(0, 1/d^3)$. Since there are d^2 terms, $\lambda_i = \mathcal{N}(0, 1/d)$. In general, for the retrieval of a vector from a latent $\mathbf{z}$ with m bound terms,

$$\hat{\mathbf{x}} = (\mathbf{c}^{(i)})^\dagger \otimes \mathbf{z} = \underbrace{(1+\xi)\mathbf{x}}_{\text{retrieval}} + \underbrace{\boldsymbol{\eta}}_{\text{intrinsic noise}} + \underbrace{\sum_{k \neq i}^{m} \boldsymbol{\lambda}^{(k)}}_{\text{cross-talk noise}},$$

where $\mathbf{c}^{(i)}$ is the cue associated with a target we are trying to retrieve, and $\boldsymbol{\lambda}^{(k)}$ is the cross talk term due to the k'th bound pair in the representation.

Thus, we can interpret the retrieval as being composed of 3 main parts, the intended retrieval target, the intrinsic noise, and the cross talk noise. Since our signal and noise terms are both zero mean, we can quantify the per component signal power as $S = 1/d$, and the per component noise power as $N = \frac{d-1}{d^2} + \frac{m-1}{d} = \frac{m}{d} - \frac{1}{d^2}$. For large d, we have $N \approx \frac{m}{d}$. Then the per component SNR for a retrieval is

$$\text{SNR} = \frac{1/d}{m/d} = \frac{1}{m}.$$

C.2 Importance of the HRR initialization

The analysis in Appendix C.1 shows that the zero-mean requirement for the HRR vectors ensures that all the noise terms are also zero-mean. The per component variance of $1/d$ ensures that the expected squared norm of HRR vectors is one. This allows us to ensure that the norm of a representation does not grow unbounded as it is bound with more vectors. This was a desirable property for the original intended purpose of the HRR, which was to be able to create representations of arbitrary depth (Plate, 1995). For our case, this requirement is not strictly necessary since we only use HRR vectors to create bundles of m bound pairs, not for binding arbitrary amounts. If we were to instead consider the case in which HRR vector components were instead sampled from $\mathcal{N}(0, c)$, where $c > 0$. Then we can find that the binding variance becomes

$$\text{Var}((a \otimes b)_i) = d \cdot c^2.$$

For retrieval, the per component cross talk variances becomes

$$\text{Var}(\lambda_i) = md^2c^3,$$

and the intrinsic noise becomes

$$\text{Var}(\eta_i) = d(d-1)c^3 \approx d^2 \cdot c^3.$$

The per component signal power becomes $S = \text{Var}((c_0^2 + \cdots + c_d^2) \cdot x_i) = (2dc^2 + c^2d^2) \cdot c \approx c^2d^2 \cdot c$, for $d \gg c$, and thus the SNR is

$$\text{SNR} = \frac{d^2c^3}{(m-1)d^2c^3 + d^2c^3} = \frac{d^2c^3}{md^2c^3} = \frac{1}{m},$$

thus showing that the per component SNR actually remains unchanged with respect to the variance of the atomic HRR vectors. This shows that the $1/d$ variance requirement from the original HRR implementation may not be necessary for our purposes, and thus any reasonable choice, c for the per component variance may also work. However, the original choice of variance is retained for consistency with the original HRR implementation and other works using the HRR. The results in Fig. 3 support this finding. The model learns to keep the zero mean requirement as it is essential for the unbinding process to work. The variances drift from the $1/d$ constant associated with the HRR, but this finding supports the analysis above, as the $1/d$ requirement is not strict. This leaves room for

the model to converge on a separate variance that is more optimal for that particular dataset, which we believe is what the figure reflects.

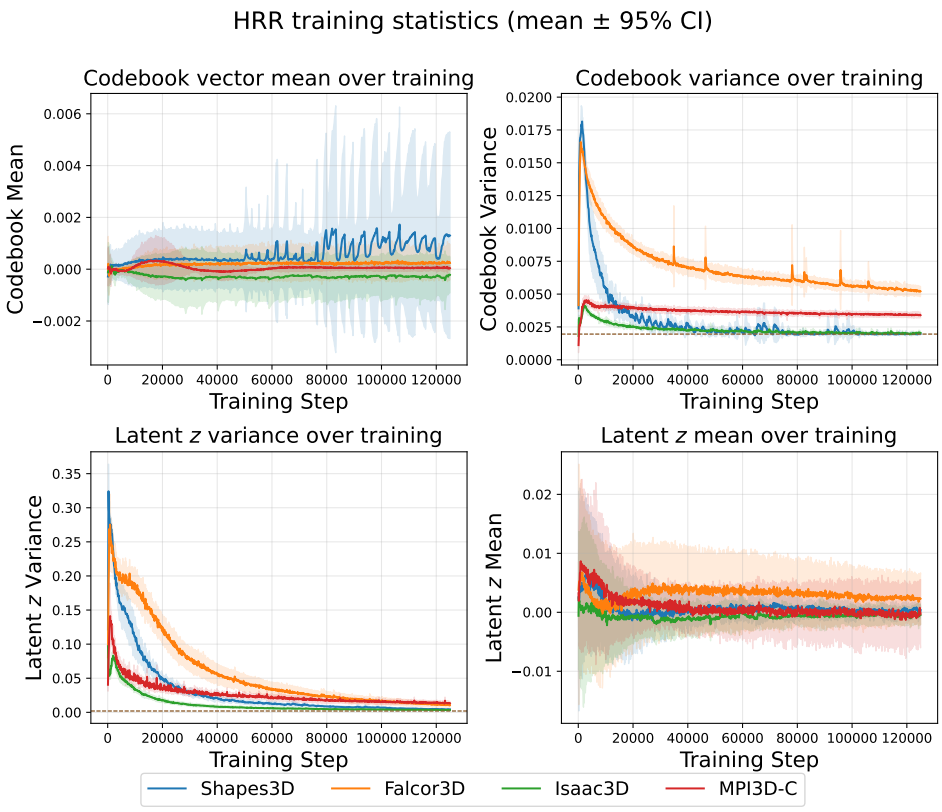


Figure 3: Mean and variance over training iterations, for both the codebook vectors and the latent representations. Most statistics converge on stable values as training progresses.

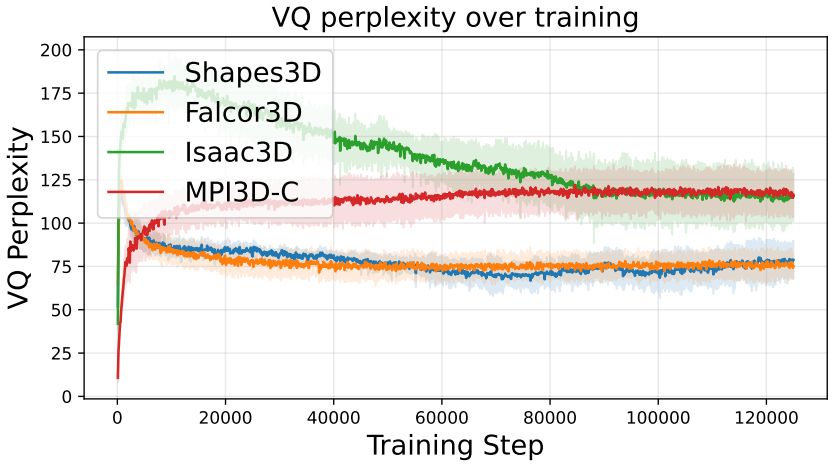


Figure 4: Codebook usage over training iterations, measured by perplexity. We can see that the model learns to utilize the codebook vectors, with the harder datasets having higher codebook usage.

D PROOFS OF THEORETICAL RESULTS

Next, we prove Propositions 1 and 2 from Sections 3.3 and 3.4. In this section, expectations are with respect to the HRR initialization distribution (each component i.i.d. $\mathcal{N}(0, 1/d)$) and the data distribution, as appropriate. We write $\tilde{\mathbf{V}}_{1:m}$ for the tuple $(\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_m)$ and $\tilde{\mathbf{V}}_{<i}$ for $(\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_{i-1})$.

D.1 PROOF OF PROPOSITION 2 (APPROXIMATE SLOT INDEPENDENCE)

We write $q(\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_m \mid \mathbf{x})$ for the joint distribution of the unbound retrievals conditional on an input $\mathbf{x}$, where the randomness is induced by the HRR initialization of the symbol vectors $\mathbf{s}_i \in \mathcal{S}$. Although these symbols are fixed after model initialization, treating them as random in the analysis allows us to quantify the typical dependence induced by HRR cross-talk.

By the chain rule of probability,

$$\log \frac{q(\tilde{\mathbf{V}}_{1:m} \mid \mathbf{x})}{\prod_i q_i(\tilde{\mathbf{v}}_i \mid \mathbf{x})} \;=\; \sum_{i=2}^{m} \log \frac{q_i(\tilde{\mathbf{v}}_i \mid \tilde{\mathbf{V}}_{<i}, \mathbf{x})}{q_i(\tilde{\mathbf{v}}_i \mid \mathbf{x})}. \tag{6}$$

The term where $i = 1$ is identically zero since the conditioning set is empty. Taking expectations under $q(\tilde{\mathbf{V}}_{1:m} \mid \mathbf{x})$,

$$D_{\mathrm{KL}}\big(q \,\|\, \textstyle\prod_i q_i\big) \;=\; \sum_{i=2}^{m} \mathbb{E}_q\Big[\log \tfrac{q_i(\tilde{\mathbf{v}}_i \mid \tilde{\mathbf{V}}_{<i}, \mathbf{x})}{q_i(\tilde{\mathbf{v}}_i \mid \mathbf{x})}\Big] \;=\; \sum_{i=2}^{m} \mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big)\,. \tag{7}$$

We bound the right-hand side of (7) by treating the total correlation as a single aggregate quantity, rather than bounding each term $\mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big)$ individually. As we discuss in Section D.4, bounding each term in isolation leads to expressions involving the smallest eigenvalue of a conditioning block, which is not accessible from the operator norms we can establish. Instead, we work with the closed-form KL representation of the full sum and bound it through a single log-determinant expression. This approach requires an understanding of the block covariance structure of the joint retrieval $\tilde{\mathbf{V}}_{1:m}$, which we develop next.

D.2 COVARIANCE BLOCK STRUCTURE

The mean of the joint retrieval distribution $q(\tilde{\mathbf{V}}_{1:m} \mid \mathbf{x})$ is $(\mathbf{v}_1(\mathbf{x}), \ldots, \mathbf{v}_m(\mathbf{x}))$. The block covariance matrix $\Sigma \in \mathbb{R}^{md \times md}$, with block entries

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \cdots & \Sigma_{1m} \\ \Sigma_{21} & \Sigma_{22} & \cdots & \Sigma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{m1} & \cdots & \cdots & \Sigma_{mm} \end{bmatrix},$$

where any entry $\Sigma_{ij} = \mathbb{E}\left[\tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_j^{\mathsf{T}}\right] \in \mathbb{R}^{d \times d}$. Now, recall that the structure of a retrieval can be expressed as

$$\tilde{\mathbf{v}}_i = (1 + \xi_i)\mathbf{v}_i + \eta_i + \sum_{k \neq i} \lambda^{(ik)},$$

where $\lambda^{(ik)}$ represents the cross-talk introduced from $\mathbf{s}_i^{\dagger} \otimes (\mathbf{s}_k \otimes \mathbf{v}_k)$. From the definition of the covariance, we have

$$\begin{aligned} \mathrm{Cov}\left(\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_j \mid \tilde{\mathbf{V}}_{1:m}, \mathbf{x}\right) &= \mathbb{E}\left[(\tilde{\mathbf{v}}_i - \mathbf{v}_i)((\tilde{\mathbf{v}}_j - \mathbf{v}_j)^{\mathsf{T}} \mid \tilde{\mathbf{V}}_{1:m}, \mathbf{x}\right] \\ &= \mathbb{E}\left[\epsilon_i \epsilon_j^{\mathsf{T}} \middle| \tilde{\mathbf{V}}_{1:m}, \mathbf{x}\right] \\ &= \mathbb{E}\left[\left(\xi_i \mathbf{v}_i + \eta_i + \sum_{\ell \neq i} \lambda^{(ik)}\right)\left(\xi_j \mathbf{v}_j + \eta_j + \sum_{r \neq j} \lambda^{(jr)}\right)^{\mathsf{T}} \middle| \tilde{\mathbf{V}}_{1:m}, \mathbf{x}\right] \end{aligned}$$

D.2.1 DIAGONAL BLOCKS

We begin by analyzing the diagonal blocks of the block covariance matrix, Σ. Specifically, we'd like to find the structure of an entry in Σ_{ii}, defined as

$$(\Sigma_{ii})_{ab} = \mathbb{E}\left[\left(\xi_i \mathbf{v}_i + \eta_i + \sum_{\ell \neq i} \lambda^{(i\ell)}\right)_a \cdot \left(\xi_i \mathbf{v}_i + \eta_i + \sum_{r \neq i} \lambda^{(ir)}\right)_b \middle| \tilde{\mathbf{V}}_{1:m}, \mathbf{x}\right].$$

We begin by computing the expected values of the terms in the resulting product.

1. $\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a \cdot (\xi_i \mathbf{v}_i)_b\right] = (\mathbf{v}_i)_a (\mathbf{v}_i)_b \, \mathbb{E}\left[\xi_i^2\right] = (\mathbf{v}_i)_a (\mathbf{v}_i)_b \cdot \frac{2}{d}$

2. $\mathbb{E}\left[(\eta_i)_a (\eta_i)_b\right] \neq 0$

3. $\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a (\eta_i)_b\right] = (\mathbf{v}_i)_a \mathbb{E}\left[\xi_i (\eta_i)_b\right] = 0$

4. $\mathbb{E}\left[(\eta_i)_a (\lambda^{(i\ell)})_b\right] = \mathbb{E}\left[(\eta_i)_a (\mathbf{s}_i^\dagger \otimes \mathbf{s}_\ell \otimes \mathbf{v}_\ell)_b\right]$

   Observe that $(\eta_i)_a$ will contain traces of elements of $\mathbf{s}_i$, so the unbinding operation will introduce terms with squared elements of $\mathbf{s}_i$. However, these terms will still have elements of $\mathbf{s}_\ell$ to the first power, which in expectation, from the HRR distribution, will result in

   $$\mathbb{E}\left[(\eta_i)_a (\lambda^{(i\ell)})_b\right] = 0.$$

5. $\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a (\lambda^{(i\ell)})_b\right]$

   Applying the same reasoning as in 4), entries from $\mathbf{s}_\ell$ will remain in the product to first power, resulting in

   $$\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a (\lambda^{(i\ell)})_b\right] = 0.$$

6. $\mathbb{E}\left[(\lambda^{(i\ell)})_a (\lambda^{(ir)})_b\right] \neq 0$

   For $\ell \neq r$, the expected value will evaluate to zero due to the elements of $\mathbf{s}_\ell$ and $\mathbf{s}_r$ appearing to first degree, however, the case where $\ell = r$ will not be zero since squared elements of the symbols will appear.

From the analysis above, we see that the covariance, Σ_{ii}, can be expressed as

$$\Sigma_{ii} = \mathbb{E}\left[(\xi_i \mathbf{v}_i + \eta_i)(\xi_i \mathbf{v}_i + \eta_i)^\mathsf{T}\right] + \sum_{\ell \neq i} \mathbb{E}\left[\lambda^{(i\ell)} (\lambda^{(i\ell)})^\mathsf{T}\right],$$

as the other cross-terms cancel. Then, applying the definitions of circular convolution and circular correlation, we can express an entry from a single cross-talk contribution. Letting $\mathbf{s}_k^{(i)}$ denote the k'th

entry of $\mathbf{s}_i$, we have

$$\begin{aligned}
\mathbb{E}\left[(\lambda^{(i\ell)})_a(\lambda^{(i\ell)})_b\right] &= \mathbb{E}\left[\left(\sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(\ell)}(\mathbf{v}_\ell)_{k+a-m}\right)\cdot\left(\sum_{k'=0}^{d-1}\sum_{m'=0}^{d-1}\mathbf{s}_{k'}^{(i)}\mathbf{s}_{m'}^{(\ell)}(\mathbf{v}_\ell)_{k'+b-m'}\right)\right] \\
&= \mathbb{E}\left[\sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\sum_{k'=0}^{d-1}\sum_{m'=0}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(\ell)}\mathbf{s}_{k'}^{(i)}\mathbf{s}_{m'}^{(\ell)}(\mathbf{v}_\ell)_{k+a-m}(\mathbf{v}_\ell)_{k'+b-m'}\right] \\
&= \sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\sum_{k'=0}^{d-1}\sum_{m'=0}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(\ell)}\mathbf{s}_{k'}^{(i)}\mathbf{s}_{m'}^{(\ell)}\right](\mathbf{v}_\ell)_{k+a-m}(\mathbf{v}_\ell)_{k'+b-m'} \\
&= \sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(\ell)}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(\ell)}\right](\mathbf{v}_\ell)_{k+a-m}(\mathbf{v}_\ell)_{k+b-m} \\
&= \frac{1}{d^2}\sum_{k=0}^{d-1}\sum_{m=0}^{d-1}(\mathbf{v}_\ell)_{k+a-m}(\mathbf{v}_\ell)_{k+b-m} \\
&= \frac{1}{d^2}\sum_{k=0}^{d-1}\sum_{n=0}^{d-1}(\mathbf{v}_\ell)_{a+n}(\mathbf{v}_\ell)_{b+n} \\
&= \frac{1}{d}\sum_{n=0}^{d-1}(\mathbf{v}_\ell)_{a+n}(\mathbf{v}_\ell)_{b+n},
\end{aligned}$$

where $n = k - m \mod d$. Thus, the contribution of the cross-talk term can be written as

$$\sum_{\ell\neq i}\mathbb{E}\left[(\lambda^{(i\ell)})_a(\lambda^{(i\ell)})_b\right] = \sum_{\ell\neq i}\frac{1}{d}\sum_{n=0}^{d-1}(\mathbf{v}_\ell)_{a+n}(\mathbf{v}_\ell)_{b+n}.$$

Evaluating the non-crosstalk contributions, we have that

$$\begin{aligned}
\mathbb{E}\left[(\xi_i\mathbf{v}_i+\eta_i)_a(\xi_i\mathbf{v}_i+\eta_i)_b\right] &= \mathbb{E}\left[(\xi_i\mathbf{v}_i)_a(\xi_i\mathbf{v}_i)_b + (\eta_i)_a(\eta_i)_b\right] \\
&= (\mathbf{v}_i)_a(\mathbf{v}_i)_b\,\mathbb{E}\left[\xi_i^2\right] + \mathbb{E}\left[(\eta_i)_a(\eta_i)_b\right] \\
&= (\mathbf{v}_i)_a(\mathbf{v}_i)_b\cdot\frac{2}{d} + \mathbb{E}\left[(\eta_i)_a(\eta_i)_b\right].
\end{aligned}$$

We can derive an expression for $(\eta_i)_a$ directly from the unbinding definition,

$$\begin{aligned}
(\tilde{\mathbf{v}}_i)_a &= \sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}(\mathbf{v}_i)_{k+a-m} \\
&= \sum_{k=0}^{d-1}\left(\mathbf{s}_k^{(i)}\mathbf{s}_k^{(i)}(\mathbf{v}_i)_{k+a-k} + \sum_{m\neq k}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}(\mathbf{v}_i)_{k+a-m}\right) \\
&= \sum_{k=0}^{d-1}\left(\left(\mathbf{s}_k^{(i)}\right)^2(\mathbf{v}_i)_a + \sum_{m\neq k}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}(\mathbf{v}_i)_{k+a-m}\right) \\
&= \left[\left(\mathbf{s}_1^{(i)}\right)^2 + \left(\mathbf{s}_2^{(i)}\right)^2 + \cdots + \left(\mathbf{s}_d^{(i)}\right)^2\right](\mathbf{v}_i)_a + \sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}(\mathbf{v}_i)_{k+a-m} \\
&= \|\mathbf{s}_i\|^2(\mathbf{v}_i)_a + \sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}(\mathbf{v}_i)_{k+a-m}.
\end{aligned}$$

So we have that

$$\begin{aligned}
\mathbb{E}\left[(\eta_i)_a(\eta_i)_b\right] &= \mathbb{E}\left[\left(\sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}(\mathbf{v}_i)_{k+a-m}\right)\cdot\left(\sum_{k'=0}^{d-1}\sum_{m'\neq k'}^{d-1}\mathbf{s}_{k'}^{(i)}\mathbf{s}_{m'}^{(i)}(\mathbf{v}_i)_{k'+b-m'}\right)\right] \\
&= \mathbb{E}\left[\sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\sum_{k'=0}^{d-1}\sum_{m'\neq k'}^{d-1}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}\mathbf{s}_{k'}^{(i)}\mathbf{s}_{m'}^{(i)}(\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{k'+b-m'}\right] \\
&= \sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\sum_{k'=0}^{d-1}\sum_{m'\neq k'}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}\mathbf{s}_{k'}^{(i)}\mathbf{s}_{m'}^{(i)}\right](\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{k'+b-m'}
\end{aligned}$$

Observe that the expectation is non-zero only when $k = k', m = m'$ and when $k = m', m = k'$, thus,

$$\begin{aligned}
\mathbb{E}\left[(\eta_i)_a(\eta_i)_b\right] &= \sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}\right](\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{k+b-m} \\
&\quad + \sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(i)}\mathbf{s}_m^{(i)}\mathbf{s}_k^{(i)}\right](\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{m+b-k} \\
&= \frac{1}{d^2}\sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}(\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{k+b-m} + \frac{1}{d^2}\sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}(\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{m+b-k} \\
&= \frac{1}{d^2}\left[\sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}(\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{k+b-m} + \sum_{k=0}^{d-1}\sum_{m\neq k}^{d-1}(\mathbf{v}_i)_{k+a-m}(\mathbf{v}_i)_{m+b-k}\right] \\
&= \frac{1}{d}\left[\sum_{n\neq 0}(\mathbf{v}_i)_{a+n}(\mathbf{v}_i)_{b+n} + \sum_{n\neq 0}(\mathbf{v}_i)_{a+n}(\mathbf{v}_i)_{b-n}\right].
\end{aligned}$$

Finally, we can state

$$\begin{aligned}
(\Sigma_{ii})_{ab} = \underbrace{(\mathbf{v}_i)_a(\mathbf{v}_i)_b\cdot\frac{2}{d}}_{\mathbb{E}[(\xi_i\mathbf{v}_i)_a(\xi_i\mathbf{v}_i)_b]} &+ \underbrace{\sum_{\ell\neq i}\frac{1}{d}\sum_{n=0}^{d-1}(\mathbf{v}_\ell)_{a+n}(\mathbf{v}_\ell)_{b+n}}_{\sum_{\ell\neq i}\mathbb{E}\left[(\lambda^{(i\ell)})_a(\lambda^{(i\ell)})_b\right]} \\
&+ \underbrace{\frac{1}{d}\left[\sum_{n\neq 0}(\mathbf{v}_i)_{a+n}(\mathbf{v}_i)_{b+n} + \sum_{n\neq 0}(\mathbf{v}_i)_{a+n}(\mathbf{v}_i)_{b-n}\right]}_{\mathbb{E}[(\eta_i)_a(\eta_i)_b]}.
\end{aligned}$$

Observe that when $a = b$, and when $d \gg 0$, we'll have, for the average case

$$\begin{aligned}
(\Sigma_{ii})_{aa} &= (\mathbf{v}_i)_a^2 \cdot \frac{2}{d} + \sum_{\ell \neq i} \frac{1}{d} \sum_{n=0}^{d-1} (\mathbf{v}_\ell)_{a+n}^2 + \frac{1}{d} \left[\sum_{n \neq 0} (\mathbf{v}_i)_{a+n}^2 + \sum_{n \neq 0} (\mathbf{v}_i)_{a+n} (\mathbf{v}_i)_{b-n} \right] \\
&= \underbrace{(\mathbf{v}_i)_a^2}_{1/d} \cdot \frac{2}{d} + \frac{1}{d} \sum_{\ell \neq i} \underbrace{\|\mathbf{v}_\ell\|^2}_{\approx 1} + \frac{1}{d} \left[\underbrace{\left(\|\mathbf{v}_i\|^2 - (\mathbf{v}_i)_a^2 \right)}_{\approx 1 - 1/d} + \underbrace{\sum_{n \neq 0} (\mathbf{v}_i)_{a+n} (\mathbf{v}_i)_{b-n}}_{\approx 0} \right] \\
&\approx \frac{2}{d^2} + \frac{m-1}{d} + \frac{1}{d} - \frac{1}{d^2} \\
&= \frac{1}{d^2} + \frac{m}{d} \\
&\approx \frac{m}{d}.
\end{aligned}$$

The approximations defined under the braces come from the fact that elements of $\mathbf{v}_i$ are sampled from the HRR distribution, $\mathcal{N}(0, 1/d)$.

In the alternative case, for $a \neq b$, we have $(\Sigma_{ii})_{ab} \approx 0$ due to the products of terms that have zero mean. Thus, we have the approximation

$$\Sigma_{ii} \approx \frac{m}{d} \mathbf{I}.$$

D.2.2 Off-diagonal blocks

For the off-diagonal blocks, we are interested in

$$(\Sigma_{ij})_{ab} = \mathbb{E} \left[\left(\xi_i \mathbf{v}_i + \eta_i + \sum_{\ell \neq i} \lambda^{(i\ell)} \right)_a \cdot \left(\xi_j \mathbf{v}_j + \eta_j + \sum_{r \neq j} \lambda^{(jr)} \right)_b \middle| \tilde{\mathbf{V}}_{1:m}, \mathbf{x} \right].$$

By going through cases again, we make the following observations.

1. $\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a \cdot (\xi_j \mathbf{v}_j)_b\right] = (\mathbf{v}_i)_a (\mathbf{v}_j)_b \, \mathbb{E}\left[\xi_i \xi_j\right] = 0$
2. $\mathbb{E}\left[(\eta_i)_a (\eta_j)_b\right] = 0$
3. $\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a (\eta_j)_b\right] = (\mathbf{v}_i)_a \mathbb{E}\left[\xi_i (\eta_j)_b\right] = 0$
4. $\mathbb{E}\left[(\eta_i)_a (\lambda^{(jr)})_b\right] = 0$

   We have already seen that for the case when $i = j$, the expectation is zero. Additionally, when $i \neq j$, the expectation is still zero, as entries of $\mathbf{s}_i$, $\mathbf{s}_j$, and $\mathbf{s}_r$, all appear at first power.
5. $\mathbb{E}\left[(\xi_i \mathbf{v}_i)_a (\lambda^{(jr)})_b\right] = 0$

   Applying the same logic as 4), we see that this expectation is also zero.
6. $\mathbb{E}\left[(\lambda^{(i\ell)})_a (\lambda^{(jr)})_b\right]$

   For this case, the expectation will be non-zero only for the case when $\ell = j$ and $r = i$.

Examining case six closer, similarly as before, we can write

$$\begin{aligned}(\Sigma_{ij})_{ab} &= \sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\sum_{k'=0}^{d-1}\sum_{m'=0}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(j)}\mathbf{s}_{k'}^{(j)}\mathbf{s}_{m'}^{(i)}\right](\mathbf{v}_j)_{k+a-m}(\mathbf{v}_i)_{k'+b-m'} \\ &= \sum_{k=0}^{d-1}\sum_{m=0}^{d-1}\mathbb{E}\left[\mathbf{s}_k^{(i)}\mathbf{s}_m^{(j)}\mathbf{s}_m^{(j)}\mathbf{s}_k^{(i)}\right](\mathbf{v}_j)_{k+a-m}(\mathbf{v}_i)_{m+b-k} \\ &= \frac{1}{d^2}\sum_{k=0}^{d-1}\sum_{m=0}^{d-1}(\mathbf{v}_j)_{k+a-m}(\mathbf{v}_i)_{m+b-k} \\ &= \frac{1}{d}\sum_{n=0}^{d-1}(\mathbf{v}_j)_{a+n}(\mathbf{v}_i)_{b-n}.\end{aligned}$$

This establishes that entries of Σ_{ij} are at least $\mathcal{O}(\frac{1}{d})$. Then, observe that from the definition of circular convolution, we can rewrite the final equality as

$$\begin{aligned}(\Sigma_{ij})_{ab} &= \frac{1}{d}\sum_{n=0}^{d-1}(\mathbf{v}_j)_{a+n}(\mathbf{v}_i)_{b-n} \\ &= \frac{1}{d}(\mathbf{v}_j \otimes \mathbf{v}_i)_{a+b}.\end{aligned}$$

We can then express the squared Frobenius norm of Σ_{ij} as

$$\begin{aligned}\|\Sigma_{ij}\|_F^2 &= \sum_{a=1}^{d}\sum_{b=1}^{d}(\Sigma_{ij})_{ab}^2 \\ &= \frac{1}{d^2}\sum_{a=0}^{d-1}\sum_{b=0}^{d-1}|(\mathbf{v}_j \otimes \mathbf{v}_i)_{a+b}|^2 \\ &= \frac{1}{d^2}\cdot d\sum_{c=0}^{d-1}|(\mathbf{v}_j \otimes \mathbf{v}_i)_c|^2 \\ &= \frac{1}{d}\sum_{c=0}^{d-1}|(\mathbf{v}_j \otimes \mathbf{v}_i)_c|^2 \\ &= \frac{1}{d}\|(\mathbf{v}_j \otimes \mathbf{v}_i)\|^2.\end{aligned}$$

Since each entry of $(\mathbf{v}_j \otimes \mathbf{v}_i)$ has variance $\frac{1}{d}$, we can say that $(\mathbf{v}_j \otimes \mathbf{v}_i)_c = \mathcal{O}(\frac{1}{\sqrt{d}})$. From this, we can establish that

$$\begin{aligned}\|(\mathbf{v}_j \otimes \mathbf{v}_i)\|^2 &= \sum_{c=0}^{d-1}|(\mathbf{v}_j \otimes \mathbf{v}_i)_c|^2 \\ &= \sum_{c=0}^{d-1}\mathcal{O}\left(\frac{1}{d}\right) \\ &= \mathcal{O}(1).\end{aligned}$$

Thus, we have that

$$\|\Sigma_{ij}\|_F^2 = \frac{1}{d}\underbrace{\|(\mathbf{v}_j \otimes \mathbf{v}_i)\|^2}_{\mathcal{O}(1)},$$

which implies that $\|\Sigma_{ij}\|_F^2$ is $\mathcal{O}\left(\frac{1}{d}\right)$.

D.2.3 SHARPENED OFF-DIAGONAL OPERATOR NORM

In general, for a matrix $\mathbf{A}$, we define the operator norm as $\|\mathbf{A}\|_{\mathrm{op}} = \max_{\|\mathbf{x}\|\neq 0} \frac{\|\mathbf{A}\mathbf{x}\|}{\|\mathbf{x}\|}$. We also establish a sharper bound on $\|\Sigma_{ij}\|_{\mathrm{op}}$. While the Frobenius bound gives $\|\Sigma_{ij}\|_{\mathrm{op}} \leq \|\Sigma_{ij}\|_F = \mathcal{O}(1/\sqrt{d})$, the circulant structure of Σ_{ij} yields a tighter estimate. Recall $(\Sigma_{ij})_{ab} = \frac{1}{d}(\mathbf{v}_j \otimes \mathbf{v}_i)_{a+b}$, i.e. $\Sigma_{ij} = \frac{1}{d}\,\mathrm{H}(\mathbf{v}_j \otimes \mathbf{v}_i)$ where $\mathrm{H}(\mathbf{g})_{ab} = g_{(a+b) \bmod d}$. The matrix $\mathrm{H}(\mathbf{g})$ is a circulant composed with the index-reversal permutation, $\mathrm{H}(\mathbf{g}) = P\,\mathrm{Circ}(\mathbf{g})$ with P orthogonal, so its singular values coincide with those of $\mathrm{Circ}(\mathbf{g})$, namely the moduli $\{|\hat{g}_k|\}$ of the DFT of $\mathbf{g}$. Let

$$\mathbf{g} = \mathbf{v}_j \otimes \mathbf{v}_i, \qquad g_\ell = (\mathbf{v}_j \otimes \mathbf{v}_i)_\ell,$$

with all subscripts interpreted modulo d. Then

$$(\Sigma_{ij})_{ab} = \frac{1}{d} g_{a+b},$$

so

$$\Sigma_{ij} = \frac{1}{d}\begin{bmatrix} g_0 & g_1 & g_2 & \cdots & g_{d-1} \\ g_1 & g_2 & g_3 & \cdots & g_0 \\ g_2 & g_3 & g_4 & \cdots & g_1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{d-1} & g_0 & g_1 & \cdots & g_{d-2} \end{bmatrix}.$$

Equivalently,

$$\Sigma_{ij} = \frac{1}{d}\,\mathrm{H}(\mathbf{g}), \qquad \mathrm{H}(\mathbf{g})_{ab} = g_{(a+b) \bmod d}.$$

Using the convention

$$\mathrm{Circ}(\mathbf{g})_{ab} = g_{(b-a) \bmod d},$$

we have

$$\mathrm{Circ}(\mathbf{g}) = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & g_{d-1} \\ g_{d-1} & g_0 & g_1 & \cdots & g_{d-2} \\ g_{d-2} & g_{d-1} & g_0 & \cdots & g_{d-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & g_3 & \cdots & g_0 \end{bmatrix}.$$

Now define the index-reversal permutation matrix P by

$$P_{ab} = \mathbf{1}\{b \equiv -a \pmod d\}.$$

Explicitly,

$$P = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & \ldots & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \end{bmatrix}.$$

Therefore,

$$\mathrm{H}(\mathbf{g}) = P\,\mathrm{Circ}(\mathbf{g}),$$

and hence

$$\Sigma_{ij} = \frac{1}{d} P\,\mathrm{Circ}(\mathbf{v}_j \otimes \mathbf{v}_i).$$

Using that binding multiplies DFT coefficients pointwise,

$$\|\Sigma_{ij}\|_{\mathrm{op}} = \tfrac{1}{d}\max_k \big|\widehat{(\mathbf{v}_j \otimes \mathbf{v}_i)}_k\big| = \tfrac{1}{d}\max_k |\hat{\mathbf{v}}_{j,k}|\,|\hat{\mathbf{v}}_{i,k}|.$$

For $\mathbf{v} \sim$ the HRR distribution, each transform coefficient is complex Gaussian with $\mathbb{E}|\hat{\mathbf{v}}_k|^2 = 1$, so with high probability over the codebook $\max_k |\hat{\mathbf{v}}_k|^2 = \mathcal{O}(\log d)$, giving

$$\|\Sigma_{ij}\|_{\mathrm{op}} = \mathcal{O}\Big(\tfrac{\log d}{d}\Big).$$

D.3 A POSTERIOR PRECISION FLOOR

The log-determinant reduction developed below requires the joint covariance Σ to be nonsingular. We first show that, for the idealized noiseless retrieval model, Σ is in fact singular. The key point is that the retrieved slots follow an exact deterministic linear constraint. Recall that the latent structure that we use takes the form

$$\mathbf{M} = \sum_{k=1}^{m} \mathbf{s}_k \otimes \mathbf{v}_k,$$

and the retrieval of slot i is

$$\tilde{\mathbf{v}}_i = \mathbf{s}_i^{\dagger} \otimes \mathbf{M}.$$

We analyze this relation in the Fourier domain, where circular convolution and correlation diagonalize. For a real vector $\mathbf{u} \in \mathbb{R}^d$, write its discrete Fourier transform as

$$\hat{u}_\omega = \sum_{n=0}^{d-1} u_n e^{-2\pi i \omega n/d}, \qquad \omega = 0, \dots, d-1.$$

The circular convolution theorem gives

$$\widehat{(\mathbf{a} \otimes \mathbf{b})}_\omega = \hat{a}_\omega \hat{b}_\omega.$$

Similarly, for the unbinding operation

$$(\mathbf{c}^{\dagger} \otimes \mathbf{y})_j = \sum_{k=0}^{d-1} c_k y_{k+j},$$

a direct computation gives

$$\widehat{(\mathbf{c}^{\dagger} \otimes \mathbf{y})}_\omega = \overline{\hat{c}_\omega}\, \hat{y}_\omega.$$

Therefore,

$$\widehat{\mathbf{M}}_\omega = \sum_{k=1}^{m} \hat{s}_{k,\omega} \hat{v}_{k,\omega}.$$

Define the shared Fourier coefficient

$$Z_\omega := \widehat{\mathbf{M}}_\omega = \sum_{k=1}^{m} \hat{s}_{k,\omega} \hat{v}_{k,\omega}.$$

Then the retrieved slot satisfies

$$\widehat{\tilde{\mathbf{v}}}_{i,\omega} = \overline{\hat{s}_{i,\omega}} Z_\omega.$$

Thus every slot's retrieval at frequency ω is obtained from the same scalar Z_ω, modulated by that slot's own symbol coefficient.

Now weighing each retrieved coefficient by the conjugate of the corresponding target coefficient and summing over slots, we get

$$\sum_{i=1}^{m} \overline{\hat{v}_{i,\omega}}\, \widehat{\tilde{\mathbf{v}}}_{i,\omega} = \sum_{i=1}^{m} \overline{\hat{v}_{i,\omega}}\, \overline{\hat{s}_{i,\omega}}\, Z_\omega.$$

Factoring out Z_ω, we obtain

$$\sum_{i=1}^{m} \overline{\hat{v}_{i,\omega}}\, \widehat{\tilde{\mathbf{v}}}_{i,\omega} = Z_\omega \sum_{i=1}^{m} \overline{\hat{s}_{i,\omega} \hat{v}_{i,\omega}} = Z_\omega \overline{\sum_{i=1}^{m} \hat{s}_{i,\omega} \hat{v}_{i,\omega}}.$$

By the definition of Z_ω, this becomes

$$\sum_{i=1}^{m} \overline{\hat{v}_{i,\omega}}\, \widehat{\tilde{\mathbf{v}}}_{i,\omega} = Z_\omega \overline{Z_\omega} = |Z_\omega|^2.$$

The right-hand side is real-valued for every realization of the symbols. Hence

$$\operatorname{Im}\left(\sum_{i=1}^{m} \overline{\hat{v}_{i,\omega}}\, \widehat{\tilde{\mathbf{v}}}_{i,\omega} \right) = 0$$

identically.

Conditioned on the codebook $\{\mathbf{v}_i\}_{i=1}^m$, the coefficients $\overline{\hat{v}_{i,\omega}}$ are fixed. Also, the Fourier transform is a fixed invertible linear map on the real coordinates of each retrieved vector. Therefore

$$L_\omega(\tilde{\mathbf{V}}_{1:m}) := \mathrm{Im}\left(\sum_{i=1}^m \overline{\hat{v}_{i,\omega}}\,\widehat{\tilde{\mathbf{v}}}_{i,\omega}\right)$$

is a fixed real-linear functional of the joint retrieval vector

$$\tilde{\mathbf{V}}_{1:m} = (\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_m) \in \mathbb{R}^{md}.$$

The identity above says that

$$L_\omega(\tilde{\mathbf{V}}_{1:m}) = 0 \qquad \text{almost surely.}$$

Equivalently, there exists a real vector $a_\omega \in \mathbb{R}^{md}$, depending only on the fixed codebook and on the Fourier frequency, such that

$$a_\omega^\top \tilde{\mathbf{V}}_{1:m} = 0 \qquad \text{almost surely.}$$

Thus

$$\mathrm{Var}\left(a_\omega^\top \tilde{\mathbf{V}}_{1:m}\right) = a_\omega^\top \Sigma a_\omega = 0.$$

Because Σ is a covariance matrix, it is positive semidefinite. Hence $a_\omega^\top \Sigma a_\omega = 0$ implies

$$\Sigma a_\omega = 0.$$

Therefore a_ω is a nontrivial null direction of Σ, so Σ is not full rank and

$$\det \Sigma = 0.$$

Under the joint Gaussian model, this makes $\log\det(\bar{\Sigma}^\dagger \Sigma)$ diverge to $-\infty$, so the log-determinant expression for the total correlation becomes infinite. This singularity is therefore a property of the idealized noiseless retrieval model, not of the log-determinant reduction itself.

The resolution is that the variational model does not produce deterministic slots. Each posterior factor $q_i(\tilde{\mathbf{v}}_i \mid \mathbf{x})$ is a non-degenerate density, since the encoder assigns the slot a finite posterior precision. We model this posterior uncertainty as independent additive Gaussian noise across slots and coordinates, with per-coordinate variance $\tau^2 > 0$. Under this regularization, the joint covariance becomes

$$\Sigma_\tau = \Sigma + \tau^2 \mathbf{I}_{md},$$

and the block-diagonal marginal covariance becomes

$$\bar{\Sigma}_\tau = \bar{\Sigma} + \tau^2 \mathbf{I}_{md}.$$

The off-diagonal blocks Σ_{ij}, $i \neq j$, are unchanged, while every degenerate direction receives variance τ^2. Since $\Sigma \succeq 0$, we have

$$\Sigma_\tau = \Sigma + \tau^2 \mathbf{I}_{md} \succ 0.$$

Thus the regularized covariance is nonsingular, and the log-determinant expression is well-defined.

Henceforth, in the total-correlation bound, we replace $\bar{\Sigma}$ by $\bar{\Sigma}_\tau = \bar{\Sigma} + \tau^2 \mathbf{I}_{md}$. A useful consequence is that, because $\bar{\Sigma} \succeq 0$,

$$\lambda_{\min}(\bar{\Sigma} + \tau^2 \mathbf{I}_{md}) \geq \tau^2.$$

Therefore

$$\|(\bar{\Sigma} + \tau^2 \mathbf{I}_{md})^{-1/2}\|_{\mathrm{op}} = \frac{1}{\sqrt{\lambda_{\min}(\bar{\Sigma} + \tau^2 \mathbf{I}_{md})}} \leq \tau^{-1}. \tag{8}$$

This gives a posterior precision floor for the whitening operation. Its operator norm is controlled by τ alone, and the conditioning of the noiseless diagonal blocks Σ_{ii} never enters.

D.4 Reduction to the log-determinant

Remark 1. *A natural approach to bounding* (7) *would be to work with each* $\mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big)$ *individually. For jointly Gaussian variables, each such term equals* $-\frac{1}{2}\log\det(\mathbf{I} - R_i)$*, where* $R_i = \Sigma_{ii}^{-1/2}\Sigma_{i,<i}\Sigma_{<i,<i}^{-1}\Sigma_{<i,i}\Sigma_{ii}^{-1/2}$ *is the canonical correlation matrix. Bounding* R_i *via the submultiplicative chain*

$$\|R_i\|_{\mathrm{op}} \le \|\Sigma_{ii}^{-1/2}\|_{\mathrm{op}}^2\, \|\Sigma_{i,<i}\|_{\mathrm{op}}^2\, \|\Sigma_{<i,<i}^{\dagger}\|_{\mathrm{op}}$$

would require $\|\Sigma_{<i,<i}^{\dagger}\|_{\mathrm{op}} = 1/\lambda_{\min}(\Sigma_{<i,<i})$*, which is the* smallest *eigenvalue of* $\Sigma_{<i,<i}$ *and is not controlled by the operator norms we can establish. Moreover, even granting such a bound, the submultiplicative product discards the alignment between the ill-conditioned directions of* $\Sigma_{<i,<i}^{\dagger}$ *and the directions in which* $\Sigma_{i,<i}$ *carries mass, so the resulting estimate would not be tight. We therefore do not bound the* R_i *individually.*

Since we are ultimately interested in the aggregate quantity established in (7),

$$\sum_{i=2}^{m} \mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big) \;=\; D_{\mathrm{KL}}\big(q \,\|\, \textstyle\prod_i q_i\big),$$

we bound this total correlation directly. Under the joint Gaussian model with mean $(\mathbf{v}_1(\mathbf{x}), \ldots, \mathbf{v}_m(\mathbf{x}))$ and covariance Σ, this KL admits the closed form

$$D_{\mathrm{KL}}\big(q \,\|\, \textstyle\prod_i q_i\big) \;=\; \tfrac{1}{2}\big[\mathrm{tr}\big(\bar{\Sigma}^{\dagger}\Sigma\big) - md - \log\det\big(\bar{\Sigma}^{\dagger}\Sigma\big)\big]\,,$$

where $\bar{\Sigma} = \mathrm{blkdiag}(\Sigma_{11}, \ldots, \Sigma_{mm})$ is the covariance of the product of marginals $\prod_i q_i$.

Write $\Delta = \Sigma - \bar{\Sigma}$ for the part of Σ consisting of its off-diagonal blocks Σ_{ij} $(i \neq j)$; by construction Δ has vanishing diagonal blocks. Then $\bar{\Sigma}^{\dagger}\Sigma = \mathbf{I} + \bar{\Sigma}^{\dagger}\Delta$, and because $\bar{\Sigma}^{\dagger}$ is block-diagonal while Δ has no diagonal blocks,

$$\mathrm{tr}\big(\bar{\Sigma}^{\dagger}\Sigma\big) = md + \mathrm{tr}\big(\bar{\Sigma}^{\dagger}\Delta\big) = md + \sum_i \mathrm{tr}\big(\Sigma_{ii}^{\dagger}\,\Delta_{ii}\big) = md,$$

so the trace term cancels exactly. Introducing the symmetric matrix

$$M \;=\; \bar{\Sigma}^{-1/2}\,\Delta\,\bar{\Sigma}^{-1/2},$$

which satisfies $\det(\mathbf{I} + \bar{\Sigma}^{\dagger}\Delta) = \det(\mathbf{I} + M)$ by similarity and $\mathrm{tr}(M) = \mathrm{tr}(\bar{\Sigma}^{\dagger}\Delta) = 0$, we are left with

$$\sum_{i=2}^{m} \mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big) \;=\; -\tfrac{1}{2}\log\det(\mathbf{I} + M) \;=\; -\tfrac{1}{2}\sum_k \log(1 + \mu_k), \tag{9}$$

where $\{\mu_k\}$ are the eigenvalues of M. Since $\mathbf{I} + M = \bar{\Sigma}^{-1/2}\Sigma\,\bar{\Sigma}^{-1/2} \succeq 0$, each eigenvalue satisfies $\mu_k \ge -1$. Note that this reduction requires only the marginal whitening $\bar{\Sigma}^{-1/2}$, and does not invert the conditioning block $\Sigma_{<i,<i}$.

D.5 Bounding the total correlation

We now bound (9) with $\bar{\Sigma}$ regularized as above. The eigenvalues of M obey $\sum_k \mu_k = \mathrm{tr}(M) = 0$ and $\mu_k > -1$. For each k, the second-order Taylor expansion of $\log(1 + \cdot)$ with Lagrange remainder gives

$$\log(1 + \mu_k) = \mu_k - \frac{\mu_k^2}{2(1 + \zeta_k)^2}, \qquad \zeta_k \text{ between } 0 \text{ and } \mu_k.$$

In either sign of μ_k one has $1 + \zeta_k > 1 + \mu_{\min} \ge 1 - \|M\|_{\mathrm{op}}$, so provided $\|M\|_{\mathrm{op}} < 1$,

$$\log(1 + \mu_k) > \mu_k - \frac{\mu_k^2}{2\,(1 - \|M\|_{\mathrm{op}})^2}.$$

Summing over k and using $\sum_k \mu_k = 0$ and $\sum_k \mu_k^2 = \|M\|_F^2$,

$$\sum_{i=2}^{m} \mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big) = -\tfrac{1}{2} \sum_k \log(1+\mu_k) \;\le\; \frac{\|M\|_F^2}{4\,(1-\|M\|_{\mathrm{op}})^2}. \tag{10}$$

We now need to bound the two quantities on the right, $\|M\|_F^2$, and $\|M\|_{\mathrm{op}}$. Writing $W = (\bar{\Sigma} + \tau^2\mathbf{I})^{-1/2}$ so that $M = W\Delta W$, the submultiplicativity of the Frobenius norm under operator-norm multiplication together with (8) gives

$$\|M\|_F = \|W\Delta W\|_F \le \|W\|_{\mathrm{op}}^2\, \|\Delta\|_F \le \tau^{-2}\, \|\Delta\|_F.$$

Since Δ collects the off-diagonal blocks, $\|\Delta\|_F^2 = \sum_{i\neq j}\|\Sigma_{ij}\|_F^2$, and substituting the established $\|\Sigma_{ij}\|_F^2 = \frac{1}{d}\|\mathbf{v}_j \otimes \mathbf{v}_i\|^2$ with $\|\mathbf{v}_j \otimes \mathbf{v}_i\|^2 = \mathcal{O}(1)$,

$$\|M\|_F^2 \;\le\; \frac{1}{\tau^4}\sum_{i\neq j}\|\Sigma_{ij}\|_F^2 \;=\; \frac{1}{\tau^4 d}\sum_{i\neq j}\|\mathbf{v}_j \otimes \mathbf{v}_i\|^2 \;=\; \mathcal{O}\!\left(\frac{m^2}{\tau^4 d}\right). \tag{11}$$

For the operator norm, the same reasoning gives $\|M\|_{\mathrm{op}} \le \tau^{-2}\|\Delta\|_{\mathrm{op}}$. Applying the triangle inequality and Cauchy-Schwarz to Δ, which collects all off-diagonal blocks Σ_{ij}, $i \neq j$,

$$\|\Delta\|_{\mathrm{op}} \le \sqrt{\sum_{i\neq j}\|\Sigma_{ij}\|_{\mathrm{op}}^2} = \sqrt{m(m-1)\,\mathcal{O}\Big(\tfrac{\log^2 d}{d^2}\Big)} = \mathcal{O}\Big(\tfrac{m\log d}{d}\Big),$$

where we use the sharpened estimate $\|\Sigma_{ij}\|_{\mathrm{op}} = \mathcal{O}(\log d/d)$ from the preceding section. Therefore $\|M\|_{\mathrm{op}} = \mathcal{O}\big(m\log d/(\tau^2 d)\big)$. The regularity condition $\|M\|_{\mathrm{op}} < 1$ of (10) holds once $d \gtrsim m\log d/\tau^2$, in which regime $(1-\|M\|_{\mathrm{op}})^{-2} = 1 + o(1)$.

Combining (10) and (11), we obtain the final estimate. Conditioned on the codebook, with per-slot posterior precision τ^2 and $d \gtrsim m\log d/\tau^2$,

$$\boxed{\sum_{i=2}^{m} \mathbb{I}\Big(\tilde{\mathbf{v}}_i;\, \tilde{\mathbf{V}}_{<i} \mid \mathbf{x}\Big) \;=\; \mathcal{O}\!\left(\frac{m^2}{\tau^4\, d}\right).}$$

The total correlation among the retrieved slots thus vanishes as d^{-1}, grows quadratically in the number of slots, and vanishes with the posterior precision through τ^{-4}.

D.6 Proof of Proposition 1 (HRR latent capacity)

We first bound $\mathbb{I}(\mathbf{x};\, \hat{\mathbf{z}})$ by a sum over slots, then bound each term by both an additive white gaussian noise (AWGN) capacity and a codebook entropy, and take the minimum.

Step 1: reduction to a sum The quantized symbolic latent $\hat{\mathbf{z}} = \sum_i \mathbf{s}_i \otimes \hat{\mathbf{v}}_i$ is a deterministic function of $(\hat{\mathbf{v}}_1, \ldots, \hat{\mathbf{v}}_m)$ given the (frozen) symbol vectors. Hence $\mathbb{I}(\mathbf{x};\, \hat{\mathbf{z}}) \le \mathbb{I}\Big(\mathbf{x};\, \hat{\mathbf{V}}_{1:m}\Big)$ by the data-processing inequality. By the chain rule and the identity $\mathbb{I}\Big(\mathbf{x};\, \hat{\mathbf{v}}_i \mid \hat{\mathbf{V}}_{<i}\Big) - \mathbb{I}(\mathbf{x};\, \hat{\mathbf{v}}_i) = \mathbb{I}\Big(\hat{\mathbf{v}}_i;\, \hat{\mathbf{V}}_{<i} \mid \mathbf{x}\Big) - \mathbb{I}\Big(\hat{\mathbf{v}}_i;\, \hat{\mathbf{V}}_{<i}\Big)$,

$$\mathbb{I}\Big(\mathbf{x};\, \hat{\mathbf{V}}_{1:m}\Big) \;\le\; \sum_{i=1}^{m}\mathbb{I}(\mathbf{x};\, \hat{\mathbf{v}}_i) \;+\; \sum_{i=2}^{m}\mathbb{I}\Big(\hat{\mathbf{v}}_i;\, \hat{\mathbf{V}}_{<i} \mid \mathbf{x}\Big) = \sum_{i=1}^{m}\mathbb{I}(\mathbf{x};\, \hat{\mathbf{v}}_i) \;+\; \mathcal{O}\left(\frac{m^2}{t^4 d}\right),$$

where we dropped the term $\mathbb{I}\Big(\hat{\mathbf{v}}_i;\, \hat{\mathbf{V}}_{<i}\Big)$. Since quantization is applied slotwise, $(\hat{\mathbf{v}}_1, \ldots, \hat{\mathbf{v}}_m)$ is a post-processing of $(\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_m)$. Therefore the conditional total correlation among the quantized slots is no larger than that of the continuous retrievals, so Proposition 2 applies to the second sum.

Step 2: per-slot AWGN bound Conditional on the encoder output, slot i is observed through the unbinding channel $\tilde{\mathbf{v}}_i = \mathbf{v}_i(\mathbf{x}) + \boldsymbol{\epsilon}_i$ where $\boldsymbol{\epsilon}_i$ has covariance $(m/d)\mathbf{I}_d$ (Appendix C). The signal $\mathbf{v}_i(\mathbf{x})$ has expected squared norm at most 1 (by the HRR distribution and the value regularizer $\mathcal{L}_{\text{value}}$ which targets $\tau_n = 1$). The capacity of a d-dimensional AWGN channel with average input power P and per-dimension noise variance σ^2 is $\frac{d}{2}\log(1 + P/(d\sigma^2))$. Substituting $P = 1$ and $\sigma^2 = m/d$ gives per-slot capacity $\frac{d}{2}\log(1 + \frac{1}{m})$. Quantization can only reduce mutual information (data-processing), so $\mathbb{I}(\mathbf{x};\, \hat{\mathbf{v}}_i) \le \mathbb{I}(\mathbf{x};\, \tilde{\mathbf{v}}_i) \le \frac{d}{2}\log(1 + \frac{1}{m})$.

Step 3: per-slot codebook bound Since $\hat{\mathbf{v}}_i$ takes values in a finite codebook of size k, $\mathbb{I}(\mathbf{x};\, \hat{\mathbf{v}}_i) \le \mathrm{H}(\hat{\mathbf{v}}_i) \le \log k$.

Step 4: combining Each term satisfies $\mathbb{I}(\mathbf{x};\, \hat{\mathbf{v}}_i) \le \min\left(\frac{d}{2}\log(1 + 1/m),\, \log k\right)$, so

$$\mathbb{I}(\mathbf{x};\, \hat{\mathbf{z}}) \;\le\; m \cdot \min\left(\tfrac{d}{2}\log(1 + 1/m),\, \log k\right) \;=\; \min\left(m \cdot \tfrac{d}{2}\log(1 + 1/m),\, m \log k\right),$$

which is the claim. □

Remark on tightness The AWGN bound is tight when the encoder produces values $\mathbf{v}_i(\mathbf{x})$ that are Gaussian-distributed with covariance proportional to identity (the capacity-achieving input). In practice, the combination of the variance regularizer $\mathcal{L}_{\text{value}}$ (which targets variance $1/d$ per entry, matching the HRR distribution) and quantization toward a learned codebook drives the encoder toward this regime. The codebook bound $m \log k$ is tight when codebook entries are used uniformly.

Design corollary For a target representation rate R in nats, Proposition 1 requires

$$m \cdot \frac{d}{2}\log(1 + 1/m) \ge R \qquad \text{and} \qquad m \log k \ge R.$$

Equivalently,

$$d \ge \frac{2R}{m \log(1 + 1/m)} \qquad \text{and} \qquad k \ge e^{R/m}.$$

The first condition captures the continuous HRR retrieval bottleneck, while the second captures the discrete codebook bottleneck. For large m, since $\log(1 + 1/m) \approx 1/m$, the dimension condition becomes approximately

$$d \gtrsim 2R.$$

Thus, increasing m lowers the effective per-slot SNR, but also distributes the target rate across more slots. These effects approximately balance in the large-m regime, causing the continuous HRR capacity to saturate near $d/2$. In contrast, the codebook requirement $k \ge e^{R/m}$ becomes easier to satisfy as m increases, since each slot needs to represent a smaller fraction of the total rate. Together, these conditions suggest to choose m to be large enough to cover the expected number of generative factors, d to be large enough to support the desired total continuous rate, and k large enough to avoid a discrete quantization bottleneck.

E Experimental setup

Choice of baselines Our baselines are chosen to position HRR against two reference points: established VAE-based disentanglement methods (Higgins et al., 2017; Chen et al., 2018) and methods that share our use of vector quantization (van den Oord et al., 2017; Hsu et al., 2023).

Hyperparameters for models As described in Section 4.1, to ensure fair comparisons we ran a sweep for each baseline and dataset over a single hyperparameter, over two seeds. We then computed the average InfoM score across both seeds for each hyperparameter, and took the hyperparameter that had the highest average InfoM scores. The values that were swept over can be found in Table 7, and are consistent with prior works (Hsu et al., 2023). The results of the sweep, the respective hyperparameters used for the main results in Table 2, can be found in Table 8.

An interesting observation early in our investigation is that performance was better when the weight of the regularization on the latent produced by the encoder, λ_{lat}, was low. We found similar performance

keeping in a range between 0 and $1e-5$. Although we keep it at zero, and thus has no influence on the optimization of our model, we still include λ_{lat} as a design parameter for future works. Although higher weight on λ_{lat} produced latent representations closer to a true expected HRR vector, it tended to trade off disentanglement performance. We hypothesize that giving the encoder more freedom in producing its representations outweighs the benefit of the structure obtained by imposing it. However, even without regularizing the latent, Fig. 3 shows us that the encoder still learns to produce latent representations that are zero mean and that have relatively constant per-dimension variance.

Computational details for experiments All models were trained using PyTorch (Paszke et al., 2019). Each training job used a single GPU, and all models were trained using the same codebase, datasets, training protocol, evaluation pipeline, and random-seed structure.

A typical HRR training run, as well as the baseline training runs under the same experimental protocol, required slightly over one hour on a single GPU. The main results reported in Table 2 required approximately 7 aggregate GPU-days. The codebook-size and latent-dimension comparisons in Appendix G required approximately another 7 aggregate GPU-days. The hyperparameter sweeps used to select baseline configurations required approximately 4 aggregate GPU-days. In total, these three groups of runs required approximately 18 aggregate GPU-days. The full research project required additional compute due to preliminary experiments, failed runs, debugging, and exploratory model variants that are not included in the reported results.

Computation of disentanglement metrics with HRRs Since our final representation, $\hat{\mathbf{z}}$, is composed of discrete components from the codebook, we use the codebook indices produced by the vector quantization process to compute the disentanglement metrics, as was done for VQ-VAE and QLAE paper (Hsu et al., 2023).

Common experiment variables All models share the same CNN architecture, inspired from the model used in the QLAE paper (Hsu et al., 2023). A detailed description of the autoencoder architecture is in Table 4. Additionally, all models share the same optimization configuration, detailed in Table 5. The HRR model uses one set of hyperparameters for all datasets, except the number of symbols m, which are listed in Table 6.

Confidence intervals and bold font criteria The results in Table 2 summarize performance over five independent seeds, specifically the sample mean of the scores for each respective metric. For each model, dataset, and metric, we report the sample mean together with a 95% confidence interval for the mean, in Tables 9 to 12. The interval is centered at the sample mean and uses a Student-t critical value with $n-1$ degrees of freedom:

$$\bar{x} \pm t_{0.975,n-1} \frac{s}{\sqrt{n}},$$

where $\bar{x}$ is the sample mean, s is the sample standard deviation computed with Bessel's correction, and $n = 5$ is the number of seeds. For bolding, we use a CI-overlap criterion. In each column, we first identify the entry with the highest upper confidence bound. We then use that entry's lower confidence bound as a threshold: any entry whose lower confidence bound meets or exceeds this threshold is also bolded.

Table 4: Architecture and shared training hyperparameters for the CNN autoencoder. d_z denotes the latent dimension and C the number of image channels. All convolutional layers omit bias terms.

Stage	**Layer**	**In**	**Out**	k	s	**Activation**
Encoder						
Conv 1	Conv2d†	C	64	4	2	LReLU(0.2)
Conv 2	Conv2d†	64	128	4	2	LReLU(0.2)
Conv 3	Conv2d†	128	256	4	2	LReLU(0.2)
Conv 4	Conv2d†	256	512	4	2	LReLU(0.2)
Flatten	—	—	8,192	—	—	—
FC 1	Linear	8,192	512	—	—	ReLU
FC 2	Linear	512	d_z	—	—	—
Decoder						
Canvas	Learnable param. $\mathbf{h}_0$	—	$512\times4\times4$	—	—	—
Style	Linear	d_z	512	—	—	ReLU
Up 1	StyleConvT‡	512	512	4	2	LReLU(0.2)
Up 2	StyleConvT‡	512	256	4	2	LReLU(0.2)
Up 3	StyleConvT‡	256	128	4	2	LReLU(0.2)
Up 4	StyleConvT‡	128	64	4	2	LReLU(0.2)
Out	Conv2d	64	C	3	1	—

† Followed by InstanceNorm2d with learnable affine parameters (γ, β); padding $p = 1$.
‡ StyleConvT: ConvTranspose2d ($p = 1$) $\rightarrow$ LReLU(0.2) $\rightarrow$ InstanceNorm2d (no affine) $\rightarrow$ AdaIN, where the per-channel scale and bias are predicted from the style vector $\mathbf{w}$ via a linear layer.

Table 5: Optimizer hyperparameters used for all runs

Hyperparameter	**Value**
Optimizer	AdamW
Max. gradient steps	125,000
Learning rate η	3×10^{-4}
Batch size	128
Gradient clip	0.5
β_1	0.9
β_2	0.999
ε	1×10^{-8}

Table 6: Model hyperparameters for the proposed HRR autoencoder. m is given for Shapes3D/Falcor3D/Isaac3D/MPI3D-C and is approximately $1.5\times$ the number of generative factors. All other values are shared across datasets.

Hyperparameter	**Symbol**	**Value**
Latent dimension	d_z	512
Number of symbols	m	9/10/14/10
Codebook size	k	512
Commitment cost	β	0.6
Latent reg. weight	λ_{lat}	0
Codebook reg. weight	λ_{cb}	1×10^{-5}
VQ loss weight	λ_{vq}	0.25
Reconstruction weight	λ_{rec}	1.0
Learning rate	η	3×10^{-4}
Codebook learning rate	η_{cb}	2η
Weight decay	—	0

Table 7: Regularization hyperparameter tuning for each baseline

method	**hyperparameter**	**values**
β-VAE (Higgins et al., 2017)	$\beta = \lambda_{\text{KL}}$	[0.1, 0.3, 1, 3, 10]
β-TCVAE (Chen et al., 2018)	$\beta = \lambda_{\text{total correlation}}$	[0.1, 0.3, 1, 3, 10]
VQ-VAE (van den Oord et al., 2017)	weight decay	[0.001, 0.01, 0.1, 1]
QLAE (Hsu et al., 2023)	weight decay	[0.001, 0.01, 0.1, 1]

Table 8: Selected regularization hyperparameters for each model and dataset. Values were chosen from the corresponding hyperparameter sweeps.

model	Shapes3D	Falcor3D	Isaac3D	MPI3D-C
β-VAE	$\beta = 10$	$\beta = 10$	$\beta = 3$	$\beta = 10$
β-TCVAE	$\beta = 3$	$\beta = 3$	$\beta = 3$	$\beta = 3$
VQ-VAE	$\lambda_{\text{wd}} = .001$	$\lambda_{\text{wd}} = .01$	$\lambda_{\text{wd}} = .001$	$\lambda_{\text{wd}} = 1$
QLAE	$\lambda_{\text{wd}} = 1$	$\lambda_{\text{wd}} = 1$	$\lambda_{\text{wd}} = 1$	$\lambda_{\text{wd}} = 1$

F LIMITATIONS

Empirical comparisons Our empirical comparison is limited by the set of baselines that could be evaluated under a shared experimental protocol. In particular, we considered including Tripod (Hsu et al., 2024), a recent follow-up to QLAE (Hsu et al., 2023) that combines multiple inductive biases for disentanglement and reports strong InfoMEC and DCI performance. However, Tripod differs from the other baselines in its architecture and computational requirements. One component of its optimization involves estimating Hessian-based regularization terms during training, which substantially increases training time. In addition, the architecture proposed for Tripod uses residual connections and differed from the CNN backbone used for the other methods in our comparison.

We evaluated Tripod on our CNN backbone shared by the other methods, with a hyperparameter sweep selecting its configuration, and report the results in Appendix I.

Our claims should be interpreted as comparisons against the baselines evaluated under our shared-backbone protocol, rather than as a definitive comparison against all recent disentanglement methods. Our main contribution is the investigation of VSA-inspired representations as an inductive bias for unsupervised disentanglement, rather than establishing absolute state-of-the-art disentanglement performance across all possible architectures and objectives.

Latent-to-source ratio The choice of using a $1.5\times$ latents to sources ratio differs from the $2\times$ ratio that was used in Hsu et al. (2023). The reason for this is that unbinding performance degrades with m in our model since the unbinding noise becomes worse the more symbol-value pairs are bound together. Thus, we use a $1.5\times$ ratio to maintain the difficulty of the problem without penalizing performance of the HRR-based model.

Increased autoencoder parameter count with HRRs We point out that due to the larger size of the latent vector produced by our model, the autoencoder inevitably ends up having more parameters due to the larger projection matrices at the bottleneck. In our experiments, the autoencoder for our HRR model has 15.4M parameters, against 14.9M for β-VAE, β-TCVAE and QLAE and 15.5M to 15.8M for VQ-VAE, whose autoencoder is the largest on every dataset. The codebook generator and denoiser add a further 1.8M parameters to our model, for 17.3M in total. Although this causes the model architectures to not be completely identical, we argue that this is a feature of our model design, as the representations of prior disentanglement works are limited to keeping vector dimensionality close to the number of ground truth factors. Thus they are not able to take advantage of increasing model capacity without also hurting disentanglement performance by creating excessive amounts of latent units.

G ABLATIONS AND SENSITIVITY

G.1 EFFECT OF HRR LATENT DIMENSION AND CODEBOOK SIZE

Although we chose a single configuration for our HRR model for results, we run sweeps over each dataset for 9 combinations of different latent dimension and codebook sizes. The results for each disentanglement metric, modularity, explicitness, and compactness, are included in Figs. 5 6 7, respectively.

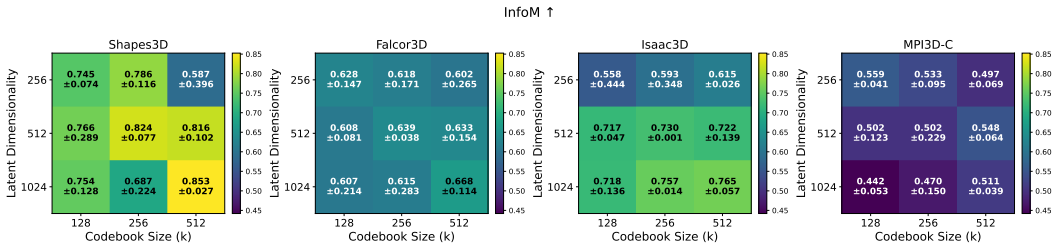


Figure 5: InfoM performance over different combinations of codebook size and latent dimension

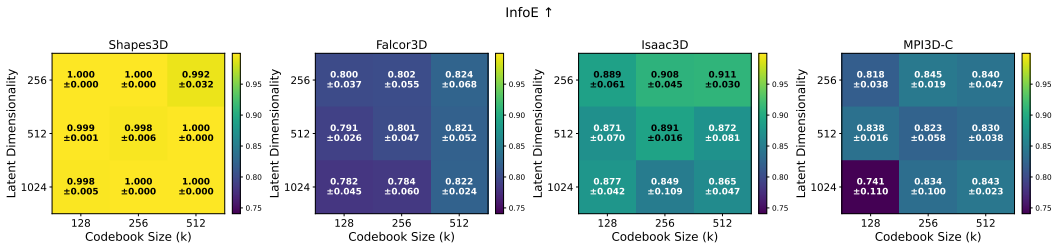


Figure 6: InfoE performance over different combinations of codebook size and latent dimension

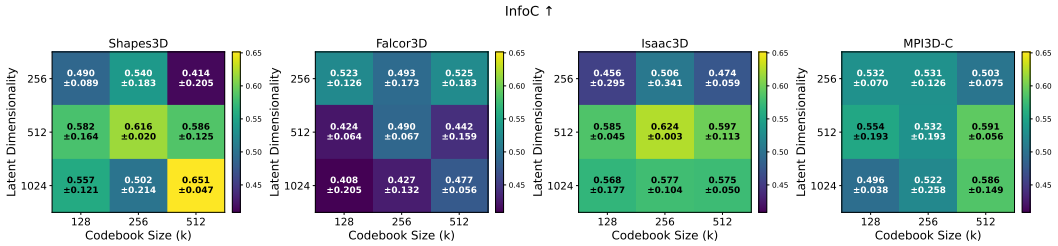


Figure 7: InfoC performance over different combinations of codebook size and latent dimension

H COMPLETE DISENTANGLEMENT RESULTS

The full disentanglement results for each dataset, with the mean and 95% confidence intervals described in Appendix E, can be found in Tables 9 to 12. Bold marks the best per column using the CI-overlap criterion, and ↑ means higher is better.

Table 9: Full results for Shapes3D

model	InfoM ↑	InfoE ↑	InfoC ↑	D ↑	I ↑	C ↑	PSNR (dB) ↑
β-VAE	0.59 ± 0.04	0.8 ± 0.1	0.47 ± 0.04	0.64 ± 0.09	$\mathbf{0.997 \pm 0.005}$	$\mathbf{0.55 \pm 0.06}$	38.9 ± 0.3
β-TCVAE	0.6 ± 0.1	0.8 ± 0.1	0.51 ± 0.04	0.6 ± 0.2	$\mathbf{0.98 \pm 0.03}$	$\mathbf{0.5 \pm 0.2}$	40.8 ± 0.8
VQ-VAE	0.6 ± 0.1	$\mathbf{1.000 \pm 0.005}$	0.44 ± 0.06	0.74 ± 0.07	$\mathbf{0.98 \pm 0.01}$	0.35 ± 0.02	$\mathbf{44.8 \pm 0.3}$
QLAE	0.8 ± 0.1	$\mathbf{0.998 \pm 0.005}$	0.52 ± 0.07	0.76 ± 0.08	$\mathbf{0.993 \pm 0.006}$	0.35 ± 0.04	36 ± 1
HRR (ours)	$\mathbf{0.85 \pm 0.02}$	$\mathbf{0.999 \pm 0.005}$	$\mathbf{0.61 \pm 0.03}$	$\mathbf{0.87 \pm 0.03}$	$\mathbf{0.998 \pm 0.005}$	$\mathbf{0.40 \pm 0.01}$	41 ± 4

Table 10: Full results for Falcor3D

model	InfoM ↑	InfoE ↑	InfoC ↑	D ↑	I ↑	C ↑	PSNR (dB) ↑
β-VAE	$\mathbf{0.64 \pm 0.03}$	0.68 ± 0.02	$\mathbf{0.59 \pm 0.07}$	0.40 ± 0.04	0.83 ± 0.01	$\mathbf{0.39 \pm 0.04}$	29.8 ± 0.2
β-TCVAE	$\mathbf{0.6 \pm 0.2}$	0.65 ± 0.02	$\mathbf{0.6 \pm 0.2}$	0.3 ± 0.2	0.79 ± 0.06	$\mathbf{0.3 \pm 0.2}$	31.0 ± 0.1
VQ-VAE	$\mathbf{0.50 \pm 0.07}$	$\mathbf{0.86 \pm 0.03}$	0.40 ± 0.03	$\mathbf{0.64 \pm 0.04}$	0.81 ± 0.02	$\mathbf{0.340 \pm 0.007}$	$\mathbf{31.4 \pm 0.2}$
QLAE	$\mathbf{0.61 \pm 0.07}$	0.79 ± 0.01	$\mathbf{0.46 \pm 0.07}$	0.50 ± 0.03	0.84 ± 0.02	$\mathbf{0.28 \pm 0.01}$	28.3 ± 0.4
HRR (ours)	$\mathbf{0.64 \pm 0.02}$	0.81 ± 0.02	$\mathbf{0.47 \pm 0.01}$	0.58 ± 0.03	$\mathbf{0.86 \pm 0.02}$	$\mathbf{0.37 \pm 0.01}$	29.2 ± 0.2

Table 11: Full results for Isaac3D

model	InfoM ↑	InfoE ↑	InfoC ↑	D ↑	I ↑	C ↑	PSNR (dB) ↑
β-VAE	0.54 ± 0.06	0.56 ± 0.03	0.46 ± 0.06	0.31 ± 0.05	0.84 ± 0.03	0.30 ± 0.04	37.0 ± 0.4
β-TCVAE	0.63 ± 0.06	0.62 ± 0.06	$\mathbf{0.6 \pm 0.1}$	0.35 ± 0.04	$\mathbf{0.87 \pm 0.01}$	0.34 ± 0.04	37.2 ± 0.4
VQ-VAE	0.70 ± 0.05	$\mathbf{0.86 \pm 0.08}$	0.5 ± 0.1	$\mathbf{0.64 \pm 0.09}$	$\mathbf{0.91 \pm 0.07}$	$\mathbf{0.36 \pm 0.06}$	$\mathbf{45 \pm 2}$
QLAE	0.66 ± 0.06	0.81 ± 0.05	0.49 ± 0.05	0.59 ± 0.04	$\mathbf{0.92 \pm 0.03}$	0.31 ± 0.02	38 ± 4
HRR (ours)	$\mathbf{0.72 \pm 0.04}$	$\mathbf{0.89 \pm 0.04}$	$\mathbf{0.60 \pm 0.05}$	$\mathbf{0.68 \pm 0.03}$	$\mathbf{0.93 \pm 0.02}$	$\mathbf{0.38 \pm 0.02}$	$\mathbf{43.2 \pm 0.4}$

Table 12: Full results for MPI3D-C

model	InfoM ↑	InfoE ↑	InfoC ↑	D ↑	I ↑	C ↑	PSNR (dB) ↑
β-VAE	0.37 ± 0.05	0.23 ± 0.03	0.29 ± 0.09	0.08 ± 0.03	0.57 ± 0.02	0.10 ± 0.03	29.5 ± 0.3
β-TCVAE	0.41 ± 0.01	0.42 ± 0.06	0.39 ± 0.06	0.27 ± 0.02	0.75 ± 0.01	0.26 ± 0.02	35.9 ± 0.1
VQ-VAE	0.5 ± 0.1	0.82 ± 0.03	0.53 ± 0.05	$\mathbf{0.49 \pm 0.08}$	0.73 ± 0.07	0.34 ± 0.02	34.9 ± 0.5
QLAE	$\mathbf{0.56 \pm 0.05}$	0.78 ± 0.04	0.48 ± 0.02	$\mathbf{0.51 \pm 0.04}$	$\mathbf{0.78 \pm 0.03}$	0.29 ± 0.02	34.2 ± 0.3
HRR (ours)	0.53 ± 0.03	$\mathbf{0.85 \pm 0.01}$	$\mathbf{0.54 \pm 0.04}$	$\mathbf{0.55 \pm 0.01}$	$\mathbf{0.78 \pm 0.01}$	$\mathbf{0.35 \pm 0.02}$	$\mathbf{37.0 \pm 0.2}$

Table 13: Shapes3D: disentanglement, paired Δ(ablated variant − full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance.

Variant	ΔInfoM ↑	ΔInfoE ↑	ΔInfoC ↑	ΔDCI-D ↑
w/o codebook generator	-0.784 ± 0.053 *	$+0.000 \pm 0.001$	-0.582 ± 0.041 *	-0.247 ± 0.055 *
w/o codebook reg.	-0.345 ± 0.166 *	$+0.000 \pm 0.001$	-0.190 ± 0.174 *	-0.245 ± 0.133 *
w/o denoiser	-0.363 ± 0.041 *	$+0.000 \pm 0.001$	-0.258 ± 0.047 *	-0.061 ± 0.067
w/o VQ (continuous slots)	-0.521 ± 0.061 *	$+0.000 \pm 0.001$	-0.596 ± 0.036 *	-0.210 ± 0.078 *
learned symbol vectors	-0.078 ± 0.103	$+0.000 \pm 0.001$	-0.033 ± 0.111	-0.053 ± 0.064

Table 14: Shapes3D: reconstruction and latent-noise robustness, paired Δ(ablated variant − full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance. PSNR under latent noise is computed per run from the reconstruction MSE pooled over 512 validation images, as in Figure 1. The clean ΔPSNR column averages per-batch PSNR over the validation set.

Variant	ΔPSNR (dB) ↑	ΔPSNR @ 0 dB ↑	ΔPSNR @ −10 dB ↑
w/o codebook generator	-9.74 ± 4.36 *	-8.19 ± 0.92 *	-5.55 ± 0.56 *
w/o codebook reg.	$+1.01 \pm 4.05$	-2.19 ± 1.62 *	-0.87 ± 1.21
w/o denoiser	$+1.54 \pm 4.31$	$+3.14 \pm 1.98$ *	-0.56 ± 0.67
w/o VQ (continuous slots)	$+3.96 \pm 4.31$	$+1.32 \pm 1.40$	-1.22 ± 0.53 *
learned symbol vectors	$+2.44 \pm 4.84$	$+0.44 \pm 0.94$	-0.00 ± 0.93

H.1 Component ablations and sensitivity

The ablation tables in this section and the binding table below are paired by seed against an HRR reference that reruns the configuration of Table 2. The Tripod and Soft TPR tables use the HRR row of Table 2. Each table caption states the number of seeds.

Component ablations. We ablate each component of the HRR model on all four datasets at a fixed latent dimensionality of $d = 512$. Each variant changes one component. Four variants remove the codebook generator, the codebook regularizer, the denoiser, or vector quantization, which lets each slot take continuous values instead of one codebook entry. One variant learns the symbol vectors. Every entry in Tables 13 to 20 is the paired per-seed difference between the ablated variant and the full HRR model over five seeds, and a star marks a 95% CI that excludes zero.

We find that vector quantization, the denoiser and the codebook generator contribute most to disentanglement. Removing any one of them lowers InfoM, and the denoiser alone accounts for between 0.248 and 0.421 InfoM. The codebook regularizer contributes measurably on Shapes3D and MPI3D-C and has no statistically significant effect on Falcor3D and Isaac3D. Finally, learned symbol vectors appear to have little effect on disentanglement across datasets. Tables 14, 16, 18 and 20 give reconstruction PSNR and the PSNR under latent noise for the same variants.

Table 15: Falcor3D: disentanglement, paired Δ(ablated variant $-$ full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance.

Variant	ΔInfoM $\uparrow$	ΔInfoE $\uparrow$	ΔInfoC $\uparrow$	ΔDCI-D $\uparrow$
w/o codebook generator	-0.559 ± 0.083 *	$+0.185 \pm 0.021$ *	-0.461 ± 0.048 *	-0.070 ± 0.060 *
w/o codebook reg.	-0.103 ± 0.195	$+0.018 \pm 0.033$	-0.009 ± 0.169	$+0.049 \pm 0.203$
w/o denoiser	-0.348 ± 0.071 *	$+0.096 \pm 0.033$ *	-0.232 ± 0.048 *	$+0.084 \pm 0.058$ *
w/o VQ (continuous slots)	-0.408 ± 0.077 *	$+0.184 \pm 0.021$ *	-0.504 ± 0.042 *	-0.213 ± 0.052 *
learned symbol vectors	-0.030 ± 0.130	-0.003 ± 0.040	-0.032 ± 0.052	-0.034 ± 0.075

Table 16: Falcor3D: reconstruction and latent-noise robustness, paired Δ(ablated variant $-$ full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance. PSNR under latent noise is computed per run from the reconstruction MSE pooled over 512 validation images, as in Figure 1. The clean ΔPSNR column averages per-batch PSNR over the validation set.

Variant	ΔPSNR (dB) $\uparrow$	ΔPSNR @ 0 dB $\uparrow$	ΔPSNR @ -10 dB $\uparrow$
w/o codebook generator	-1.58 ± 0.31 *	-1.78 ± 0.68 *	-2.97 ± 0.52 *
w/o codebook reg.	$+1.17 \pm 0.48$ *	-1.01 ± 0.46 *	-0.02 ± 0.67
w/o denoiser	-0.22 ± 0.61	-0.13 ± 0.75	-1.18 ± 0.42 *
w/o VQ (continuous slots)	$+2.99 \pm 0.46$ *	-0.52 ± 0.39 *	-1.74 ± 0.77 *
learned symbol vectors	-0.36 ± 1.80	$+1.02 \pm 0.39$ *	$+0.29 \pm 0.43$

Table 17: Isaac3D: disentanglement, paired Δ(ablated variant $-$ full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance.

Variant	ΔInfoM $\uparrow$	ΔInfoE $\uparrow$	ΔInfoC $\uparrow$	ΔDCI-D $\uparrow$
w/o codebook generator	-0.256 ± 0.048 *	-0.064 ± 0.039 *	-0.246 ± 0.076 *	-0.156 ± 0.032 *
w/o codebook reg.	-0.065 ± 0.132	$+0.008 \pm 0.048$	-0.019 ± 0.071	-0.022 ± 0.075
w/o denoiser	-0.421 ± 0.080 *	$+0.079 \pm 0.020$ *	-0.303 ± 0.086 *	-0.008 ± 0.055
w/o VQ (continuous slots)	-0.416 ± 0.051 *	$+0.128 \pm 0.024$ *	-0.536 ± 0.064 *	-0.172 ± 0.050 *
learned symbol vectors	-0.059 ± 0.060	$+0.027 \pm 0.031$	$+0.045 \pm 0.113$	$+0.011 \pm 0.064$

Table 18: Isaac3D: reconstruction and latent-noise robustness, paired Δ(ablated variant $-$ full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance. PSNR under latent noise is computed per run from the reconstruction MSE pooled over 512 validation images, as in Figure 1. The clean ΔPSNR column averages per-batch PSNR over the validation set.

Variant	ΔPSNR (dB) $\uparrow$	ΔPSNR @ 0 dB $\uparrow$	ΔPSNR @ -10 dB $\uparrow$
w/o codebook generator	-6.42 ± 2.51 *	-0.35 ± 0.97	-1.49 ± 0.36 *
w/o codebook reg.	$+0.68 \pm 1.71$	-0.15 ± 0.71	$+0.72 \pm 0.89$
w/o denoiser	-2.63 ± 2.10 *	-2.82 ± 1.74 *	-1.10 ± 0.59 *
w/o VQ (continuous slots)	$+4.40 \pm 1.85$ *	-0.45 ± 1.33	-1.18 ± 0.79 *
learned symbol vectors	$+0.58 \pm 2.35$	$+2.24 \pm 1.02$ *	$+1.47 \pm 0.31$ *

Table 19: MPI3D-C: disentanglement, paired Δ(ablated variant $-$ full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance.

Variant	ΔInfoM $\uparrow$	ΔInfoE $\uparrow$	ΔInfoC $\uparrow$	ΔDCI-D $\uparrow$
w/o codebook generator	-0.272 ± 0.067 *	$+0.138 \pm 0.023$ *	-0.311 ± 0.057 *	-0.068 ± 0.057 *
w/o codebook reg.	-0.065 ± 0.026 *	$+0.045 \pm 0.022$ *	-0.028 ± 0.036	-0.016 ± 0.037
w/o denoiser	-0.248 ± 0.057 *	$+0.078 \pm 0.036$ *	-0.125 ± 0.069 *	$+0.028 \pm 0.049$
w/o VQ (continuous slots)	-0.279 ± 0.038 *	$+0.169 \pm 0.025$ *	-0.546 ± 0.047 *	-0.233 ± 0.046 *
learned symbol vectors	-0.018 ± 0.086	$+0.024 \pm 0.033$	$+0.003 \pm 0.054$	-0.009 ± 0.050

Table 20: MPI3D-C: reconstruction and latent-noise robustness, paired Δ(ablated variant $-$ full HRR model). The HRR reference reruns the paper configuration over five seeds and differs from Table 2 by rerun variance. PSNR under latent noise is computed per run from the reconstruction MSE pooled over 512 validation images, as in Figure 1. The clean ΔPSNR column averages per-batch PSNR over the validation set.

Variant	ΔPSNR (dB) $\uparrow$	ΔPSNR @ 0 dB $\uparrow$	ΔPSNR @ -10 dB $\uparrow$
w/o codebook generator	$-2.57 \pm 0.50^{*}$	$+2.59 \pm 1.00^{*}$	$+0.16 \pm 0.60$
w/o codebook reg.	$+1.06 \pm 0.22^{*}$	$-1.04 \pm 0.83^{*}$	$-1.15 \pm 0.71^{*}$
w/o denoiser	$-3.68 \pm 0.73^{*}$	$+1.30 \pm 1.28^{*}$	$+0.00 \pm 1.05$
w/o VQ (continuous slots)	$+4.27 \pm 0.32^{*}$	$+2.40 \pm 0.50^{*}$	$+0.43 \pm 0.38^{*}$
learned symbol vectors	$+0.45 \pm 0.36^{*}$	$+3.87 \pm 0.58^{*}$	$+2.05 \pm 0.51^{*}$

Table 21: Latent-to-source ratio sensitivity (HRR, seeds 42–44, mean ± 95% CI, $t = 4.303$). m is the number of HRR cues; the ratio is m over the number of ground-truth factors. The $\approx 1.5\times$ row is the submitted configuration. * marks a ratio whose *paired* per-seed difference from the submitted configuration has a 95% CI excluding zero. Disentanglement is flat between $\approx 1.5\times$ and $2.0\times$ on every dataset. InfoC is reported for completeness but is *not* comparable across m: its plug-in mutual-information estimator carries a positive bias per latent, so a perfectly compact representation also scores lower as m grows. The submitted row is the HRR reference, a rerun of the paper configuration over three seeds, and differs from Table 2 by rerun variance.

Dataset	Ratio	m	InfoM ↑	InfoE ↑	InfoC ↑	DCI-D ↑	PSNR (dB) ↑
Shapes3D	$1.0\times$	6	0.575 ± 0.260	0.999 ± 0.003	0.595 ± 0.184	0.698 ± 0.144 *	41.47 ± 4.75
	$\approx 1.5\times$ (subm.)	9	0.837 ± 0.052	0.999 ± 0.002	0.598 ± 0.079	0.869 ± 0.023	39.67 ± 11.63
	$2.0\times$	12	0.845 ± 0.058	1.000 ± 0.000	0.473 ± 0.065	0.874 ± 0.025	40.27 ± 11.91
Falcor3D	$1.0\times$	7	0.678 ± 0.311	0.814 ± 0.071	0.657 ± 0.257	0.621 ± 0.201	28.59 ± 0.95
	$\approx 1.5\times$ (subm.)	10	0.649 ± 0.123	0.810 ± 0.045	0.502 ± 0.108	0.607 ± 0.078	29.04 ± 0.52
	$2.0\times$	14	0.645 ± 0.134	0.846 ± 0.025 *	0.408 ± 0.065	0.618 ± 0.134	29.87 ± 0.81 *
Isaac3D	$1.0\times$	9	0.723 ± 0.146	0.880 ± 0.080	0.670 ± 0.315	0.670 ± 0.097	41.68 ± 3.11
	$\approx 1.5\times$ (subm.)	14	0.709 ± 0.122	0.880 ± 0.053	0.523 ± 0.165	0.664 ± 0.100	43.23 ± 1.09
	$2.0\times$	18	0.735 ± 0.067	0.890 ± 0.047	0.489 ± 0.045	0.683 ± 0.023	43.19 ± 1.37
MPI3D-C	$1.0\times$	7	0.454 ± 0.237	0.823 ± 0.084	0.568 ± 0.125	0.519 ± 0.110	36.78 ± 0.61
	$\approx 1.5\times$ (subm.)	10	0.551 ± 0.095	0.831 ± 0.054	0.528 ± 0.057	0.563 ± 0.136	37.06 ± 0.21
	$2.0\times$	14	0.561 ± 0.043	0.867 ± 0.037	0.496 ± 0.038	0.597 ± 0.026	37.66 ± 0.93

Table 22: Binding algebra × codebook regularizer at matched capacity, budget and protocol. HLB is the Hadamard-derived Linear Binding VSA (Alam et al., 2024). Mean ± 95% CI over three seeds. * marks a binding whose paired per-seed difference from HRR at the same λ_{cb} excludes zero. The HRR rows at the submitted λ_{cb} are the HRR reference, a rerun of the paper configuration over three seeds, and differ from Table 2 by rerun variance.

Dataset	λ_{cb}	Binding	InfoM ↑	InfoC ↑	DCI-D ↑	PSNR (dB) ↑
Shapes3D	1×10^{-5}	HRR (ours)	0.837 ± 0.052	0.598 ± 0.079	0.869 ± 0.023	39.67 ± 11.63
		HLB	0.647 ± 0.130 *	0.458 ± 0.212	0.773 ± 0.093	42.19 ± 2.74
	0	HRR (ours)	0.462 ± 0.437	0.367 ± 0.396	0.597 ± 0.332	41.53 ± 1.38
		HLB	0.316 ± 0.418	0.284 ± 0.276	0.551 ± 0.442	41.70 ± 0.06
MPI3D-C	1×10^{-5}	HRR (ours)	0.551 ± 0.095	0.528 ± 0.057	0.563 ± 0.136	37.06 ± 0.21
		HLB	0.477 ± 0.149	0.527 ± 0.096	0.558 ± 0.015	37.05 ± 0.03
	0	HRR (ours)	0.499 ± 0.080	0.506 ± 0.135	0.537 ± 0.069	38.12 ± 0.44
		HLB	0.438 ± 0.106	0.449 ± 0.139	0.497 ± 0.149	38.05 ± 0.49

Latent-to-source ratio. We set the number of symbol vectors m to $1.0\times$, $1.5\times$, and $2.0\times$ the number of ground-truth factors and train the HRR model over three seeds at each setting, and Table 21 shows the results. The $1.5\times$ ratio is the configuration used for the main results. InfoM plateaus past the $1.5\times$ ratio on most datasets. The $1.0\times$ ratio gives the lowest InfoM on Shapes3D and MPI3D-C, while also having the widest interval across all datasets.

Binding algebra and codebook regularizer. We replace HRR binding with the HLB vector symbolic architecture (Alam et al., 2024), which is also real valued, on Shapes3D and MPI3D-C over three seeds, running each binding with and without the codebook regularizer, and Table 22 shows the results. HRR reaches a higher mean InfoM than HLB on both datasets, by 0.190 on Shapes3D and by 0.074 on MPI3D-C. This experiment also shows that the codebook regularizer is essential regardless of the binding. Removing it lowers the mean of every disentanglement metric under both architectures on both datasets, and lowers InfoM by 0.375 under HRR and by 0.331 under HLB on Shapes3D.

Table 23: Tripod at its sweep-selected configuration versus HRR (shared QLAE backbone, 125k steps). Mean ± 95% CI, HRR over five seeds and Tripod over three. ‡ one seed collapses even at the sweep-selected configuration, so the mean averages healthy and failed runs.

Dataset	Model	InfoM ↑	InfoC ↑	DCI-D ↑	PSNR (dB) ↑
Shapes3D	HRR (ours)	0.850 ± 0.022	0.609 ± 0.029	0.867 ± 0.028	40.58 ± 3.69
	Tripod (sweep-selected) ‡	0.717 ± 0.589	0.392 ± 0.842	0.516 ± 1.110	30.22 ± 39.80
Falcor3D	HRR (ours)	0.644 ± 0.020	0.468 ± 0.010	0.584 ± 0.028	29.21 ± 0.23
	Tripod (sweep-selected)	0.568 ± 0.138	0.487 ± 0.231	0.284 ± 0.083	28.79 ± 0.37
Isaac3D	HRR (ours)	0.721 ± 0.038	0.600 ± 0.054	0.676 ± 0.032	43.23 ± 0.45
	Tripod (sweep-selected)	0.729 ± 0.108	0.607 ± 0.127	0.498 ± 0.064	42.90 ± 0.81

Table 24: Soft TPR (Sun et al., 2024) trained under our budget and scored by our metric pipeline, against the HRR reference. Mean ± 95% CI over five seeds. **Bold** marks entries tied with the leader among the unsupervised rows. `us` zeroes Soft TPR's weak-supervision losses and uses unpaired batches, which matches our setting, and `ws` trains on paired observations with known factor changes. † marks a metric where `ws` is strictly ahead of HRR. Soft TPR uses its own encoder at $1.63\times$ to $1.64\times$ the parameter count of our encoder.

Dataset	Model	Sup.	InfoM ↑	InfoC ↑	DCI-D ↑	DCI-C ↑
Shapes3D	HRR (ours)	none	$\mathbf{0.850 \pm 0.022}$	$\mathbf{0.609 \pm 0.029}$	$\mathbf{0.867 \pm 0.028}$	$\mathbf{0.400 \pm 0.011}$
	Soft TPR (us)	none	0.596 ± 0.095	0.413 ± 0.080	0.696 ± 0.076	0.352 ± 0.027
	Soft TPR (ws)	paired GT	0.696 ± 0.146	0.480 ± 0.114	0.740 ± 0.118	0.371 ± 0.060
MPI3D-C	HRR (ours)	none	$\mathbf{0.531 \pm 0.034}$	$\mathbf{0.535 \pm 0.042}$	$\mathbf{0.545 \pm 0.014}$	$\mathbf{0.346 \pm 0.021}$
	Soft TPR (us)	none	0.373 ± 0.038	0.422 ± 0.038	0.395 ± 0.026	0.323 ± 0.036
	Soft TPR (ws)	paired GT	$0.756 \pm 0.088^{\dagger}$	$0.763 \pm 0.131^{\dagger}$	$0.705 \pm 0.030^{\dagger}$	$0.495 \pm 0.027^{\dagger}$

I Additional baselines

Tripod. We ran Tripod (Hsu et al., 2024), a more modern disentanglement baseline, on the shared QLAE backbone for all four datasets. Many runs at the authors' recommended per-dataset hyperparameters collapsed so we ran a short hyperparameter sweep to find better values. The results with the new hyperparameters are in Table 23. HRR reaches higher DCI-D than Tripod on Shapes3D, Falcor3D and Isaac3D. HRR has the higher mean InfoM on Shapes3D, Falcor3D and matches Tripod's InfoM on Isaac3D. No configuration in our sweep avoided latent collapse on MPI3D-C, thus we omit the results for it.

Soft TPR. Soft TPR (Sun et al., 2024) learns compositional representations with a soft tensor product structure and is the closest published method to ours. However, we note some key differences that distinguish our method. Firstly, Soft TPR is designed to train on paired observations with known factor changes. Our model is designed and evaluated without any supervision, on unpaired images. Soft TPR can also be trained with its weak-supervision losses set to zero and unpaired batches. Table 24 reports that setting as `us`. We find that the unsupervised version of Soft TPR scores below HRR on every metric in Table 24 on both Shapes3D and MPI3D-C. With weak supervision, Soft TPR scores above HRR on MPI3D-C. Soft TPR binds its roles and fillers with a tensor product, so the representation dimension grows with the product of the role and filler dimensions. HRR binds by circular convolution at a fixed dimension d, without expanding the size of the latent.

Table 25: Probe score on MPI3D-C (narrow recombination-to-range), split by factor family. Mean ± 95% CI over five seeds. Deficit is in-distribution minus holdout. **Bold** marks the best holdout and the deficit closest to zero in each family.

	accuracy (5 categorical, chance 0.333)		R^{2+} (2 ordered, floor 0)	
Model	Holdout ↑	Deficit ↓	Holdout ↑	Deficit ↓
HRR (ours)	0.922 ± 0.088	0.063 ± 0.083	$\mathbf{0.404 \pm 0.050}$	-0.018 ± 0.015
VQ-VAE	$\mathbf{0.932 \pm 0.027}$	$\mathbf{0.013 \pm 0.020}$	0.277 ± 0.040	-0.016 ± 0.008
QLAE	0.775 ± 0.050	0.073 ± 0.060	0.281 ± 0.098	$\mathbf{-0.010 \pm 0.012}$
β-TCVAE	0.635 ± 0.085	0.082 ± 0.011	0.214 ± 0.050	-0.037 ± 0.016

J COMPOSITIONAL GENERALIZATION

Protocol. We follow Montero et al. (2021), who distinguish recombination-to-element (R2E), which withholds one combination of every generative factor, from recombination-to-range (R2R), which withholds all combinations of a subset of factors while the remaining factors span their full range. Standard models already succeed at R2E, so we evaluate R2R at two scales. The broad R2R condition withholds 12.5% of each dataset. On Shapes3D it withholds every image whose object shape is a pill and whose object hue is at or above 0.5, the shape-to-object-hue split of Montero et al. (2021). On MPI3D-C it withholds every image whose object shape is 2 and whose horizontal axis is below 20. The narrow R2R condition withholds a single pairing of object color and object shape on MPI3D-C, 6.25% of the dataset, with the remaining five factors spanning their full range. In every condition the model sees each factor value during training and is evaluated on combinations of those values that it never encountered.

Following the protocol of Xu et al. (2022), we evaluate each representation with linear probes fit on 500 random training images, and call the probe's accuracy or R^2 on a set of images its probe score. Categorical factors use logistic regression and report accuracy. Ordered factors use ridge regression and report R^{2+}, the coefficient of determination clipped at zero. A pooled score is the mean over factors of these per-factor scores, and the per-family scores in the tables below average over the factors of that family. The deficit is the in-distribution score minus the held-out score.

Results. Under narrow R2R (Table 25), HRR scores highest in the ordered family and is statistically indistinguishable from VQ-VAE in the categorical family. HRR shows no measurable compositional deficit in the categorical family, and in the ordered family its score on withheld combinations is higher than on seen ones by 0.018 ± 0.015. Pooled over factors and paired by seed, HRR's held-out score is ahead of VQ-VAE, the only other vector-valued representation here, on four of five seeds, a difference that is not statistically significant.

Under broad R2R, HRR has the highest held-out probe score in both factor families on MPI3D-C (Table 27) and on Shapes3D (Table 26). On the Shapes3D shape factor, HRR and QLAE are statistically indistinguishable. Pooled over factors and paired by seed, HRR's held-out score is ahead of VQ-VAE on every seed in both broad conditions, and the lead is statistically significant on MPI3D-C. The ordered family on MPI3D-C sits low for every model because one of its two factors is the dimension whose range was withheld.

We note that probe input width differs across models (Table 28). VQ-VAE, for instance, has a wider representation than HRR's 512 dimensions. All four models share the same convolutional encoder, so Table 29 also probes the shared encoder output, which has 8192 dimensions for every model. At the encoder output at least one baseline scores above HRR in every condition. From encoder output to bottleneck, HRR retains the most held-out probe score in all three conditions. Tables 30 to 35 give the per-factor probe scores.

Table 26: Probe score on Shapes3D (recombination-to-range), split by factor family. Mean ± 95% CI over five seeds. Deficit is in-distribution minus holdout. **Bold** marks the best holdout and the deficit closest to zero in each family. Shapes3D has a single categorical factor.

	accuracy (1 categorical, chance 0.250)		R^{2+} (5 ordered, floor 0)	
Model	Holdout ↑	Deficit ↓	Holdout ↑	Deficit ↓
HRR (ours)	$\mathbf{0.936 \pm 0.088}$	$\mathbf{0.064 \pm 0.088}$	$\mathbf{0.808 \pm 0.110}$	0.140 ± 0.081
VQ-VAE	0.811 ± 0.442	0.189 ± 0.442	0.648 ± 0.191	0.208 ± 0.166
QLAE	0.916 ± 0.062	0.084 ± 0.062	0.632 ± 0.040	$\mathbf{0.128 \pm 0.006}$
β-TCVAE	0.499 ± 0.317	0.257 ± 0.351	0.539 ± 0.088	0.143 ± 0.038

Table 27: Probe score on MPI3D-C (recombination-to-range), split by factor family. Mean ± 95% CI over five seeds. Deficit is in-distribution minus holdout. **Bold** marks the best holdout and the deficit closest to zero in each family.

	accuracy (5 categorical, chance 0.333)		R^{2+} (2 ordered, floor 0)	
Model	Holdout ↑	Deficit ↓	Holdout ↑	Deficit ↓
HRR (ours)	$\mathbf{0.922 \pm 0.016}$	0.070 ± 0.012	$\mathbf{0.215 \pm 0.062}$	0.182 ± 0.065
VQ-VAE	0.828 ± 0.056	0.106 ± 0.029	0.140 ± 0.081	0.207 ± 0.107
QLAE	0.766 ± 0.131	$\mathbf{0.062 \pm 0.069}$	0.131 ± 0.115	$\mathbf{0.146 \pm 0.057}$
β-TCVAE	0.634 ± 0.063	0.100 ± 0.033	0.039 ± 0.083	0.175 ± 0.027

Table 28: Probe input width per model, measured from a forward pass rather than inferred from configs. VQ-VAE quantises the same slot count against the same 512-entry codebook as HRR and still presents the wider representation (576 or 640 vs. 512 dimensions).

Condition · dataset	Model	Probe dim	Slots × slot dim
Broad · Shapes3D	HRR (ours)	512	9×512
	VQ-VAE	576	9×64
	QLAE	9	9×1
	β-TCVAE	9	9×1
Broad · MPI3D-C	HRR (ours)	512	10×512
	VQ-VAE	640	10×64
	QLAE	10	10×1
	β-TCVAE	10	10×1
Narrow · MPI3D-C	HRR (ours)	512	10×512
	VQ-VAE	640	10×64
	QLAE	10	10×1
	β-TCVAE	10	10×1

Table 29: Linear probe score on withheld combinations at the shared convolutional encoder output, which is 8192 dimensions for every model, and at each model's own bottleneck. Scores are means over factors, pooling accuracy and clipped R^2. Mean $\pm$ 95% CI over five seeds, and Change is the paired per-seed difference.

Condition · dataset	Model	Encoder output	Bottleneck	Change
Broad · Shapes3D	HRR (ours)	0.810 ± 0.009	0.830 ± 0.088	$+0.020 \pm 0.096$
	VQ-VAE	0.800 ± 0.041	0.675 ± 0.228	-0.125 ± 0.240
	QLAE	0.721 ± 0.014	0.679 ± 0.042	-0.042 ± 0.046
	β-TCVAE	0.812 ± 0.018	0.532 ± 0.086	-0.279 ± 0.082
Broad · MPI3D-C	HRR (ours)	0.700 ± 0.022	0.720 ± 0.022	$+0.020 \pm 0.040$
	VQ-VAE	0.729 ± 0.056	0.631 ± 0.035	-0.098 ± 0.039
	QLAE	0.698 ± 0.038	0.585 ± 0.087	-0.113 ± 0.058
	β-TCVAE	0.595 ± 0.009	0.464 ± 0.054	-0.132 ± 0.047
Narrow · MPI3D-C	HRR (ours)	0.880 ± 0.016	0.774 ± 0.053	-0.106 ± 0.056
	VQ-VAE	0.940 ± 0.006	0.745 ± 0.012	-0.195 ± 0.019
	QLAE	0.915 ± 0.018	0.634 ± 0.028	-0.281 ± 0.028
	β-TCVAE	0.799 ± 0.026	0.514 ± 0.066	-0.284 ± 0.053

Table 30: Per-factor probe score on Shapes3D, recombination-to-range, held-out combinations. Mean ± 95% CI over five seeds. **bold** marks the best model per factor. † marks the factors whose combination was withheld. R^{2+} is clipped at zero, so 0.000 means worse than predicting the mean.

Factor	Metric	Chance	HRR (ours)	VQ-VAE	QLAE	β-TCVAE
floor_hue	R^{2+}	0	$\mathbf{0.940 \pm 0.064}$	0.771 ± 0.254	0.656 ± 0.008	0.583 ± 0.056
wall_hue	R^{2+}	0	$\mathbf{0.941 \pm 0.039}$	0.873 ± 0.124	0.635 ± 0.015	0.618 ± 0.032
object_hue†	R^{2+}	0	$\mathbf{0.178 \pm 0.495}$	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
scale	R^{2+}	0	$\mathbf{0.992 \pm 0.015}$	0.938 ± 0.172	0.926 ± 0.196	0.713 ± 0.426
shape†	acc.	0.250	$\mathbf{0.936 \pm 0.088}$	0.811 ± 0.442	0.916 ± 0.062	0.499 ± 0.317
orientation	R^{2+}	0	$\mathbf{0.990 \pm 0.009}$	0.656 ± 0.467	0.942 ± 0.089	0.782 ± 0.272
unweighted mean	—	—	0.830 ± 0.088	0.675 ± 0.228	0.679 ± 0.042	0.532 ± 0.086

Table 31: Per-factor probe score on Shapes3D, recombination-to-range, in-distribution. Mean ± 95% CI over five seeds. **bold** marks the best model per factor. † marks the factors whose combination was withheld. R^{2+} is clipped at zero, so 0.000 means worse than predicting the mean.

Factor	Metric	Chance	HRR (ours)	VQ-VAE	QLAE	β-TCVAE
floor_hue	R^{2+}	0	$\mathbf{0.938 \pm 0.071}$	0.844 ± 0.153	0.656 ± 0.010	0.584 ± 0.055
wall_hue	R^{2+}	0	$\mathbf{0.958 \pm 0.021}$	0.878 ± 0.120	0.640 ± 0.015	0.618 ± 0.035
object_hue†	R^{2+}	0	$\mathbf{0.854 \pm 0.127}$	0.707 ± 0.058	0.638 ± 0.027	0.559 ± 0.059
scale	R^{2+}	0	0.998 ± 0.003	$\mathbf{0.999 \pm 0.001}$	0.926 ± 0.196	0.872 ± 0.188
shape†	acc.	0.250	1.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.999 ± 0.002	0.756 ± 0.415
orientation	R^{2+}	0	$\mathbf{0.991 \pm 0.009}$	0.848 ± 0.100	0.940 ± 0.089	0.779 ± 0.274
unweighted mean	—	—	0.957 ± 0.030	0.879 ± 0.051	0.800 ± 0.036	0.695 ± 0.103

K ADDITIONAL NOISE RESULTS

We include further results for the noise robustness experiment for the remaining datasets. Figs. 9, 10, 11 show the results on Isaac3D, Falcor3D, and MPI3D-C, respectively.

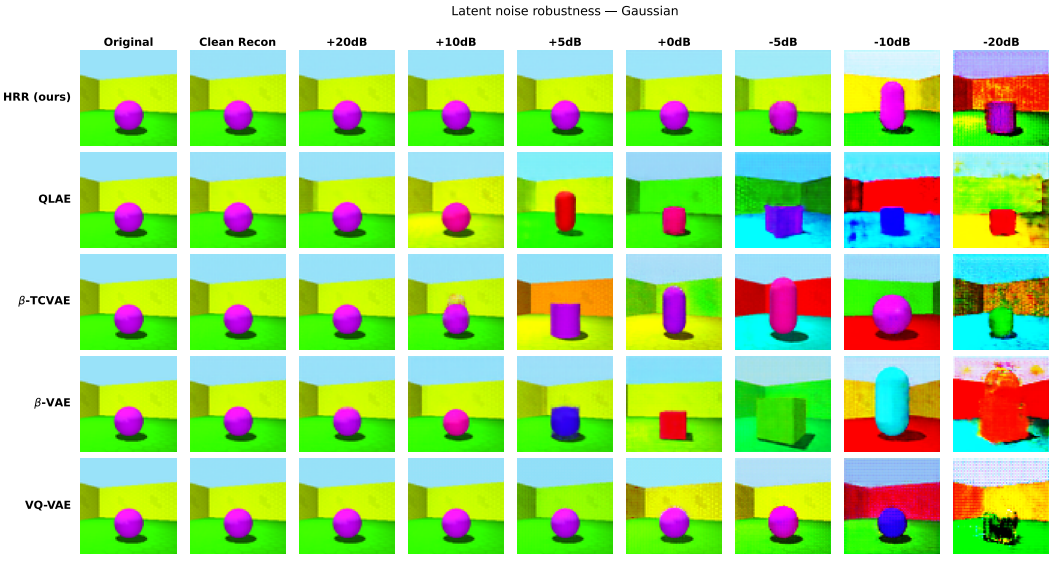


Figure 8: Each row is a visualization of how reconstruction quality degrades as noise intensity increases for each model. For each model, the seed with the highest InfoM score was used for evaluation. The HRR model and VQ-VAE maintain their semantic content the best.

Table 32: Per-factor probe score on MPI3D-C, recombination-to-range, held-out combinations. Mean ± 95% CI over five seeds. **bold** marks the best model per factor. † marks the factors whose combination was withheld. R^{2+} is clipped at zero, so 0.000 means worse than predicting the mean.

Factor	Metric	Chance	HRR (ours)	VQ-VAE	QLAE	β-TCVAE
object_color	acc.	0.250	$\mathbf{0.949 \pm 0.050}$	0.793 ± 0.125	0.876 ± 0.045	0.757 ± 0.039
object_shape†	acc.	0.250	$\mathbf{0.879 \pm 0.058}$	0.829 ± 0.075	0.661 ± 0.263	0.232 ± 0.216
object_size	acc.	0.500	$\mathbf{0.784 \pm 0.058}$	0.618 ± 0.095	0.517 ± 0.007	0.546 ± 0.034
camera_height	acc.	0.333	$\mathbf{1.000 \pm 0.000}$	0.999 ± 0.001	1.000 ± 0.001	1.000 ± 0.000
background_color	acc.	0.333	$\mathbf{0.999 \pm 0.003}$	0.900 ± 0.114	0.778 ± 0.382	0.633 ± 0.173
horizontal_axis†	R^{2+}	0	$\mathbf{0.000 \pm 0.000}$	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
vertical_axis	R^{2+}	0	$\mathbf{0.430 \pm 0.123}$	0.279 ± 0.162	0.263 ± 0.229	0.077 ± 0.166
unweighted mean	—	—	0.720 ± 0.022	0.631 ± 0.035	0.585 ± 0.087	0.464 ± 0.054

Table 33: Per-factor probe score on MPI3D-C, recombination-to-range, in-distribution. Mean ± 95% CI over five seeds. **bold** marks the best model per factor. † marks the factors whose combination was withheld. R^{2+} is clipped at zero, so 0.000 means worse than predicting the mean.

Factor	Metric	Chance	HRR (ours)	VQ-VAE	QLAE	β-TCVAE
object_color	acc.	0.250	$\mathbf{0.995 \pm 0.005}$	0.948 ± 0.060	0.980 ± 0.012	0.954 ± 0.005
object_shape†	acc.	0.250	$\mathbf{0.997 \pm 0.003}$	0.976 ± 0.040	0.727 ± 0.115	0.512 ± 0.076
object_size	acc.	0.500	$\mathbf{0.974 \pm 0.026}$	0.841 ± 0.086	0.660 ± 0.053	0.613 ± 0.051
camera_height	acc.	0.333	$\mathbf{1.000 \pm 0.000}$	0.999 ± 0.003	0.999 ± 0.001	0.999 ± 0.001
background_color	acc.	0.333	$\mathbf{0.996 \pm 0.008}$	0.906 ± 0.115	0.776 ± 0.381	0.589 ± 0.163
horizontal_axis†	R^{2+}	0	$\mathbf{0.260 \pm 0.058}$	0.253 ± 0.164	0.182 ± 0.068	0.164 ± 0.033
vertical_axis	R^{2+}	0	$\mathbf{0.534 \pm 0.043}$	0.440 ± 0.125	0.373 ± 0.167	0.264 ± 0.182
unweighted mean	—	—	0.822 ± 0.011	0.766 ± 0.039	0.671 ± 0.048	0.585 ± 0.040

Table 34: Per-factor probe score on MPI3D-C, narrow recombination-to-range, held-out combinations. Mean ± 95% CI over five seeds. **bold** marks the best model per factor. † marks the factors whose combination was withheld. R^{2+} is clipped at zero, so 0.000 means worse than predicting the mean.

Factor	Metric	Chance	HRR (ours)	VQ-VAE	QLAE	β-TCVAE
object_color†	acc.	0.250	1.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	1.000 ± 0.000	0.871 ± 0.268
object_shape†	acc.	0.250	0.724 ± 0.466	$\mathbf{0.880 \pm 0.054}$	0.405 ± 0.575	0.031 ± 0.066
object_size	acc.	0.500	$\mathbf{0.888 \pm 0.174}$	0.781 ± 0.163	0.741 ± 0.074	0.600 ± 0.035
camera_height	acc.	0.333	1.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	1.000 ± 0.000	1.000 ± 0.000
background_color	acc.	0.333	$\mathbf{0.999 \pm 0.000}$	0.998 ± 0.004	0.730 ± 0.340	0.671 ± 0.256
horizontal_axis	R^{2+}	0	$\mathbf{0.239 \pm 0.066}$	0.126 ± 0.047	0.134 ± 0.067	0.123 ± 0.062
vertical_axis	R^{2+}	0	$\mathbf{0.568 \pm 0.062}$	0.428 ± 0.075	0.428 ± 0.183	0.305 ± 0.039
unweighted mean	—	—	0.774 ± 0.053	0.745 ± 0.012	0.634 ± 0.028	0.514 ± 0.066

Table 35: Per-factor probe score on MPI3D-C, narrow recombination-to-range, in-distribution. Mean ± 95% CI over five seeds. **bold** marks the best model per factor. † marks the factors whose combination was withheld. R^{2+} is clipped at zero, so 0.000 means worse than predicting the mean.

Factor	Metric	Chance	HRR (ours)	VQ-VAE	QLAE	β-TCVAE
object_color†	acc.	0.250	$\mathbf{0.994 \pm 0.004}$	0.985 ± 0.008	0.978 ± 0.009	0.817 ± 0.236
object_shape†	acc.	0.250	0.991 ± 0.013	$\mathbf{0.994 \pm 0.004}$	0.841 ± 0.175	0.518 ± 0.073
object_size	acc.	0.500	$\mathbf{0.943 \pm 0.052}$	0.746 ± 0.075	0.686 ± 0.034	0.578 ± 0.028
camera_height	acc.	0.333	1.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	1.000 ± 0.000	1.000 ± 0.000
background_color	acc.	0.333	$\mathbf{0.999 \pm 0.001}$	0.997 ± 0.002	0.735 ± 0.333	0.668 ± 0.250
horizontal_axis	R^{2+}	0	$\mathbf{0.209 \pm 0.065}$	0.097 ± 0.042	0.119 ± 0.063	0.102 ± 0.045
vertical_axis	R^{2+}	0	$\mathbf{0.563 \pm 0.074}$	0.425 ± 0.068	0.424 ± 0.180	0.251 ± 0.049
unweighted mean	—	—	0.814 ± 0.018	0.749 ± 0.008	0.683 ± 0.024	0.562 ± 0.058

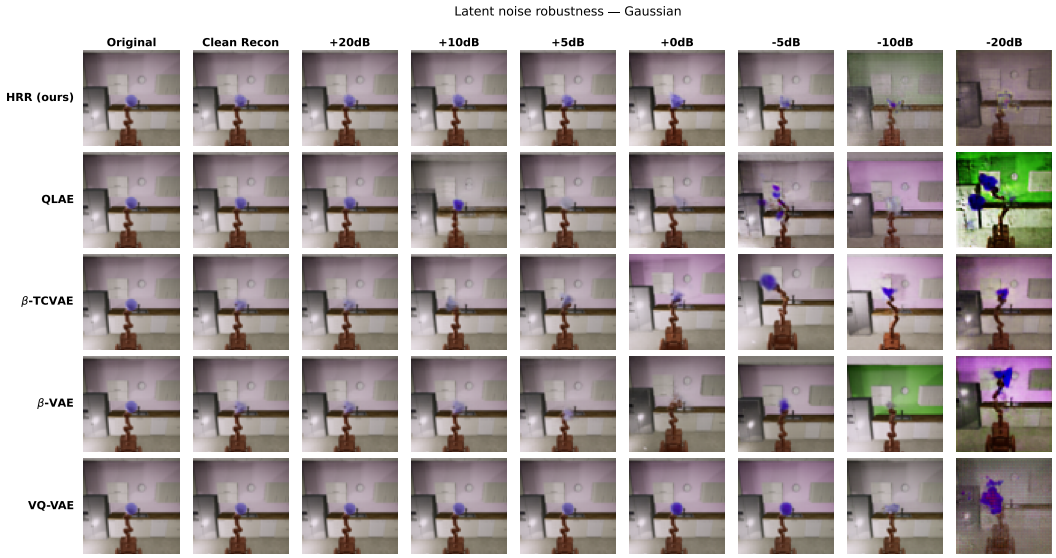


Figure 9: Each row is a visualization of how reconstruction quality degrades as noise intensity increases for each model. For each model, the seed with the highest InfoM score was used for evaluation.



Figure 10: Each row is a visualization of how reconstruction quality degrades as noise intensity increases for each model. For each model, the seed with the highest InfoM score was used for evaluation.

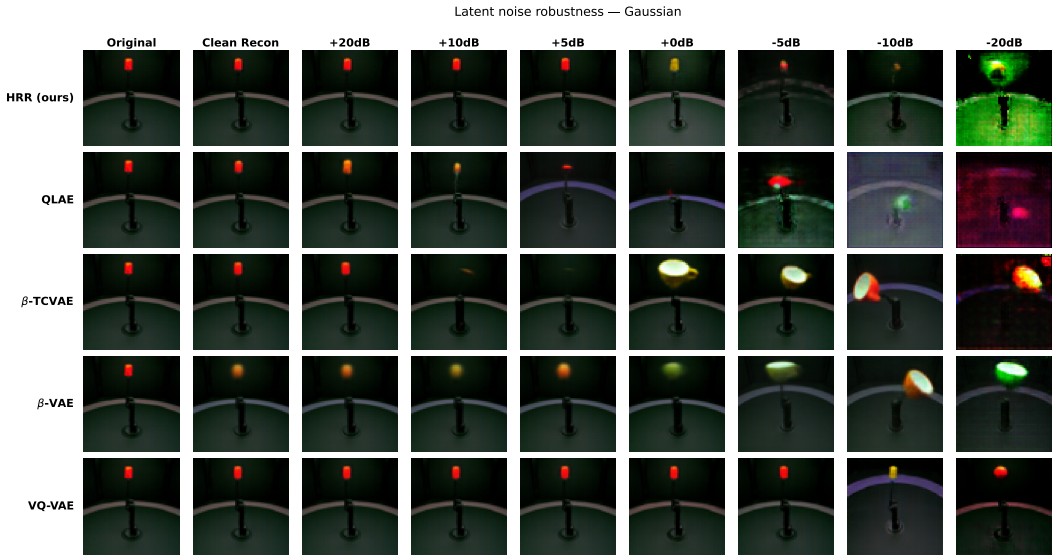


Figure 11: Each row is a visualization of how reconstruction quality degrades as noise intensity increases for each model. For each model, the seed with the highest InfoM score was used for evaluation.

L ADDITIONAL LATENT COMPONENT SWAPS

We include further results for the latent component swap experiment for the remaining datasets. Figs. 12, 13, 14 show the results on Isaac3D, Falcor3D, and MPI3D-C, respectively.

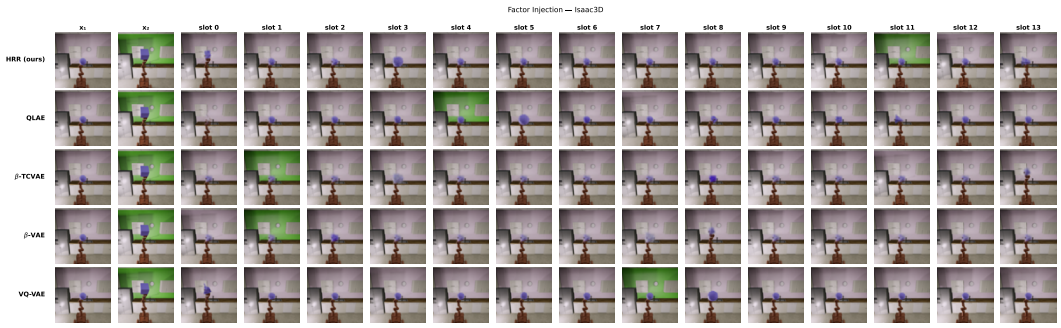


Figure 12: Latent component swaps performance for highest performing model relative to InfoM score. The slot 0 column represents taking the first latent unit from x_2, using it for the first latent unit for x_1, and decoding the result. Slot 1 does the same, but for the second latent unit, and so on. Each slot is replaced with its original value after visualization, so we expect each decoded result to manipulate either one, or no, aspects of the image.



Figure 13: Latent component swaps performance for highest performing model relative to InfoM score. The slot 0 column represents taking the first latent unit from x_2, using it for the first latent unit for x_1, and decoding the result. Slot 1 does the same, but for the second latent unit, and so on. Each slot is replaced with its original value after visualization, so we expect each decoded result to manipulate either one, or no, aspects of the image.

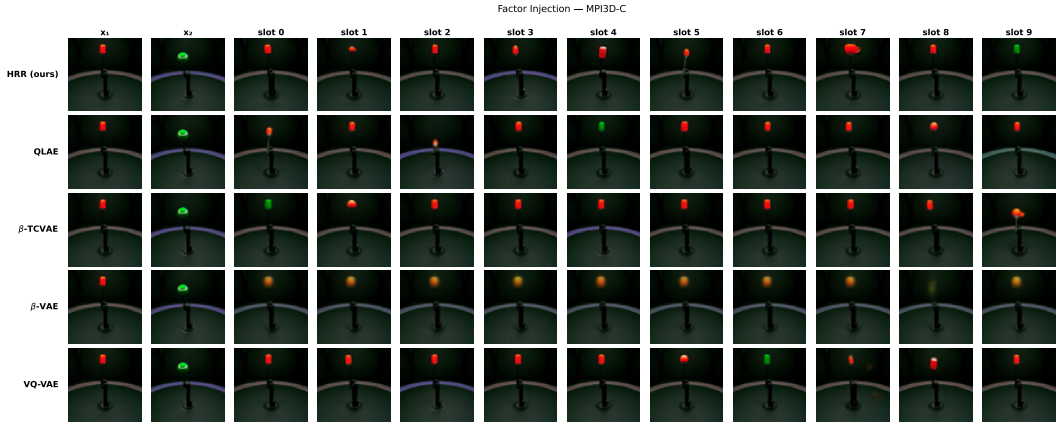


Figure 14: Latent component swaps performance for highest performing model relative to InfoM score. The slot 0 column represents taking the first latent unit from x_2, using it for the first latent unit for x_1, and decoding the result. Slot 1 does the same, but for the second latent unit, and so on. Each slot is replaced with its original value after visualization, so we expect each decoded result to manipulate either one, or no, aspects of the image.

M LATENT INTERPOLATIONS

A natural extension of component swapping is to swap slots cumulatively rather than individually. Starting from a source image's latent $\mathbf{z}_1$, we progressively replace slots with the corresponding slots from a target image's latent $\mathbf{z}_2$. At step i, slots 1 through i have been swapped from $\mathbf{z}_2$. For a disentangled representation, the decoded sequence should walk from the source image to the target one factor at a time, with each step changing only the most recently swapped factor. Figure 15 visualizes this for the HRR-based model and all the baselines, evaluated on the same image. Cumulative swapping is a more stringent test of slot independence than the single-component swap experiment of Section 4.4. In this scenario, errors from earlier swaps could distort later ones if slots were entangled, but each step in the HRR-based model's sequence changes one factor cleanly, providing qualitative evidence consistent with Proposition 2. We note that this is the natural notion of interpolation for a quantized symbolic latent, since smooth blending of codebook entries is not meaningful and a discrete walk through factor configurations is what compositional structure should support.

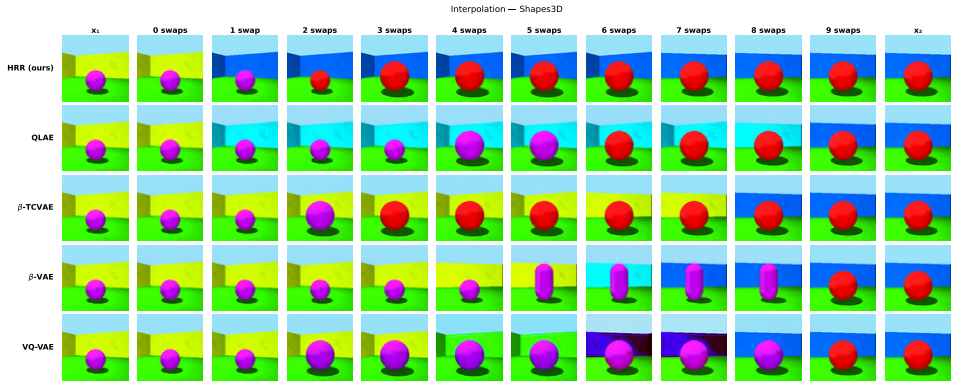


Figure 15: Latent interpolation swaps components progressively until the representation is fully transformed into that of the second image. We see that in the interpolation, the HRR-based model does not introduce any extraneous generative factors that were not present in the original two images. For instance, no decoded results for the HRR introduce a light blue wall, as the QLAE and β-VAE do.

We include further results for the latent interpolation experiment for the remaining datasets. Figs. 16, 17, 18 show the results on Isaac3D, Falcor3D, and MPI3D-C, respectively.

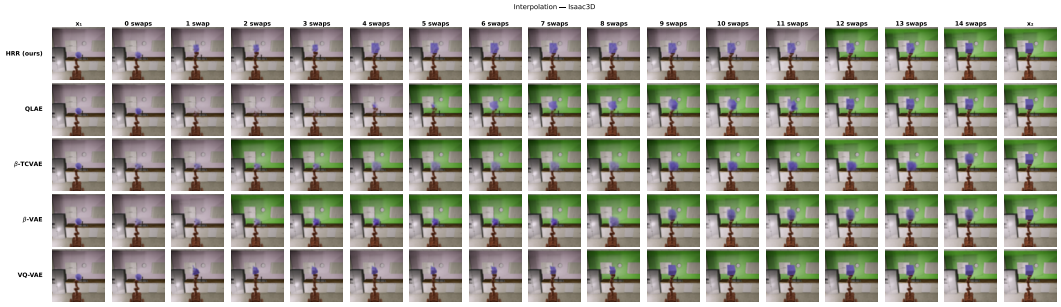


Figure 16: Latent interpolation swaps components progressively until the representation is fully transformed into that of the second image.

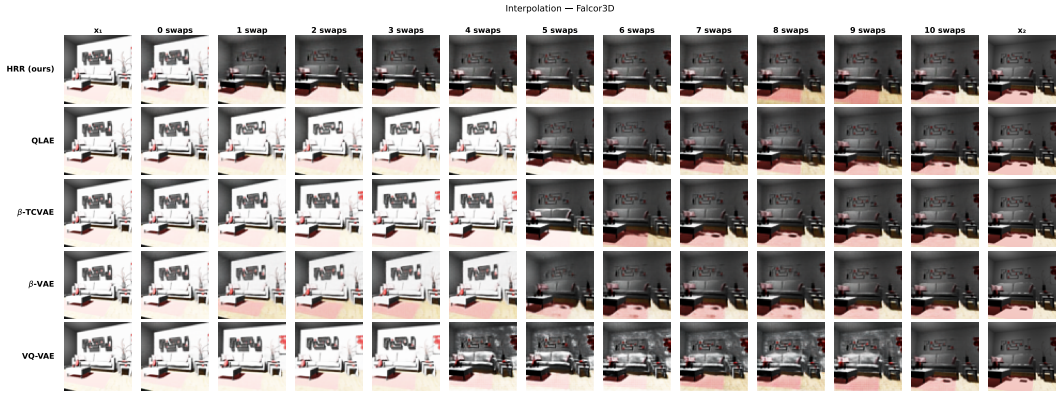


Figure 17: Latent interpolation swaps components progressively until the representation is fully transformed into that of the second image.

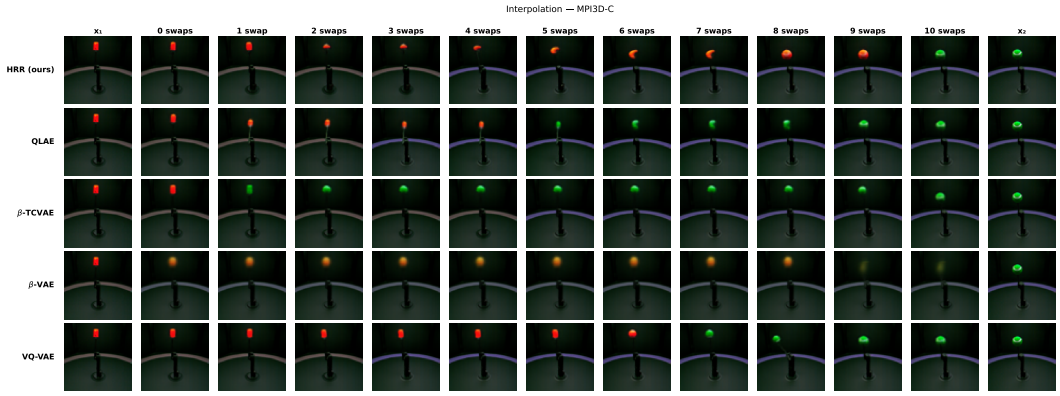


Figure 18: Latent interpolation swaps components progressively until the representation is fully transformed into that of the second image.