Quantifying and Learning Linear Symmetry-Based Disentanglement

Loek Tonnaer [*1 2] **Luis A. Pérez Rey** [*1 2 3] **Vlado Menkovski** [1 2] **Mike Holenderski** [1 2] **Jacobus W. Portegies** [1 2]

Abstract

The definition of Linear Symmetry-Based Disentanglement (LSBD) formalizes the notion of linearly disentangled representations, but there is currently no metric to quantify LSBD. Such a metric is crucial to evaluate LSBD methods and to compare to previous understandings of disentanglement. We propose $\mathcal{D}_{\text{LSBD}}$, a mathematically sound metric to quantify LSBD, and provide a practical implementation for $\text{SO}(2)$ groups. Furthermore, from this metric we derive LSBD-VAE, a semi-supervised method to learn LSBD representations. We demonstrate[1] the utility of our metric by showing that (1) common VAE-based disentanglement methods don't learn LSBD representations, (2) LSBD-VAE, as well as other recent methods, *can* learn LSBD representations needing only limited supervision on transformations, and (3) various desirable properties expressed by existing disentanglement metrics are also achieved by LSBD representations.

1. Introduction

Learning low-dimensional representations that disentangle the underlying factors of variation in data is considered an important step towards interpretable machine learning with good generalization. To address the fact that there is no consensus on what disentanglement entails and how to formalize it, Higgins et al. (2018) propose a formal definition for Linear Symmetry-Based Disentanglement, or LSBD, arguing that underlying real-world symmetries give exploitable structure to data (see Sect. 3).

[*]Equal contribution [1]Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands [2]Eindhoven Artificial Intelligence Systems Institute (EAISI), Eindhoven, the Netherlands [3]Prosus, Amsterdam, The Netherlands. Correspondence to: Loek Tonnaer <l.m.a.tonnaer@tue.nl>, Luis A. Pérez Rey <l.a.perez.rey@tue.nl>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[1]Code available at https://github.com/luis-armando-perez-rey/lsbd-vae

LSBD emphasizes that the variability in data observations is often due to some transformations, and that good data representations should reflect these transformations. A typical setting is that of an agent interacting with its environment. An action of the agent will transform some aspect of the environment and its observation thereof, but keeps all other aspects invariant. It is often easy and cheap to register the actions that an agent performs and how they transform the observed environment, which can provide useful information for learning disentangled representations.

However, there is currently no general metric to quantify LSBD. Such a metric is crucial to properly evaluate methods aiming to learn LSBD representations and to relate LSBD to previous definitions of disentanglement. Although previous works have evaluated LSBD by measuring performance on downstream tasks (Caselles-Dupré et al., 2019) or by measuring specific traits related to LSBD (Painter et al., 2020; Quessard et al., 2020), none of these evaluation methods directly quantify LSBD according to its formal definition.

We propose $\mathcal{D}_{\text{LSBD}}$, a well-formalized and generally applicable metric that quantifies the level of LSBD in learned data representations (Sect. 4). We show an intuitive justification of this metric, as well as its theoretical derivation. We also provide a practical implementation to compute $\mathcal{D}_{\text{LSBD}}$ for common $\text{SO}(2)$ symmetry groups. Furthermore, we show that our metric formulation can be used to derive a semi-supervised method to learn LSBD representations, which we call LSBD-VAE (Sect. 5). To make LSBD-VAE more widely applicable, we also demonstrate how to disentangle symmetric properties from other non-symmetric properties, and how to quantify this disentanglement with $\mathcal{D}_{\text{LSBD}}$.

We show the utility of $\mathcal{D}_{\text{LSBD}}$ by quantifying LSBD in a number of settings, for a variety of datasets with underlying $\text{SO}(2)$ symmetries and other non-symmetric properties (Sect. 6 & 7). First, we evaluate common VAE-based disentanglement methods and show that most don't learn LSBD representations. Second, we evaluate LSBD-VAE and other recent methods that specifically target LSBD, showing that they *can* obtain much better $\mathcal{D}_{\text{LSBD}}$ scores while needing only limited supervision on transformations. Third, we compare $\mathcal{D}_{\text{LSBD}}$ with existing disentanglement metrics, showing that various desirable properties expressed with these metrics are also achieved by LSBD representations.

2. Related Work

Plenty of works have focused on learning and quantifying disentangled representations recently, but research has shown that there is little consensus about the exact definition of disentanglement and methods often do not achieve it as well as they proclaim (Locatello et al., 2019). To introduce some much-needed formalization, Higgins et al. (2018) proposed to define disentanglement with respect to symmetry transformations acting on the data. They used group theory to provide two formal definitions, which we refer to as (Linear) Symmetry-Based Disentanglement, or (L)SBD. In this paper we focus only on LSBD, not SBD.

Several methods have been proposed to learn LSBD representations (Caselles-Dupré et al., 2019; Painter et al., 2020; Quessard et al., 2020). These methods also learn to represent the transformations acting on the input data, assuming various levels of supervision on these transformations. Other methods have previously focused on capturing transformations of the data outside the context of disentanglement as well (Cohen & Welling, 2015; Sosnovik et al., 2019; Worrall et al., 2017).

Although some of these works do propose metrics that measure some aspect of LSBD, none of them provide a general metric that directly quantifies LSBD according to its formal definition and for any data representation. Painter et al. (2020) mention two metrics: *Independence Score* measures whether the actions of the subgroups have effects on independent vector spaces, *Factor Leakage* only measures the number of dimensions in which the subgroup actions are encoded, which is not a property required by LSBD. Neither are general quantifications of LSBD. Additionally, Quessard et al. (2020) also propose a "metric", but this is in fact a loss component particular to their group representation parameterization and cannot be used as a general metric for LSBD.

3. Linear Symmetry-Based Disentanglement

Higgins et al. (2018) provide a formal definition of linear disentanglement that connects symmetry transformations affecting the real world (from which data is observed) to the internal representations of a model. The definition is grounded in concepts from *group theory*, we provide a more detailed description of these concepts in Appendix A.

The definition[2] considers a group G of symmetry transformations acting on the *data space* X through the group action $\cdot : G \times X \to X$. In particular, G can be decomposed as the direct product of K groups $G = G_1 \times \ldots \times G_K$. A model's internal representation of data is modeled with the *encoding* function $h : X \to Z$ that maps data to the *embedding space* Z. The definition for Linearly Symmetry-Based Disentangled (LSBD) representations then formalizes the requirement that a model's encoding h should reflect and disentangle the transformation properties of the data, and that the transformation properties of the model's encoding should be linear. The exact definition is as follows:

Definition: Linear Symmetry-Based Disentanglement (LSBD) A model's encoding map $h : X \to Z$, where Z is a vector space, is LSBD with respect to the group decomposition $G = G_1 \times \ldots \times G_K$ if

1. there is a decomposition of the embedding space $Z = Z_1 \oplus \ldots \oplus Z_K$ into K vector subspaces,
2. there are group representations for each subgroup in the corresponding vector subspace $\rho_k : G_k \to \mathrm{GL}(Z_k)$, $k \in \{1, \ldots, K\}$
3. the group representation $\rho : G \to \mathrm{GL}(Z)$ acts on Z as

$$\rho(g) \cdot z = (\rho_1(g_1) \cdot z_1, \ldots, \rho_K(g_K) \cdot z_K), \quad (1)$$

   for $g = (g_1, \ldots, g_K) \in G$ and $z = (z_1, \ldots, z_K) \in Z$ with $g_k \in G_k$ and $z_k \in Z_k$.
4. the map h is *equivariant* with respect to the actions of G on X and Z, i.e. , for all $x \in X$ and $g \in G$ it holds that $h(g \cdot x) = \rho(g) \cdot h(x)$.

Furthermore, we say that a group representation ρ is *linearly disentangled* with respect to the group decomposition $G = G_1 \times \ldots \times G_K$ if it satisfies criteria 1 to 3 from the LSBD definition above.

4. Quantifying LSBD: $\mathcal{D}_{\mathrm{LSBD}}$

4.1. Intuition: Measuring Equivariance with Dispersion

To motivate our metric, let's first assume a setting in which a suitable *linearly disentangled* group representation ρ is known. Let's further assume that the dataset of observations can be expressed with respect to G acting on some base point $x_0 \in X$, i.e. $\{x_n\}_{n=1}^N = \{g_n \cdot x_0\}_{n=1}^N$. Formally, this assumes that the action of G on X is *regular*. In this case, we can use the inverse group elements g_n^{-1} to transform each data point toward the base point x_0, i.e.

$$x_0 = g_1^{-1} \cdot x_1 = \ldots = g_N^{-1} \cdot x_N. \quad (2)$$

Since ρ is *linearly disentangled*, we only need to measure the *equivariance* of the encoding map h to quantify LSBD. Equivariance is achieved when $h(g \cdot x) = \rho(g) \cdot h(x)$, for all

[2] The original definition actually considers an additional set of world states W, but our definition is more practical and can be shown to be the same under mild conditions, see Appendix B.

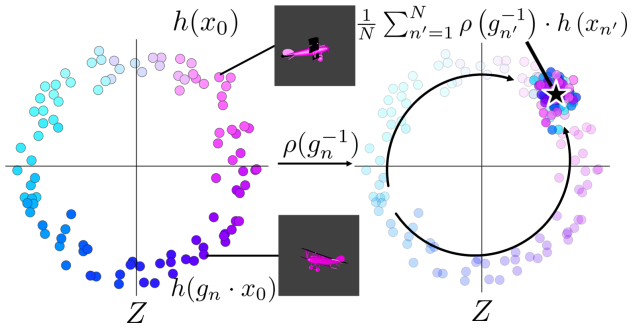


Figure 1: A dataset of images from a rotating object expressed in terms of the group $G = \mathrm{SO}(2)$ acting on a base image x_0. It is possible to quantify the level of LSBD of an encoding map h by measuring its equivariance with respect to a group representation ρ. Since all data has been generated from x_0, equivariance can be measured as the dispersion of the points $\{\rho(g_n^{-1}) \cdot h(x_n)\}_{n=1}^N$.

$g \in G, x \in X$. Given the dataset described above, we can check this property for $x \in \{x_n\}_{n=1}^N$ and $g \in \{g_n\}_{n=1}^N$.[3] In particular, from Equation (2) we can see that we have equivariance if

$$h(x_0) = \rho(g_1^{-1}) \cdot h(x_1) = \ldots = \rho(g_N^{-1}) \cdot h(x_N). \quad (3)$$

This not only characterizes perfect equivariance, but also allows for an efficient way to quantify how close we are to true equivariance, by measuring the *dispersion* of the points $\{\rho(g_n^{-1}) \cdot h(x_n)\}_{n=1}^N$.[4] Given a suitable norm $\|\cdot\|_Z$ in Z, we can thus quantify LSBD in this setting as

$$\frac{1}{N}\sum_{n=1}^{N}\left\|\rho(g_n^{-1}) \cdot h(x_n) - M^*\right\|_Z^2, \quad \text{with } M^* = \frac{1}{N}\sum_{n'=1}^{N}\rho(g_{n'}^{-1}) \cdot h(x_{n'}), \quad (4)$$

i.e. we compute the mean M^* of $\{\rho(g_n^{-1}) \cdot h(x_n)\}_{n=1}^N$ and use the average squared distance to this mean for points in $\{\rho(g_n^{-1}) \cdot h(x_n)\}_{n=1}^N$ as our LSBD metric, see Fig. 1.

However, this formulation requires knowing the right *linearly disentangled* group representation and a suitable norm in Z. Moreover, it implicitly assumes a uniform probability measure over the group elements $\{g_n\}_{n=1}^N$. In the next section we formulate our metric for a more general setting.

[3] Note that $\{g_n\}_{n=1}^N$ can be used to describe all known group transformations between elements in the dataset by means of composition and inverses, since $x_i = g_i \cdot (g_j^{-1} \cdot x_j)$. Thus it suffices to check equivariance for these N group transformations.

[4] Note that we do not actually need to know x_0 nor $h(x_0)$.

4.2. $\mathcal{D}_{\mathrm{LSBD}}$: A Metric for LSBD

Generalizing the ideas from the previous section with concepts from *measure theory*, we propose a metric to measure the level of LSBD of any encoding $h : X \to Z$ given a data probability measure μ on X, provided that μ can be written as the pushforward $G_X(\cdot, x_0)_\# \nu$ of some probability measure ν on G by the function $G_X(\cdot, x_0)$ for some base point x_0. More formally,

$$\begin{aligned}\mu(A) &= G_X(\cdot, x_0)_\# \nu(A) \\ &= \nu\left(\{g \in G \mid G_X(g, x_0) \in A\}\right),\end{aligned} \quad (5)$$

for Borel subsets $A \subset X$. Note that this is only possible if the action G_X is *transitive*.

For example, the situation of a dataset with N datapoints $\{x_n\}_{n=1}^N = \{g_n \cdot x_0\}_{n=1}^N$ corresponds to the case in which ν and μ are empirical measures on the group G and data space X, respectively:

$$\nu := \frac{1}{N}\sum_{i=1}^{N}\delta_{g_i}, \qquad \mu := \frac{1}{N}\sum_{i=1}^{N}\delta_{x_i}. \quad (6)$$

We define the metric $\mathcal{D}_{\mathrm{LSBD}}$ for an encoding h and a measure μ as

$$\mathcal{D}_{\mathrm{LSBD}} := \inf_{\rho \in \mathcal{P}(G,Z)} \int_G \left\|\rho(g)^{-1} \cdot h(g \cdot x_0) - M_{\rho,h,x_0}\right\|_{\rho,h,\mu}^2 d\nu(g), \quad \text{with } M_{\rho,h,x_0} = \int_G \rho(g')^{-1} \cdot h(g' \cdot x_0) d\nu(g'), \quad (7)$$

where the norm $\|\cdot\|_{\rho,h,\mu}$ is a Hilbert-space norm depending on the representation ρ, the encoding map $h : X \to Z$, and the data measure μ. More details of this norm can be found in Appendix C. Moreover, $\mathcal{P}(G, Z)$ denotes the set of *linearly disentangled representations* of G in Z. Lower values of $\mathcal{D}_{\mathrm{LSBD}}$ indicate better disentanglement, zero being optimal.

4.3. Practical Computation of $\mathcal{D}_{\mathrm{LSBD}}$

There are two main challenges for computing the metric of Equation (7). First, to calculate the integrals in the formula, all possible datapoints that can be expressed as $g \cdot x_0$ with $g \in G = G_1 \times \cdots \times G_K$ must be available. Second, the infimum of the integrals over all possible linearly disentangled representations must be estimated. This requires finding the possible invariant subspaces $Z = Z_1 \oplus \cdots \oplus Z_K$ induced by the encoding h over which the group representations are disentangled.

We present a practical implementation of an upper bound to $\mathcal{D}_{\mathrm{LSBD}}$ for an encoding function h given a dataset $\mathcal{X}$

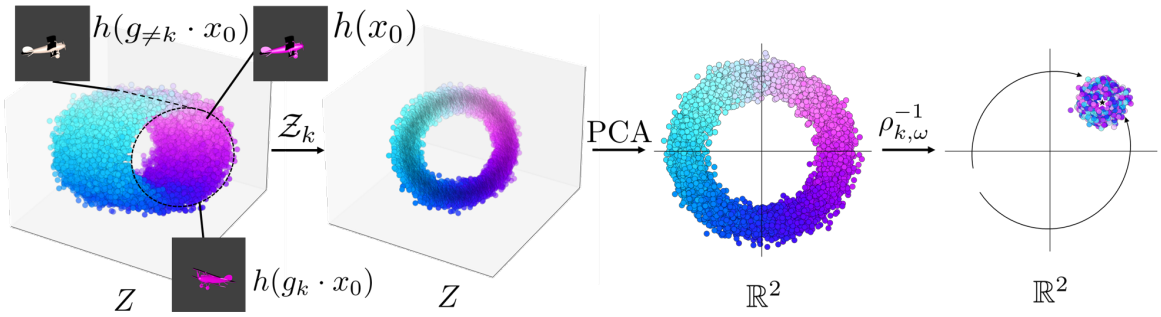


Figure 2: Consider a dataset modeled by a group decomposition $G = G_1 \times \cdots \times G_K$ acting on x_0 and embedded in a latent space Z via h. In this example the subgroup $G_k = \mathrm{SO}(2)$ models the rotations of an airplane. Other subgroups $G_{\neq k}$ could also be acting e.g. changes in airplane color. The first step to calculate the disentanglement of G_k is to construct a set of data embeddings $\mathcal{Z}_k \subseteq Z$ whose variability is due to G_k. These embeddings are then projected into a 2-dimensional space through PCA. For these projected embeddings we can describe the group representations in a simple parametric form $\rho_{k,w}$. For a given $\rho_{k,w}$ the equivariance of G_k is measured as the dispersion after applying the action of the inverse group representation $\rho_{k,w}^{-1}$.

generated by some known group transformations. This approximation of $\mathcal{D}_{\mathrm{LSBD}}$ is designed for a group decomposition $G = G_1 \times \cdots \times G_K$ where each $G_k = \mathrm{SO}(D_k)$ with $k \in \{1, \ldots, K\}$ the group of rotations in D_k dimensions. This implementation approximates the integrals of Equation (7) by using the empirical distribution of $\mathcal{X}$. The invariant subspaces of Z to the subgroup actions are found by applying a suitable change of basis. In the new basis, the disentangled group representations are expressed in a parametric form whose parameters are optimized to find the tightest bound to $\mathcal{D}_{\mathrm{LSBD}}$. See Fig. 2 for an intuitive description of the process.

Assume there is a dataset $\mathcal{X}$ that can be modeled in terms of the group decomposition $G = G_1 \times \cdots G_k$. For each G_k subgroup there is a set of known group elements $\mathcal{G}_k \subseteq G_k$ uniformly sampled such that the dataset is described in terms of all elements in $\mathcal{G} = \mathcal{G}_1 \times \cdots \times \mathcal{G}_K$ and a base point x_0 as $\mathcal{X} = \{(g_1, \ldots, g_K) \cdot x_0 | g_k \in \mathcal{G}_k,\ k \in \{1, \ldots, K\}\}$.

For each subgroup G_k we construct a set of encoded data $\mathcal{Z}_k \subseteq Z$ whose variability should only depend on the action of G_k. The set $\mathcal{Z}_k$ is given by $\mathcal{Z}_k = \{z_k(g_1, \ldots, g_K) | g_j \in \mathcal{G}_j\ , j \in \{1, \ldots, K\}\}$, in which

$$z_k(g_1, \ldots, g_K) = h((g_1, \ldots, g_K) \cdot x_0) - \frac{1}{|\mathcal{G}_k|} \sum_{g' \in \mathcal{G}_k} h((g_1, \ldots, g_{k-1}, g', g_{k+1}, \ldots, g_K) \cdot x_0). \tag{8}$$

Similar to Cohen & Welling (2014), we find a suitable change of basis that exposes the invariant subspace Z_k corresponding to the k-th subgroup G_k. The new basis is obtained from the eigenvectors resulting from applying Principal Component Analysis (PCA) to $\mathcal{Z}_k$. Each element in $\mathcal{Z}_k$ is projected into the first D_k eigenvectors. The new set is denoted as $\mathcal{Z}'_k \subseteq \mathbb{R}^{D_k}$ with elements $z'_k(g_1, \ldots, g_K) \subseteq \mathbb{R}^{D_k}$ that are the projected versions of $z_k(g_1, \ldots, g_K)$.

Quessard et al. (2020) describe how one could parameterize the subgroup representations of $SO(D_k)$ for arbitrary D_k but here we will focus on $G_k = SO(2)$. In this case, we can parameterize each subgroup representation in terms of a single integer parameter $\omega \in \mathbb{Z}$ as $\rho_{k,\omega}(g_k)$ corresponding to a 2×2 rotation matrix whose angle of rotation is ω multiplied by the known angle associated to the group element $g_k \in G_k = \mathrm{SO}(2)$. For this subgroup we can approximate the M_{ρ,h,x_0} in Equation (7) as $M_{k,\omega}$ given by

$$M_{k,\omega} = \frac{1}{|\mathcal{G}|} \sum_{(g_1, \ldots, g_K) \in \mathcal{G}} \rho_{k,\omega}(g_k^{-1}) \cdot z'(g_1, \ldots, g_K). \tag{9}$$

Similar to Equation (7) we would like to find the optimal $\rho_{k,\omega}$ that minimizes the integral over the group representations. We can define a parameter search space $\Omega \subseteq \mathbb{Z}$, e.g. $\Omega = [-10, 10]$ for finding the optimal $\omega \in \Omega$ that minimizes the dispersion, this is expressed in the following equation

$$\mathcal{D}_{\mathrm{LSBD}}^{(k)} = \min_{\omega \in \Omega} \frac{1}{|\mathcal{G}|} \sum_{(g_1, \ldots, g_K) \in \mathcal{G}} \|\rho_{k,\omega}(g_k^{-1}) \cdot z'(g_1, .., g_K) - M_{k,\omega}\|^2. \tag{10}$$

Each $\mathcal{D}_{\mathrm{LSBD}}^{(k)}$ measures the degree of equivariance of the projected embeddings for each k-th subgroup corresponding to the best fitting group representation. The upper bound to the metric is finally obtained by averaging across all subgroups: $\mathcal{D}_{\mathrm{LSBD}} \leq \frac{1}{K} \sum_{k=1}^{K} \mathcal{D}_{\mathrm{LSBD}}^{(k)}$.

Our practical implementation of $\mathcal{D}_{\text{LSBD}}$ is for $\text{SO}(2)$ subgroups, however the procedure can in principle be extended to other subgroups as well. A practical implementation of the metric requires (i) identifying the subspaces invariant to a subgroup and (ii) identifying a parametric representation of the subgroup that can be fitted to the subspace data representations. In cases where the exact form of the subgroup is unknown, an option is to use the method by Pfau et al. (2020) to factorize the submanifolds associated with different generative factors.

5. Learning LSBD: LSBD-VAE

In this section we present LSBD-VAE, a semi-supervised VAE-based method to learn LSBD representations. The main idea is to train an unsupervised Variational Autoencoder (VAE) (Kingma & Welling, 2014; Rezende et al., 2014) with a suitable latent space topology, and use our metric as an additional loss term for batches of transformation-labeled data.

Assumptions LSBD-VAE requires some knowledge about the group structure G that is to be disentangled. Concretely, the group and its decomposition $G = G_1 \times \ldots \times G_K$ should be known, as well as a suitable *linearly disentangled* group representation $\rho : G \to \text{GL}(Z)$ and a latent space $Z = Z_1 \oplus \ldots \oplus Z_K$. Moreover, we assume there exists an embedded submanifold $Z_G \subseteq Z$ such that the action of G on Z restricted to Z_G is *regular*, and Z_G is invariant under the action. Only Z_G will then be used as the codomain for the encoding map, $h : X \to Z_G$.

We demonstrate the assumptions above for the common group structure $G = \text{SO}(2) \times \text{SO}(2)$. For the group representation $\rho = \rho_1 \oplus \rho_2$, with $Z = \mathbb{R}^2 \oplus \mathbb{R}^2$, we can use rotation matrices in $\mathbb{R}^2$ for ρ_1 and ρ_2. We can then use 1-spheres $S^1 = \{z \in \mathbb{R}^2 : \|z\| = 1\}$ for the embedded submanifold: $Z_G = S^1 \times S^1$. In this case, the action of G on Z restricted to Z_G is indeed *regular*, and Z_G is invariant under the action.

Requiring the group structure G to be known is a relatively strong assumption, which limits the practical applicability of our method. However, a group structure can often be given as expert knowledge, like the presence of cyclic factors such as rotation, or in situations where transformations between observed data can easily be acquired such as in reinforcement learning.

Unsupervised Learning on Latent Manifold To learn encodings only on the latent manifold Z_G, we use a Diffusion Variational Autoencoder (ΔVAE) (Perez Rey et al., 2020). ΔVAEs can use any closed Riemannian manifold embedded in a Euclidean space as a latent space (or latent manifold), provided that a certain *projection function* from the Euclidean embedding space into the latent manifold is known and the *scalar curvature* of the manifold is available. The ΔVAE uses a parametric family of posterior approximates obtained from a diffusion process over the latent manifold. To estimate the intractable terms of the negative ELBO, the reparameterization trick is implemented via a random walk.

In the case of S^1 as a latent (sub)manifold, we consider $\mathbb{R}^2$ as the Euclidean embedding space, and the projection function[5] $\Pi : \mathbb{R}^2 \to S^1$ normalizes points in the embedding space: $\Pi(z) = z/|z|$. The scalar curvature of S^1 is 0.

Semi-Supervised Learning with Transformation Labels Caselles-Dupré et al. (2019) proved that LSBD representations cannot be inferred from a training set of unlabeled observations, but that access to the transformations between data points is needed. They therefore use a training set of observation pairs with a given transformation between them.

However, we posit that only a limited amount of supervision is sufficient. Since obtaining supervision on transformations is typically more expensive than obtaining unsupervised observations, it is desirable to limit the amount of supervision needed.

Therefore, we augment the unsupervised ΔVAE with a supervised method that makes use of transformation-labeled batches, i.e. batches $\{x_m\}_{m=1}^M$ such that $x_m = g_m \cdot x_1$ for $m = 2, \ldots, M$, where the transformations g_m (and thus their group representations $\rho(g_m)$) are known and are referred to as *transformation labels*. The simplified version of the metric from Equation (4) can then be used for each batch as an additional loss term (with $x_0 = x_1$), as it is differentiable under the assumptions described above (using the Euclidean norm).

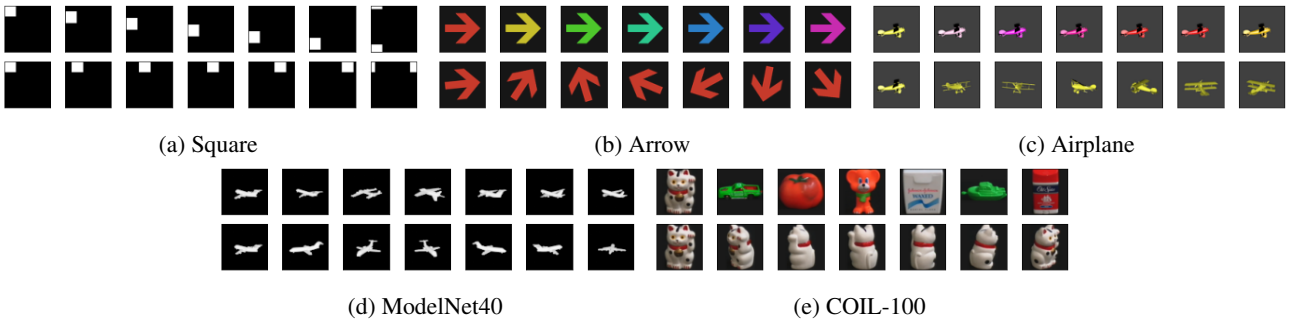
We make a small adjustment to Equation (4) for the purpose of our method, since the mean computed there does not typically lie on the latent manifold Z_G. Thus, we use the projection Π from the ΔVAE to project the mean onto Z_G. Writing the encodings as $z_m := h(x_m)$, the additional loss term for a transformation-labeled batch $\{x_m\}_{m=1}^M$ becomes

$$\mathcal{L}_{LSBD} = \frac{1}{M} \sum_{m=1}^{M} \left\| \rho(g_m^{-1}) \cdot z_m - \Pi \left(\frac{1}{M} \sum_{m=1}^{M} \rho(g_m^{-1}) \cdot z_m \right) \right\|^2, \tag{11}$$

where $g_1 = e$, the group identity.

Moreover, instead of feeding the encodings z_m to the decoder, we use $\rho(g_m) \cdot \overline{z}$, where

[5] This projection function is not defined for $z = \mathbf{0}$, but this value does not occur in practice.



(a) Square (b) Arrow (c) Airplane

(d) ModelNet40 (e) COIL-100

Figure 3: Example images from each of the datasets used. Each row shows different examples from a single factor changing.

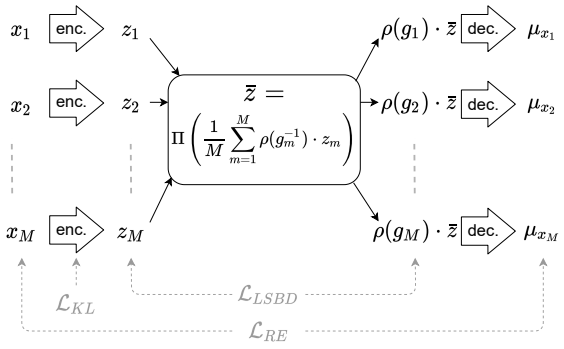


Figure 4: Overview of the supervised part of LSBD-VAE.

$\overline{z} = \Pi\left(\frac{1}{M}\sum_{m=1}^{M}\rho(g_m^{-1})\cdot z_m\right)$. This encourages the decoder to follow the required group structure. This only affects the reconstruction loss component of the ΔVAE.

Fig. 4 illustrates the supervised part of our method for a transformation-labeled batch $\{x_m\}_{m=1}^{M}$. The loss function is the regular ELBO (but with adjusted decoder input as described above) as used in ΔVAE plus an additional term $\gamma \cdot \mathcal{L}_{LSBD}$, where γ is a weight hyperparameter to control the influence of the supervised loss component. By alternating unsupervised and supervised training (using the same encoder and decoder), we have a method that makes use of both unlabeled and transformation-labeled observations.

6. Experimental Setup

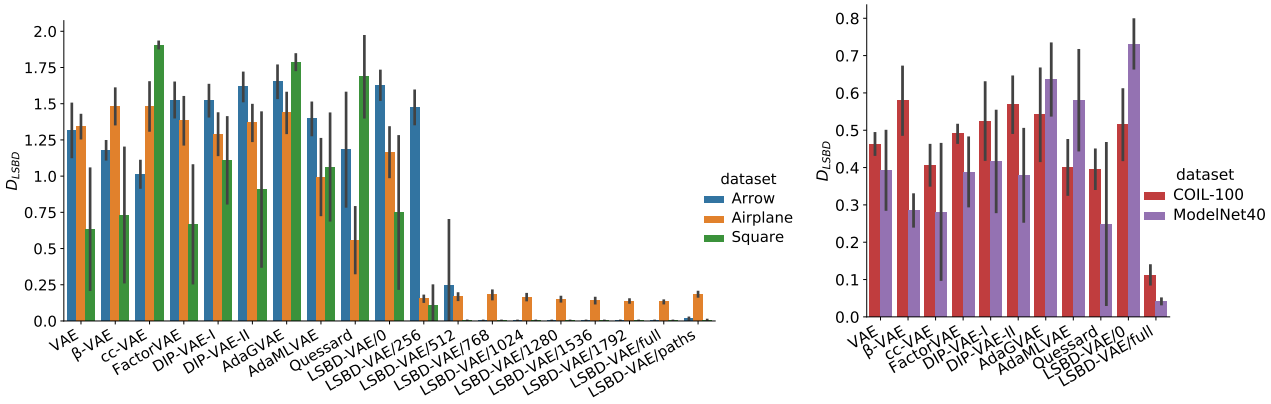
Data We evaluate the disentanglement of several models on three different image datasets (Square, Arrow, and Airplane) with a known group decomposition $G = \mathrm{SO}(2) \times \mathrm{SO}(2)$ describing the underlying transformations. For each subgroup a fixed number of $|\mathcal{G}_k| = 64$ with $k \in \{1, 2\}$ transformations is selected. The datasets exemplify different group actions of $\mathrm{SO}(2)$: periodic translations, in-plane rotations, out-of-plane rotations, and periodic hue-shifts.

In real settings, not all variability in the data can be modelled by the actions of a group. Therefore, we also evaluate the same models on two datasets ModelNet40 (Wu et al., 2014) and COIL-100 (Nene et al., 1996) that consist of images from various objects (i.e. non-symmetric variation) under known out-of-plane rotations ($\mathrm{SO}(2)$ symmetries). In many settings it is easy to obtain labels for such rotations, e.g. when the camera or object angle is controlled by an agent. See Fig. 3 for examples of the datasets. For more details, see Appendix E.

Note that we do not evaluate our LSBD-VAE method and $\mathcal{D}_{\mathrm{LSBD}}$ metric on traditional disentanglement datasets as evaluated by Locatello et al. (2019), since these datasets lack a clear underlying group structure. However, our results on the ModelNet40 and COIL-100 datasets show that our method can disentangle properties with a group structure from properties without such a structure.

LSBD-VAE with Semi-Supervised Labelled Pairs For the Square, Arrow, and Airplane datasets we test LSBD-VAE with transformation-labeled batches of size $M = 2$. More specifically, for each experiment we randomly select L disjoint pairs of data points, and label the transformation between the data points in each pair. We vary the number of labeled pairs L from 0 (corresponding to a ΔVAE) to $N/2$ (in which case each data point is involved in exactly one labeled pair). We set the weight γ of the supervised loss component to $\gamma = 100$ for all experiments. We choose $M = 2$ for our experiments since it is the most limited setting for LSBD-VAE. Higher values of M would provide stronger supervision, so successful results with $M = 2$ imply that good results can also be achieved for higher values of M (but not necessarily vice versa).

For the COIL-100 and ModelNet40 datasets, we train LSBD-VAE on batches containing images of one particular object from all different angles (72 and 64 for COIL-100 and ModelNet40, respectively). Each batch is labelled with transformations $(g_1, e), \ldots, (g_M, e)$, where g_m represent rotations, and the unit transformation e indicates that the object is unchanged. To represent the rotations we use a S^1 latent



(a) Datasets with $\mathrm{SO}(2) \times \mathrm{SO}(2)$ symmetries

(b) Datasets with $\mathrm{SO}(2)$ and non-symmetric variation

Figure 5: $\mathcal{D}_{\mathrm{LSBD}}$ scores for all methods on all datasets

space as in ΔVAE, whereas for the object identity we use a 5-dimensional Euclidean space with standard Gaussian prior as in regular VAEs. LSBD is measured as the disentanglement of rotations in the latent space. For these experiments we used $\gamma = 1$.

LSBD-VAE with Paths of Consecutive Observations It is often cheap to obtain transformation labels in settings where we can apply simple transformations and observe its effect, such as an agent navigating its environment. By registering actions (e.g. rotate left over a given angle) and the resulting observations, we can construct a path of consecutive views with known in-between transformations. We can then use these paths to train a LSBD-VAE.

For the datasets with $G = G_1 \times G_2 = \mathrm{SO}(2) \times \mathrm{SO}(2)$ (Square, Arrow, Airplane), we generate random paths by consecutively applying one randomly chosen transformation from $\{g_1, g_1^{-1}, g_2, g_2^{-1}\}$ where $g_k \in G_k$ for $k \in \{1, 2\}$, starting from randomly chosen observations. In our experiments, we generate 50 paths of length 100, and g_k corresponds to an SO(2) transformation corresponding to an angle of $\frac{3}{64} 2\pi$ radians. Example paths can be found in Fig. 8 in the Appendix.

For the COIL-100 and ModelNet40 datasets there is only one group to disentangle. Therefore, similar random walks are not very meaningful here, and we do not evaluate them for these datasets.

Other Disentanglement Methods We furthermore test a number of known disentanglement methods for comparison, including traditional disentanglement methods as well as methods focusing on LSBD. In particular, we use `disentanglement_lib` (Locatello et al., 2019) to train a regular VAE (Kingma & Welling, 2014; Rezende et al., 2014), β-VAE (Higgins et al., 2017), CC-VAE (Burgess et al., 2018), FactorVAE (Kim & Mnih, 2018), and DIP-VAE-I/II (Kumar et al., 2018). We also include two weakly-supervised models, AdaGVAE and AdaMLVAE (Locatello et al., 2020), which are trained on pairs of data with few changing factors, to test whether this kind of supervision is helpful for LSBD. Furthermore we evaluate the method from Quessard et al. (2020) that focuses on LSBD. We also tested ForwardVAE (Caselles-Dupré et al., 2019), but show only limited results since we were not able to reproduce any reasonable results for our datasets.

Most of these methods have no notion of an underlying group structure, and thus do not give a fully fair comparison with our LSBD-VAE method. However, we emphasize that the main goal of our experiments is to investigate properties of disentangled representations from both the traditional and the LSBD perspective.

Disentanglement Metrics We use encodings from all methods to evaluate $\mathcal{D}_{\mathrm{LSBD}}$, as well as common traditional disentanglement metrics from `disentanglement_lib`: Beta (Higgins et al., 2017), Factor (Kim & Mnih, 2018), SAP (Kumar et al., 2018), DCI Disentanglement (Eastwood & Williams, 2018), Mutual Information Gap (MIG) (Chen et al., 2018), and Modularity (MOD) (Ridgeway & Mozer, 2018).

Further Details More information about the architectures, epochs and hyperparameters can be found in Appendix F. For the traditional disentanglement methods trained on Square, Arrow and Airplane datasets the latent spaces have 4 dimensions, since these are the minimum number of dimensions necessary to learn LSBD representations for an underlying $\mathrm{SO}(2) \times \mathrm{SO}(2)$ symmetry group, see (Higgins et al., 2018; Caselles-Dupré et al., 2019). For COIL-100 and ModelNet40 we use latent spaces with 7 dimensions for

a fair comparison with the LSBD-VAE method.

7. Results: Evaluating LSBD with $\mathcal{D}_{\text{LSBD}}$

We now highlight three key observations from our experimental results. In particular, we differentiate between the methods (VAE, β-VAE, CC-VAE, FACTOR, DIP-I, DIP-II) and metrics (BETA, FACTOR, SAP, DCI, MIG, MOD) that approach disentanglement in the *traditional* sense, and methods (ΔVAE, QUESSARD, LSBD-VAE) and metric ($\mathcal{D}_{\text{LSBD}}$) that focus specifically on LSBD. The full quantitative results can be found in Appendix H. Further qualitative results can be found in Appendix G.

7.1. Traditional Disentanglement Methods Don't Learn LSBD Representations

Fig. 5 summarizes the $\mathcal{D}_{\text{LSBD}}$ scores (lower is better) for all methods on all datasets. Bars show the mean scores over 10 runs for each method, the vertical lines represent standard deviations. LSBD-VAE/L indicates our method trained on L labelled pairs (LSBD-VAE/0 corresponds to the unsupervised ΔVAE), LSBD-VAE/full indicates our method where all images are involved in exactly one labelled pair. and LSBD-VAE/paths indicates our method trained with paths of consecutive observations. Note that LSBD-VAE obtained very good scores (near 0) on the Arrow and Square datasets, hence the missing bars.

None of the traditional disentanglement methods achieve good $\mathcal{D}_{\text{LSBD}}$ scores, even if they score well on other traditional disentanglement metrics. This implies that LSBD isn't achieved by traditional methods. Moreover, from the full results in Appendix H we see that the traditional methods on these datasets do not achieve good scores on all traditional metrics. In particular, SAP, DCI, and MIG scores are low. We believe this is a result of the cyclic nature of the symmetries underlying our datasets, further emphasizing the need for disentanglement methods that can capture such symmetries.

The SAP and MIG scores measure to what extent generative factors are disentangled into a single latent dimension. However, since the factors in our dataset are inherently cyclic due to their symmetry structure, they cannot be properly represented in a single latent dimension, as shown by Perez Rey et al. (2020). Instead, at least two dimensions are needed to continuously represent each cyclic factor in our data. A similar conclusion was made by Caselles-Dupré et al. (2019) and Painter et al. (2020).

DCI disentanglement measures whether a latent dimension captures at most one generative factor. This is accomplished by measuring the importance of each latent dimension in predicting the true generative factor using boosted trees. However, since the generative factors are cyclic, the performance of the boosted tree classifiers is far from optimal, thus providing more importance to several dimensions in predicting the generative factors and giving overall lower DCI scores.

7.2. LSBD-VAE and other LSBD Methods *Can* Learn LSBD Representations with Limited Supervision on Transformations

From Fig. 5 we observe that methods focusing specifically on LSBD can score higher on $\mathcal{D}_{\text{LSBD}}$, showing that they are indeed more suitable to learn LSBD representations. In particular, LSBD-VAE got very good $\mathcal{D}_{\text{LSBD}}$ scores for all datasets. Moreover, our experiments on the Arrow, Airplane, and Square datasets also show that only limited supervision suffices to obtain good $\mathcal{D}_{\text{LSBD}}$ scores with low variability, either with few transformation-labelled pairs or with paths of consecutive observations that are easy to obtain in agent-environment settings.

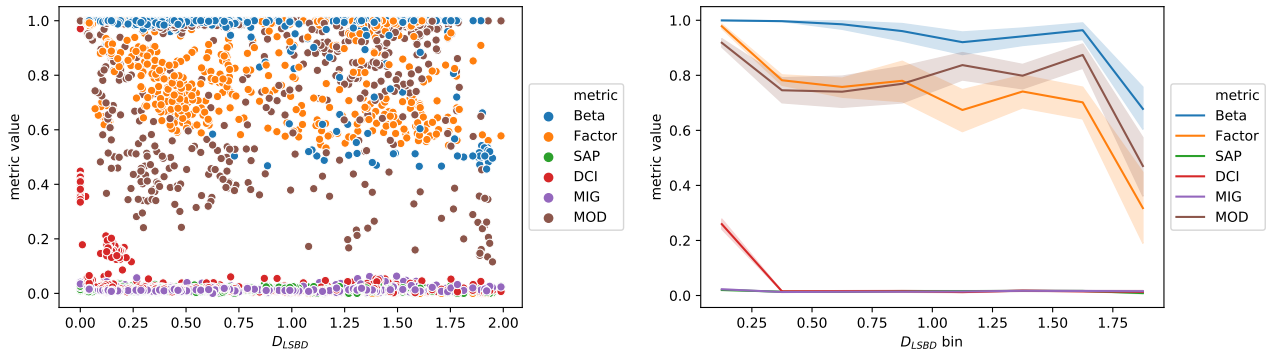
We only partially managed to reproduce the results from Quessard et al. (2020) on our datasets. Their method scored fairly well on the Airplane, ModelNet40, and COIL-100 datasets, but did not do well on the Square and Arrow dataset in our experiments.

Furthermore, we tested ForwardVAE by Caselles-Dupré et al. (2019), but we could not produce any reasonable results on our datasets. Therefore, we do not include scores for this method. We did manage to reproduce ForwardVAE's results on the Flatland dataset used in the original paper, for which we computed a mean $\mathcal{D}_{\text{LSBD}}$ score of 0.012 with standard deviation 0.001 over 10 runs, confirming that ForwardVAE indeed learns LSBD representations for Flatland.

7.3. LSBD Representations Also Satisfy Previous Disentanglement Notions

Our results also indicate that LSBD captures various desirable properties that are expressed by traditional disentanglement metrics. In Fig. 6 we compare $\mathcal{D}_{\text{LSBD}}$ scores with scores for previous disentanglement metrics. Note that for $\mathcal{D}_{\text{LSBD}}$ lower is better, whereas for all other metrics higher is better. As we noted before, good scores on traditional disentanglement metrics don't necessarily imply good $\mathcal{D}_{\text{LSBD}}$ scores. Conversely however, methods that score well on $\mathcal{D}_{\text{LSBD}}$ also score well on many traditional disentanglement metrics, often even outperforming the traditional methods. In particular, from the full results (see Appendix H) we see that LSBD-VAE matches or outperforms the traditional methods on the BETA, FACTOR and MOD metrics, and achieves much better scores for the DCI metric where traditional methods scored poorly.

The MIG and SAP scores are still low for methods focusing on LSBD. This is expected however, as explained earlier in



(a) Scatter plot

(b) Binned by $\mathcal{D}_{\text{LSBD}}$ value

Figure 6: Comparing $\mathcal{D}_{\text{LSBD}}$ to previous disentanglement metrics

Section 7.1. This was also observed by Painter et al. (2020) for different datasets.

8. Conclusion

We presented $\mathcal{D}_{\text{LSBD}}$, a metric to quantify Linear Symmetry-Based Disentanglement (LSBD) as defined by Higgins et al. (2018). We used this metric formulation to motivate LSBD-VAE, a semi-supervised method to learn LSBD representations given some expert knowledge on the underlying group symmetries that are to be disentangled.

We used $\mathcal{D}_{\text{LSBD}}$ to evaluate various disentanglement methods, both traditional methods and recent methods that specifically focus on LSBD, and showed that LSBD-VAE can learn LSBD representations where traditional methods fail to do so. We also compared $\mathcal{D}_{\text{LSBD}}$ to traditional disentanglement metrics, showing that LSBD captures many of the same desirable properties that are expressed by existing disentanglement methods. Conversely, we also showed that traditional disentanglement methods and metrics do not usually achieve or measure LSBD.

Challenges that remain are expanding and testing LSBD-VAE and $\mathcal{D}_{\text{LSBD}}$ on different group structures, towards more practical applications, as well as focusing on the utility of LSBD representations for downstream tasks.

9. Acknowledgements

This work has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737459 (project Productive4.0). This Joint Undertaking receives support from the European Union Horizon 2020 research and innovation program and Germany, Austria, France, Czech Republic, Netherlands, Belgium, Spain, Greece, Sweden, Italy, Ireland, Poland, Hungary, Portugal, Denmark, Finland, Luxembourg, Norway, Turkey.

This work has also received funding from the NWO-TTW Programme "Efficient Deep Learning" (EDL) P16-25.

References

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. Understanding disentangling in β-VAE. *arXiv preprint arXiv:1804.03599*, 2018.

Caselles-Dupré, H., Ortiz, M. G., and Filliat, D. Symmetry-based disentangled representation learning requires interaction with environments. In *Advances in Neural Information Processing Systems*, pp. 4606–4615, 2019.

Chen, T. Q., Li, X., Grosse, R. B., and Duvenaud, D. K. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pp. 2615–2625, 2018.

Cohen, T. and Welling, M. Learning the irreducible representations of commutative Lie groups. *31st International Conference on Machine Learning*, pp. 3757–3770, 2014.

Cohen, T. S. and Welling, M. Transformation properties of learned visual representations. In *3rd International Conference on Learning Representations*, 2015.

Community, B. O. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2020. URL http://www.blender.org.

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

Hall, B. C. *Lie Groups, Lie Algebras, and Representations*, volume 222 of *Graduate Texts in Mathematics*. Springer International Publishing, Cham, 2015. ISBN 978-3-319-13466-6. doi: 10.1007/978-3-319-13467-3. URL http://link.springer.com/10.1007/978-3-319-13467-3.

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

Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., and Lerchner, A. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

Kim, H. and Mnih, A. Disentangling by factorising. In *International Conference on Machine Learning*, pp. 2649–2658, 2018.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

Kumar, A., Sattigeri, P., and Balakrishnan, A. Variational inference of disentangled latent concepts from unlabeled observations. In *International Conference on Learning Representations*, 2018.

Locatello, F., Bauer, S., Lucic, M., Gelly, S., Schölkopf, B., and Bachem, O. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, 2019.

Locatello, F., Poole, B., Raetsch, G., Schölkopf, B., Bachem, O., and Tschannen, M. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, 2020. URL https://proceedings.mlr.press/v119/locatello20a.html.

Nene, S. A., Nayar, S. K., Murase, H., et al. Columbia object image library (coil-20). 1996.

Painter, M., Prugel-Bennett, A., and Hare, J. Linear disentangled representations and unsupervised action estimation. *Advances in Neural Information Processing Systems*, 33, 2020.

Perez Rey, L. A., Menkovski, V., and Portegies, J. Diffusion variational autoencoders. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 2704–2710, 2020.

Pfau, D., Higgins, I., Botev, A., and Racanière, S. Disentangling by Subspace Diffusion. pp. 1–21, 2020. URL http://arxiv.org/abs/2006.12982.

Quessard, R., Barrett, T. D., and Clements, W. R. Learning Group Structure and Disentangled Representations of Dynamical Environments. *Advances in Neural Information Processing Systems*, 33, 2020.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning-Volume 32*, 2014.

Ridgeway, K. and Mozer, M. C. Learning deep disentangled embeddings with the f-statistic loss. In *Advances in Neural Information Processing Systems*, pp. 185–194, 2018.

Soatto, S. Steps Towards a Theory of Visual Information: Active Perception, Signal-to-Symbol Conversion and the Interplay Between Sensing and Control. *arXiv preprint arXiv:1110.2053*, 2011.

Sosnovik, I., Szmaja, M., and Smeulders, A. Scale-Equivariant Steerable Networks. *International Conference on Learning Representations*, pp. 1–14, 2019.

TensorFlowDatasets, . TensorFlow Datasets, a collection of ready-to-use datasets. https://www.tensorflow.org/datasets, 2021.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5028–5037, 2017.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. 3D ShapeNets: A Deep Representation for Volumetric Shapes. 2014. ISSN 10636919. doi: 10.1109/CVPR.2015.7298801. URL http://arxiv.org/abs/1406.5670.

A. Preliminaries: Group Theory

In this appendix, we summarize some concepts from group theory that are important to understand the main text of the paper. Group theory provides a useful language to formalize the notion of symmetry transformations and their effects. For a more elaborate discussion we refer the reader to the book from Hall (2015) on group theory.

Group A group is a non-empty set G together with a binary operation $\circ : G \times G \to G$ that satisfies three properties:

1. *Associativity*: For all $f, g, h \in G$, it holds that $f \circ (g \circ h) = (f \circ g) \circ h$.
2. *Identity*: There exists a unique element $e \in G$ such that for all $g \in G$ it holds that $e \circ g = g \circ e = g$.
3. *Inverse*: For all $g \in G$ there exists an element $g^{-1} \in G$ such that $g^{-1} \circ g = g \circ g^{-1} = e$.

Direct product Let G and G' be two groups. The *direct product*, denoted by $G \times G'$, is the group with elements $(g, g') \in G \times G'$ with $g \in G$ and $g' \in G'$, and the binary operation $\circ : G \times G' \to G \times G'$ such that $(g, g') \circ (h, h') = (g \circ h, g' \circ h')$.

Lie group A Lie group is a group where G is a smooth manifold, this means it can be described in a local scale with a set of continuous parameters and that one can interpolate continuously between elements of G.

Group action Let A be a set and G a group. The *group action* of G on A is a function $G_A : G \times A \to A$ that has the properties [6]

1. $G_A(e, a) = a$ for all $a \in A$
2. $G_A(g, (G_A(g', a)) = G_A(g \circ g', a)$ for all $g, g' \in G$ and $a \in A$

Regular action The action of G on A is regular if for every pair of elements $a, a' \in A$ there exists a unique $g \in G$ such that $g \cdot a = a'$.

Group representation A *group representation* of G in the vector space V is a function $\rho : G \to GL(V)$ (where $GL(V)$ is the general linear group on V) such that for all $g, g' \in G$ $\rho(g \circ g') = \rho(g) \circ \rho(g')$ and $\rho(e) = \mathbb{I}_V$, where $\mathbb{I}_V$ is the identity matrix.

Direct sum of representations The direct sum of two representations $\rho_1 : G \to GL(V)$ in V and $\rho_2 : G \to GL(V')$ in V' is a group representation $\rho_1 \oplus \rho_2 : G \to GL(V \oplus V')$ over the direct sum $V \oplus V'$, defined for $v \in V$ and $v' \in V'$ as:

$$(\rho_1 \oplus \rho_2)(g) \cdot (v, v') = (\rho_1(g) \cdot v, \rho_2(g) \cdot v') \tag{12}$$

B. Linear Symmetry-Based Disentanglement: Definition with respect to World States

Higgins et al. (2018) provide a formal definition of linear disentanglement that connects symmetry transformations affecting the real world (from which data is generated) to the internal representations of a model. In the main text, we provide a definition from the perspective of a group action on the data directly, but the original definition considers an extra conceptual world state as well. Here, we describe the original setting in more detail, and explain why we choose a more direct and practical version of the definition.

The definition assumes the following setting. W is the set of possible world states, with underlying symmetry transformations that are described by a group G and its action $\cdot : G \times W \to W$ on W. In particular, G can be decomposed as the direct product of K groups $G = G_1 \times \ldots \times G_K$. Data is obtained via an *observation* function $b : W \to X$ that maps world states to observations in a *data space* X. A model's internal representation of data is modeled with the *encoding* function $h : X \to Z$ that maps data to the *embedding space* Z. Together, the observation and the encoding constitute the model's internal representation of the real world $f : W \to Z$ with $f(w) = h \circ b(w)$. The definition for Linearly Symmetry-Based Disentangled (LSBD) representations then formalizes the requirement that a model's internal representation f should reflect

[6] To avoid notational clutter, we write $G_A(g, a) = g \cdot a$ where the set A on which $g \in G$ acts can be inferred from the context.

and disentangle the transformation properties of the real world, and that the transformation properties of the model's internal representations should be linear.

The original definition considers G acting on W and involves the model's internal representation $f : W \to Z$, but since we do not directly observe W it is more practical to evaluate LSBD with respect to the encoding map $h : X \to Z$ instead. If the action of G on W is *regular*[7] and the observation map $b : W \to X$ is *injective*[8] though, we can instead define LSBD with respect to the action of G on X and the encoding map h, as shown in the main text.

C. Inner Product

To describe the norm $\|\cdot\|_{\rho,h,\mu}$ used in the definition of $\mathcal{D}_{\text{LSBD}}$ we start with an arbitrary inner product $(\cdot,\cdot)$ on the linear latent space Z. Assume that ρ is linearly disentangled and accordingly splits in irreducible representations $\rho_k : G \to Z_k$ where $Z = Z_1 \oplus \cdots \oplus Z_K$ for some $K \in \mathbb{N}$. We will define a new inner product $\langle\cdot,\cdot\rangle_{\rho,h,\mu}$ on Z as follows. First of all we declare Z_k and Z_m to be orthogonal with respect to $\langle\cdot,\cdot\rangle_{\rho,h,\mu}$ if $k \neq m$. We denote by π_k the orthogonal projection on Z_k.

For $z, z' \in Z_i$, we set

$$\langle z, z'\rangle_{\rho,h,\mu} := \lambda_{k,h,\mu}^{-1} \int_{g\in G} (\rho(g)\cdot z, \rho(g)\cdot z')d\mathfrak{m}(g) \tag{13}$$

where $\mathfrak{m}$ is the (bi-invariant) Haar measure normalized such that $\mathfrak{m}(G) = 1$ and set

$$\lambda_{k,h,\mu} := \int_X \int_G \|\pi_k(h(x))\|^2 d\mathfrak{m}(g)d\mu(x) \tag{14}$$

if the integral on the right-hand side is strictly positive and otherwise we set $\lambda_k := 1$. This construction completely specifies the new inner product, and it has the following properties:

- the subspaces Z_k are mutually orthogonal,
- $\rho_k(g)$ is orthogonal on Z_k for every $g \in G$, in other words ρ_k maps to the orthogonal group on Z_k. Moreover, ρ maps to the orthogonal group on Z. This follows directly from the bi-invariance of the Haar measure and the definition of $\langle\cdot,\cdot\rangle_{\rho,h,\mu}$.
- If π_k is the orthogonal projection to Z_k, then

$$\int_X \|\pi_k(h(x))\|^2_{\rho,h,\mu} d\mu(x) = 1 \tag{15}$$

  if the integral on the left is strictly positive.

For an arbitrary pair $z, z' \in Z$ the inner product $\langle\cdot,\cdot\rangle_{\rho,h,\mu}$ is given by

$$\langle z, z'\rangle_{\rho,h,\mu} = \sum_{k=1}^{K} \lambda_{k,h,\mu}^{-1} \int_{g\in G} (\rho(g)\cdot\pi_k(z), \rho(g)\cdot\pi_k(z'))d\mathfrak{m}(g) \tag{16}$$

D. Evaluation of Equivariance by $\mathcal{D}_{\text{LSBD}}$

We will now give an alternative expression for the disentanglement metric $\mathcal{D}_{\text{LSBD}}$, since it will more visibly relate to the definition of equivariance. To avoid notational cluttering, in this section we will denote the norm $\|\cdot\|_{\rho,h,\mu}$ as $\|\cdot\|_*$. Let $\rho \in \mathcal{P}(G, Z)$ be a linear disentangled representation of G in Z. By expanding the inner product (or by using usual

[7] This assumption holds in most practical cases with a suitable description of G.

[8] This is typically the case, but if not it can be solved through active sensing, see Soatto (2011).

computation rules for expectations and variances), we first find that

$$\begin{aligned}
&\int_G \left\| \rho(g)^{-1} \cdot h(g \cdot x_0) - \int_G \rho(g')^{-1} \cdot h(g' \cdot x_0) d\nu(g') \right\|_*^2 d\nu(g) \\
&= \int_G \left\| \rho(g)^{-1} \cdot h(g \cdot x_0) \right\|_*^2 d\nu(g) - \left\| \int_G \rho(g)^{-1} \cdot h(g \cdot x_0) d\nu(g) \right\|_*^2 \\
&= \frac{1}{2} \int_G \int_G \| \rho(g)^{-1} \cdot h(g \cdot x_0) - \rho(g')^{-1} \cdot h(g' \cdot x_0) \|_*^2 d\nu(g) d\nu(g').
\end{aligned} \tag{17}$$

We now use that ρ maps to the orthogonal group for $(\cdot, \cdot)_*$, so that we can write the same expression as

$$\frac{1}{2} \int_G \int_G \| \rho(g \circ g'^{-1})^{-1} \cdot h(((g \circ g'^{-1}) \cdot g') \cdot x_0) - h(g' \cdot x_0) \|_*^2 d\nu(g) d\nu(g'). \tag{18}$$

This brings us to the alternative characterization of $\mathcal{D}_{\text{LSBD}}$ as

$$\mathcal{D}_{\text{LSBD}} = \inf_{\rho \in \mathcal{P}(G,Z)} \frac{1}{2} \int_G \int_G \| \rho(g \circ g'^{-1})^{-1} h(((g \circ g'^{-1}) \cdot g') \cdot x_0) - h(g' \cdot x_0) \|_*^2 d\nu(g) d\nu(g'). \tag{19}$$

In particular, if for every data point x there is a unique group element g_x such that $x = g_x \cdot x_0$, the disentanglement metric $\mathcal{D}_{\text{LSBD}}$ can also be written as

$$\inf_{\rho \in \mathcal{P}(G,Z)} \frac{1}{2} \int_G \int_X \| \rho(g \circ g_x^{-1})^{-1} h((g \circ g_x^{-1}) \cdot x) - h(x) \|_*^2 d\nu(g) d\mu(x), \tag{20}$$

in which the equivariance condition appears prominently. The condition becomes even more apparent if ν is in fact the Haar measure itself, in which case the metric equals

$$\inf_{\rho \in \mathcal{P}(G,Z)} \frac{1}{2} \int_G \int_X \| \rho(g)^{-1} \circ h(g \cdot x) - h(x) \|_*^2 d\mathfrak{m}(g) d\mu(x). \tag{21}$$

E. Datasets

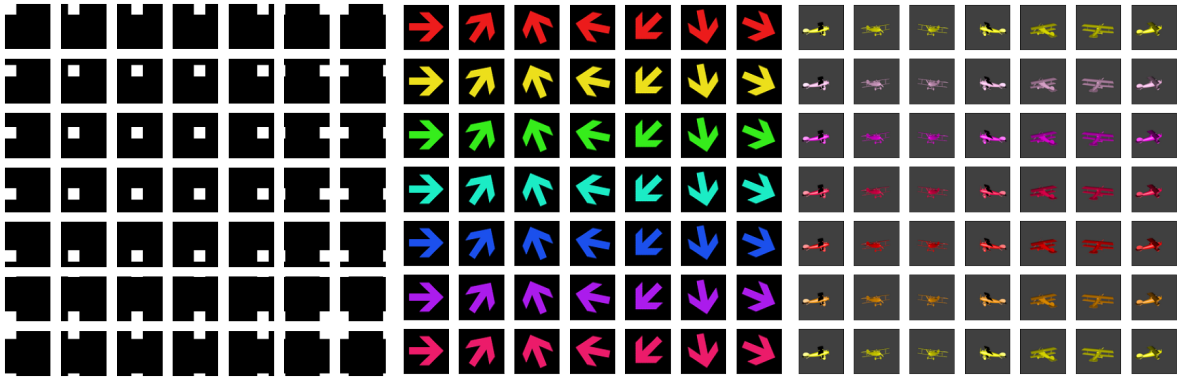
All datasets contain 64×64 pixel images. The Square, Arrow and Airplane datasets have a known group decomposition $G = \text{SO}(2) \times \text{SO}(2)$ describing the underlying transformations. In these three datasets, for each subgroup a fixed number of $|\mathcal{G}_k| = 64$ with $k \in \{1, 2\}$ transformations is selected. Each image is generated from a single initial data point upon which all possible group actions are applied, resulting in datasets with $|\mathcal{G}_1| \cdot |\mathcal{G}_2| = 4096$ images. The datasets exemplify different group actions of $\text{SO}(2)$: periodic translations, in-plane rotations, out-of-plane rotations, and periodic hue-shifts, see Fig. 7.

The ModelNet40 and the COIL-100 datasets consist of different objects rotating with respect to a vertical axis (out-of-plane rotation). For these datasets the group $G = \text{SO}(2)$ describes the underlying transformations that each object undergoes, see Fig. 7. The different objects can be seen as non-symmetric variability in the data. In this particular case, each object has its own base-point x_0 from which data is generated. The metric $\mathcal{D}_{\text{LSBD}}$ is then evaluated per object instance for the group $G = \text{SO}(2)$, the value of $\mathcal{D}_{\text{LSBD}}$ is calculated and averaged across all available objects. Fig. 8 shows some example paths of consecutive observations for the Square, Arrow, and Airplane datasets, as explained in Sect. 6.

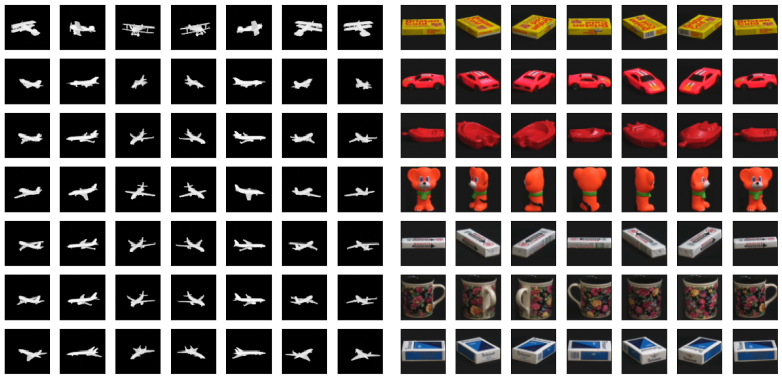
Square This dataset consists of a set of images of a black background with a square of 16×16 white pixels. The dataset is generated applying vertical and horizontal translations of the white square considering periodic boundaries.

Arrow This dataset consists of a set of images depicting a colored arrow at a given orientation. The dataset is generated by applying cyclic shifts of its color and in-plane rotations. The cyclic color shifts were obtained by preselecting a fixed set of 64 colors from a circular hue axis. The in-plane rotations were obtained by rotating the arrow along an axis perpendicular to the picture plane over 64 predefined positions.

Airplane This dataset consists of renders obtained using Blender v2.7 (Community, 2020) from a 3D model of an airplane within the ModelNet40 dataset (Wu et al., 2014) (this dataset is provided for the convenience of academic research only). We created each image by varying two properties: the airplane's color and its orientation with respect to the render camera. The orientation was changed via rotation with respect to a vertical axis (out-of-plane rotation). The colors of the model were selected from a predefined cyclic set of colors similar to the arrow rotation dataset.



(a) Square (b) Arrow (c) Airplane



(d) ModelNet40 (e) COIL-100

Figure 7: Example images from each of the datasets used. Each image corresponds to an example data point for a combination of two factors, e.g. color and orientation. The factors change horizontally and vertically. The boundaries for the Square, Arrow and Airplane dataset are periodic. For the ModelNet40 and COIL-100 dataset, the vertical direction represents different object instances and the horizontal direction represents the rotation of the corresponding object.

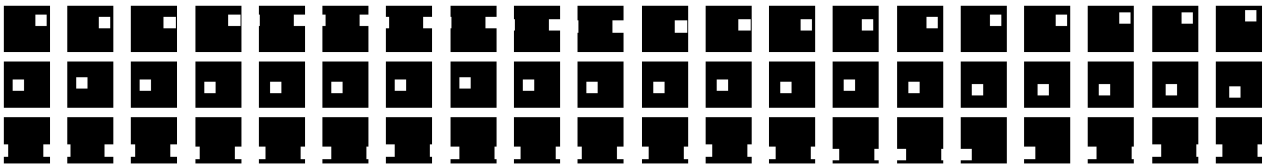
ModelNet40 This dataset also consists of a dataset of renders obtained using Blender v2.7 (Community, 2020) from the 626 training 3D models within the airplane category of the ModelNet40 dataset (Wu et al., 2014). We created each image by varying each airplane's orientation with respect to the render camera, via rotation with respect to a vertical axis (out-of-plane rotation). In this case we used 64 orientations for each object, i.e. $|\mathcal{G}| = 64$, for a total of 626 objects, thus the dataset consists of 40,064 images.

COIL-100 This dataset (Nene et al., 1996) consists of images from 100 objects placed on a turntable against a black background. For each object, 72 views of the rotated object are provided. The original images have a resolution of 128×128 and were re-scaled to 64×64 to match our other datasets. In this case for each object $|\mathcal{G}| = 72$, thus the total dataset consists of 7200 images. This dataset is intended for non-commercial research purposes only. This dataset was obtained using TensorFlowDatasets (2021).

F. Experimental Settings and Hyperparameters

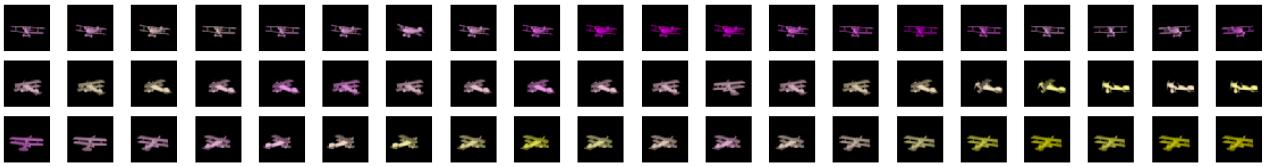
F.1. Architectures

Table 1 shows the encoder and decoder architectures used for almost all methods and datasets. The encoder's last layer depends on the method. For VAE, cc-VAE, FactorVAE, DIP-I, DIP-II, two dense layers with 4 units each were used. For LSBD-VAE and ΔVAE two dense layers with 4 and 2 units each were used. For Quessard a single dense layer with 4 units was used. The only model that was not trained with this architectures was LSBD-VAE/0 method for the ModelNet40 dataset the reason for this choice was that during training the loss was getting NaN values, in this case the architecture used was that



(a) Square

(b) Arrow



(c) Airplane

Figure 8: Example paths of consecutive observations.

of Table 2.

Table 1: Encoder and decoder architectures used in most methods.

Encoder	
Input	size (64,64, number channels)
Conv	filters 32, kernel 4, stride 2, ReLU
Conv	filters 32, kernel 4, stride 2, ReLU
Conv	filters 64, kernel 4, stride 2, ReLU
Conv	filters 64, kernel 4, stride 2, ReLU
Dense	units 256, ReLU
Dense(x2)	units depend on method
Decoder	
Input	size (number of latent dimensions)
Dense	units 256, ReLU
Dense	units 4*4*64, ReLU
Reshape	(4,4,64)
ConvT	filters 64, kernel 4, stride 2, ReLU
ConvT	filters 32, kernel 4, stride 2, ReLU
ConvT	filters 32, kernel 4, stride 2, ReLU
ConvT	filters (number channels), kernel 4, stride 2, Sigmoid

F.2. Hyperparameters

Table 3 shows the hyperparameters used to train each model for all datasets. Table 4 shows the hyperparameters used to train the LSBD-VAE models for each dataset. In the latter case, the number of epochs for the LSBD-VAE model were increased. The range of values used for the scale parameter t were increased for ModelNet40 and COIL-100 datasets since it was noticed that this provided better results in terms of data reconstruction and disentanglement. For the Arrow dataset, a

Table 2: Encoder and decoder architecture used to train LSBD-VAE/0 for ModelNet40 dataset.

ENCODER	
INPUT	SIZE (64, 64, NUMBER CHANNELS)
DENSE	UNITS 512, RELU, BATCH NORMALIZATION
DENSE	UNITS 256, RELU, BATCH NORMALIZATION
DENSE(X2)	UNITS DEPEND ON METHOD
DECODER	
INPUT	SIZE (NUMBER OF LATENT DIMENSIONS)
DENSE	UNITS 256, RELU, BATCH NORMALIZATION
DENSE	UNITS 512, RELU, BATCH NORMALIZATION
DENSE	UNITS 64*64*NUMBER OF CHANNELS, SIGMOID
RESHAPE	(64, 64, NUMBER OF CHANNELS)

value of $\gamma = 1$ was producing unstable results. However, the values 10, 100, 1000 or even 10000 were producing good results without significant changes among them. Therefore the value 100 was used for the datasets with the same structure (Square, Arrow and Airplane). For the ModelNet40 and COIL-100 the experiments showed that this hyperparameter for values as high as 10000 could affect the reconstructions, thus a lower value $\gamma = 1$ was chosen.

The training of the weakly-supervised models AdaGVAE and AdaMLVAE was done with a data generator that organized the available training data into pairs. The only condition introduced in Locatello et al. (2020) to train these models was to provide paired data with few factors changing among them. For our datasets, two factors change.

Table 3: Model hyperparameters for all datasets

MODEL	PARAMETERS
VAE	TRAINING STEPS 30000
β-VAE	$\beta = 5$, TRAINING STEPS 30000
CC-VAE	$\beta = 5$,$\gamma = 1000$, $c_{max} = 15$, ITERATION THRESHOLD 3500, TRAINING STEPS 30000
FACTOR	$\gamma = 1$, EPOCHS 30000
DIP-I	$\lambda_{od} = 1$, $\lambda_d = 10$, TRAINING STEPS 30000
DIP-II	$\lambda_{od} = 1$, $\lambda_d = 1$, TRAINING STEPS 30000
ADAGVAE	$\beta = 1$, EPOCHS 500
ADAMLVAE	$\beta = 1$, EPOCHS 500
QUESSARD	$\lambda = 0.01$, TRAJECTORIES 3000

Table 4: LSBD-VAE hyperparameters for all datasets

DATASETS	PARAMETERS
SQUARE, ARROW, AIRPLANE	$t \in [10^{-10}, 10^{-9}]$, $\gamma = 100.0$, EPOCHS 1500
MODELNET40	$t \in [10^{-10}, 10^{-5}]$, $\gamma = 1.0$, EPOCHS 1500
COIL-100	$t \in [10^{-10}, 10^{-5}]$, $\gamma = 1.0$, EPOCHS 6000

F.3. Hardware & Running Time

The hardware used across all experiments was a DGX station with 4 NVIDIA GPUs V100 and 32GB . Only one GPU was used per experiment. The running time for the LSBD-VAE across all 9 degrees of supervision $L \in \{0, 256, 768, 1024, 1280, 1536, 1792, 2048\}$ and all 10 runs (total $9 \cdot 10$ repetitions) for the datasets were: Arrow 33 ± 4 minutes Airplane 29 ± 4 minutes and Square 28 ± 4 minutes. The running time for the LSBD-VAE across 2 degrees of supervision and 10 runs (total $2 \cdot 10$ repetitions) for ModelNet40 was 136 ± 10 minutes and for COIL-100 90 ± 6 minutes. For the method from (Quessard et al., 2020) the training times were approximately 30 minutes across all datasets. The training times for the methods from `disentanglement_lib` (Locatello et al., 2019) were not measured.

F.4. Code Licenses

The `disentanglement_lib` (Locatello et al., 2019) code is registered with an Apache 2.0 License while the code used to reproduce the method by Quessard et al. (2020) is registered with an MIT license.

G. Qualitative Results

G.1. Data Generation

Inspecting data generated by a model can help understand the structure of the learnt latent space in a qualitative way. Fig. 9 shows generated data obtained by sampling and decoding ten latent variables for each of the models trained on the COIL-100 and ModelNet40 airplanes datasets. Each latent variable is sampled from the prior over the latent space and decoded to produce an image.

In general, all models but one produce similar results consisting of objects with unclear shape or identity. It is important to highlight the AdaGVAE weakly-supervised model trained on COIL-100 since it appears to have a degenerate decoder producing only yellow objects. Such behaviour occurs for all ten trained instances of the AdaGVAE model.

Even though the randomly generated images seem to have no clear identity or shape for COIL-100, LSBD-VAE allows to better determine the identity of such sampled models, by showing multiple orientations thanks to the structure of its latent space. LSBD-VAE uses a latent space combining an S^1 manifold encouraged to encode information about the $SO(2)$ rotations and an Euclidean latent space encouraged to represent the information about the object's identity.

By first sampling a latent variable from the Euclidean latent space and combining it with a set of regularly spaced latent variables along S^1 we can observe some consistency in object identity, see Fig. 10. Such data generation cannot be directly obtained from traditional disentanglement methods since there is no clear direction representing either the object identities or the orientations.

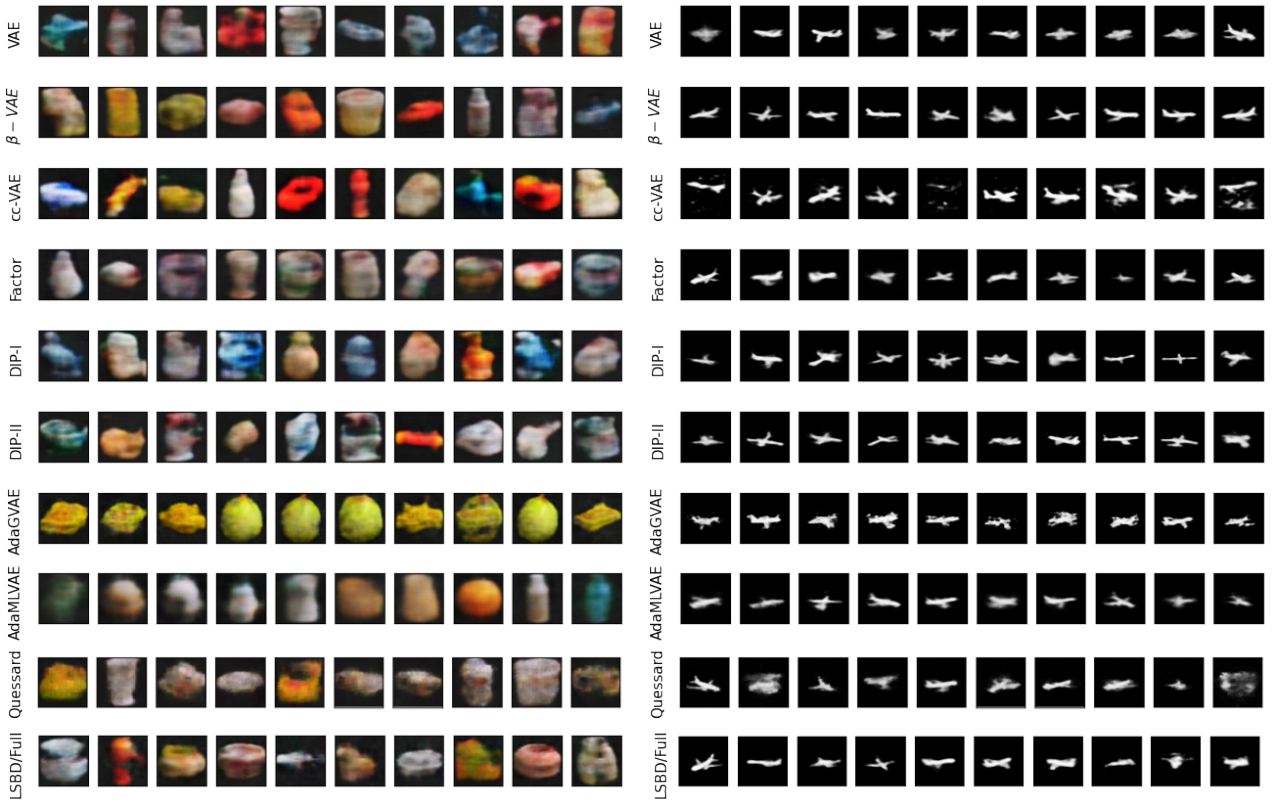
G.2. Object Interpolation

Next, we will show how the latent space is structured among the latent variables representing the objects' identities for models trained with COIL-100. We show the generated data obtained from decoding linearly interpolated latent variables between different objects to show the transitions between objects and orientations.

For LSBD-VAE the interpolation is simple; first the latent variables associated to the identity of the start and end objects are estimated by averaging the Euclidean latent variables of all images per object. Second, the linear interpolation between the object identity latent variables of the start and end object is calculated to generate a path through the object identity space. Finally, the estimated identity variables in the path are combined with regularly spaced variables in the orientation space S^1 and decoded. See Fig. 12b (a).

In the case of the traditional disentanglement methods we cannot produce a latent variable representing an object's identity, so there is no clear traversal between objects. In this case, a linear interpolation between an image from the start object to and end object is calculated and the latent variables are decoded, see Fig. 12b. Notice that we cannot easily produce an image of an object with an arbitrary orientation since we do not know the shape of the loop in the latent space representing an object.

Fig. 11 shows the generated images obtained by interpolating between two objects. We only show cc-VAE representing traditional models since that method attained the lowest $\mathcal{D}_{\text{LSBD}}$. A particularly interesting interpolation is between the wooden object and the orange cat figure. The interpolation of cc-VAE shows how a green object is also crossed in between while LSBD-VAE shows a consistent transition between the objects a visual explanation of this observation is presented in Fig. 12b (b).



(a) COIL 100

(b) ModelNet40 Airplanes

Figure 9: Images obtained by decoding latent variables sampled according to the prior over the latent space for different models trained on the COIL-100 and ModelNet40 airplanes datasets.

H. Full results

The full results for all experiments on all datasets are given in Tables 5, 6, 7, 8, and 9. We report the mean and standard deviation over 10 runs for each experiment.

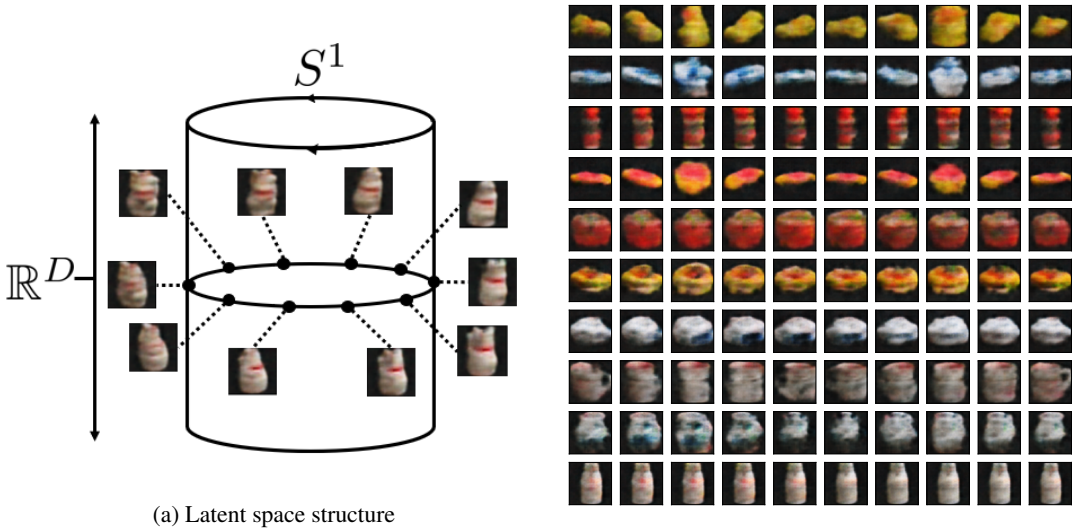
H.1. Limited Supervision Suffices to Learn LSBD Representations

The results obtained from Tables 5, 6, 7 show that we do not need transformation-labels for all data points, only a subset of labeled pairs is sufficient to learn LSBD representations. To further highlight this, Fig. 13 shows $\mathcal{D}_{\text{LSBD}}$ scores for LSBD-VAE trained on the Square, Arrow, and Airplane datasets respectively, for various values for the number of labeled pairs L. For each L and each dataset, we trained 10 models so we can report box plots of the $\mathcal{D}_{\text{LSBD}}$ scores.

For low values of L we see worse scores and high variability. But for slightly higher L, scores are consistently good, starting already at $L = 512$ for the Square, $L = 768$ for the Arrow, and $L = 256$ for the Airplane. This corresponds to respectively 25%, 37.5%, and 12.5% of the data being involved in a labeled pair. Moreover, we see that with just a little supervision we outperform the best traditional method on $\mathcal{D}_{\text{LSBD}}$. Overall, these results suggest that with some expert knowledge (about the underlying group and a suitable representation) and limited annotation of transformations, LSBD can be achieved.

H.2. Quessard Arrow

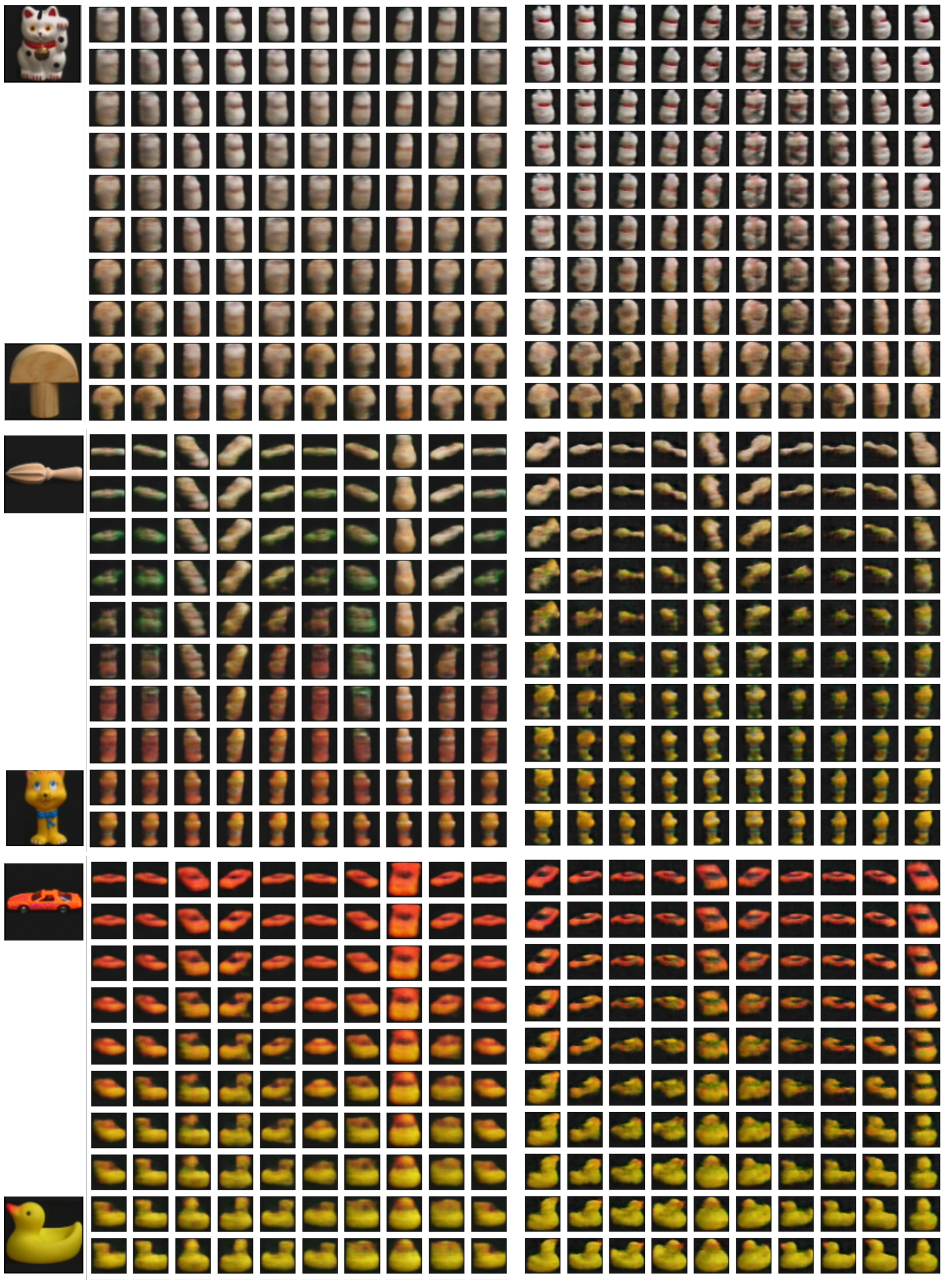
In the main text we mentioned that we did not reproduce good results with Quessard et al. (2020)'s method on the Arrow and Square dataset. We highlight a particular case for the Arrow dataset, where the method clearly learns the rotations of the arrow but fails to learn color. Fig. 14 shows reconstructed Arrow images. Since color isn't learned well, this example doesn't get a good $\mathcal{D}_{\text{LSBD}}$ score, even though rotation is properly linearly disentangled.



(a) Latent space structure

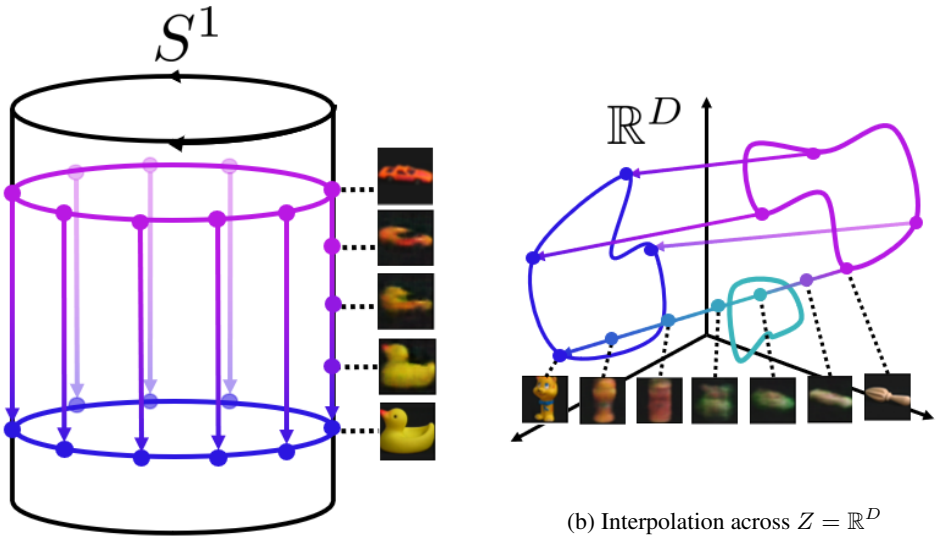
(b) Generated data

Figure 10: Image generation by traversing the circular latent variable for a sampled object identity. The high dimensional Euclidean space is depicted as a single dimension in a hyper-cylinder. (a) The latent variable corresponding to the identity is sampled from the prior over the Euclidean latent space and combined with regularly spaced latent variables on S^1. (2) Each row presents the decoded images for a fixed Euclidean latent variable while each column shows the images for a fixed latent variable on S^1. The images are obtained from decoding the latent variables with LSBD-VAE/full trained on the COIL-100 dataset.



(a) cc-VAE (b) LSBD-VAE

Figure 11: Images produced from the decoding of interpolated latent variables using cc-VAE and LSBD-VAE trained with COIL-100. Three interpolations between two objects are shown. Each column represents the transitions between objects while each row shows images that should correspond to different orientations.



(a) Interpolation across $Z = S^1 \times \mathbb{R}^D$

(b) Interpolation across $Z = \mathbb{R}^D$

Figure 12: Diagrams illustrating the interpolation between the latent variables associated to two objects. (a) Interpolation across a hyper-cylinder within $Z = S^1 \times \mathbb{R}^D$ used by LSBD-VAE. (b) Interpolation across $Z = \mathbb{R}^D$ of traditional disentanglement models. In the traditional disentanglement models the linear interpolation can show the crossing of the latent codes associated to unexpected objects.

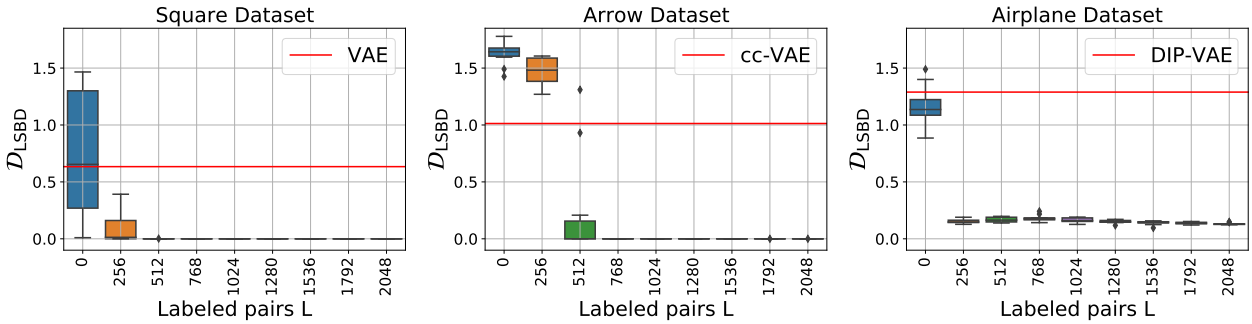


Figure 13: Box plots for $\mathcal{D}_{\text{LSBD}}$ scores over 10 training repetitions for different numbers of labeled pairs L, for all datasets. The red line indicates the best-performing traditional disentanglement method.

(a) Input

(b) Reconstructions

Figure 14: Results from Quessard et al. (2020)'s method on the Arrow dataset

Table 5: Scores for the Square dataset.

MODEL	BETA ↑	FACTOR ↑	SAP ↑	DCI ↑	MIG ↑	MOD ↑	$\mathcal{D}_{\text{LSBD}}$ ↓
VAE	$.945_{\pm.061}$	$.835_{\pm.140}$	$.019_{\pm.004}$	$.009_{\pm.005}$	$.013_{\pm.004}$	$.579_{\pm.202}$	$.634_{\pm.440}$
β-VAE	$.980_{\pm.033}$	$.913_{\pm.095}$	$.021_{\pm.006}$	$.017_{\pm.011}$	$.021_{\pm.014}$	$.642_{\pm.147}$	$.732_{\pm.488}$
CC-VAE	$.508_{\pm.023}$	$.000_{\pm.000}$	$.003_{\pm.002}$	$.007_{\pm.002}$	$.014_{\pm.004}$	$.222_{\pm.110}$	$1.905_{\pm.023}$
FACTOR	$.974_{\pm.048}$	$.910_{\pm.104}$	$.020_{\pm.003}$	$.019_{\pm.017}$	$.017_{\pm.010}$	$.712_{\pm.183}$	$.667_{\pm.428}$
DIP-I	$.972_{\pm.042}$	$.861_{\pm.097}$	$.020_{\pm.005}$	$.010_{\pm.002}$	$.011_{\pm.002}$	$.618_{\pm.117}$	$1.109_{\pm.312}$
DIP-II	$.930_{\pm.119}$	$.848_{\pm.137}$	$.018_{\pm.004}$	$.010_{\pm.004}$	$.015_{\pm.007}$	$.607_{\pm.207}$	$.907_{\pm.559}$
ADAGVAE	$.841_{\pm.230}$	$.707_{\pm.386}$	$.009_{\pm.009}$	$.024_{\pm.015}$	$.012_{\pm.005}$	$.473_{\pm.185}$	$.666_{\pm.378}$
ADAMLVAE	$.737_{\pm.208}$	$.465_{\pm.403}$	$.008_{\pm.008}$	$.016_{\pm.006}$	$.013_{\pm.007}$	$.338_{\pm.128}$	$1.063_{\pm.387}$
QUESSARD	$.504_{\pm.021}$	$.000_{\pm.000}$	$.004_{\pm.003}$	$.007_{\pm.004}$	$.018_{\pm.008}$	$.354_{\pm.213}$	$1.686_{\pm.294}$
LSBD-VAE /0	$.970_{\pm.079}$	$.913_{\pm.121}$	$.018_{\pm.003}$	$.052_{\pm.052}$	$.018_{\pm.004}$	$.884_{\pm.183}$	$.749_{\pm.554}$
LSBD-VAE /256	$1.000_{\pm.000}$	$1.000_{\pm.001}$	$.021_{\pm.004}$	$.267_{\pm.152}$	$.027_{\pm.007}$	$.986_{\pm.023}$	$.104_{\pm.147}$
LSBD-VAE /512	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.021_{\pm.006}$	$.393_{\pm.022}$	$.025_{\pm.005}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /768	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.019_{\pm.004}$	$.387_{\pm.014}$	$.025_{\pm.004}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1024	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.005}$	$.398_{\pm.020}$	$.024_{\pm.003}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1280	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.023_{\pm.003}$	$.389_{\pm.016}$	$.023_{\pm.003}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1536	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.004}$	$.398_{\pm.013}$	$.027_{\pm.002}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1792	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.020_{\pm.004}$	$.397_{\pm.016}$	$.027_{\pm.005}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /FULL	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.021_{\pm.006}$	$.380_{\pm.027}$	$.027_{\pm.005}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /PATHS							$.005_{\pm.002}$

Table 6: Scores for the Arrow dataset.

MODEL	BETA ↑	FACTOR ↑	SAP ↑	DCI ↑	MIG ↑	MOD ↑	$\mathcal{D}_{\text{LSBD}}$ ↓
VAE	$1.000_{\pm.000}$	$.646_{\pm.032}$	$.017_{\pm.004}$	$.009_{\pm.003}$	$.013_{\pm.004}$	$.961_{\pm.012}$	$1.316_{\pm.193}$
β-VAE	$.999_{\pm.002}$	$.588_{\pm.045}$	$.018_{\pm.004}$	$.008_{\pm.002}$	$.015_{\pm.005}$	$.898_{\pm.032}$	$1.178_{\pm.065}$
CC-VAE	$.982_{\pm.056}$	$.707_{\pm.102}$	$.019_{\pm.004}$	$.011_{\pm.005}$	$.016_{\pm.004}$	$.980_{\pm.038}$	$1.013_{\pm.096}$
FACTOR	$1.000_{\pm.000}$	$.659_{\pm.028}$	$.017_{\pm.003}$	$.008_{\pm.003}$	$.014_{\pm.002}$	$.935_{\pm.037}$	$1.526_{\pm.125}$
DIP-I	$1.000_{\pm.000}$	$.624_{\pm.042}$	$.020_{\pm.004}$	$.008_{\pm.002}$	$.012_{\pm.003}$	$.967_{\pm.027}$	$1.521_{\pm.113}$
DIP-II	$1.000_{\pm.000}$	$.644_{\pm.064}$	$.020_{\pm.004}$	$.009_{\pm.003}$	$.013_{\pm.004}$	$.973_{\pm.011}$	$1.616_{\pm.102}$
ADAGVAE	$1.000_{\pm.000}$	$.656_{\pm.137}$	$.016_{\pm.005}$	$.020_{\pm.009}$	$.009_{\pm.004}$	$.973_{\pm.042}$	$1.620_{\pm.147}$
ADAMLVAE	$.997_{\pm.008}$	$.706_{\pm.168}$	$.017_{\pm.007}$	$.019_{\pm.009}$	$.011_{\pm.004}$	$.943_{\pm.111}$	$1.395_{\pm.117}$
QUESSARD	$1.000_{\pm.000}$	$.596_{\pm.032}$	$.016_{\pm.006}$	$.008_{\pm.004}$	$.017_{\pm.008}$	$.999_{\pm.000}$	$1.183_{\pm.412}$
LSBD-VAE /0	$1.000_{\pm.001}$	$.664_{\pm.105}$	$.016_{\pm.002}$	$.009_{\pm.004}$	$.019_{\pm.005}$	$.897_{\pm.108}$	$1.627_{\pm.104}$
LSBD-VAE /256	$1.000_{\pm.000}$	$.662_{\pm.046}$	$.017_{\pm.005}$	$.009_{\pm.004}$	$.020_{\pm.005}$	$.963_{\pm.010}$	$1.475_{\pm.121}$
LSBD-VAE /512	$1.000_{\pm.000}$	$.956_{\pm.119}$	$.021_{\pm.006}$	$.297_{\pm.157}$	$.023_{\pm.003}$	$.967_{\pm.092}$	$.245_{\pm.474}$
LSBD-VAE /768	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.006}$	$.390_{\pm.022}$	$.026_{\pm.003}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1024	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.003}$	$.396_{\pm.026}$	$.026_{\pm.006}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1280	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.019_{\pm.005}$	$.401_{\pm.018}$	$.026_{\pm.004}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1536	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.019_{\pm.005}$	$.397_{\pm.017}$	$.026_{\pm.007}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /1792	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.020_{\pm.004}$	$.399_{\pm.018}$	$.026_{\pm.004}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /FULL	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.020_{\pm.006}$	$.444_{\pm.186}$	$.027_{\pm.004}$	$.999_{\pm.000}$	$.000_{\pm.000}$
LSBD-VAE /PATHS							$.016_{\pm.006}$

Table 7: Scores for the Airplane dataset.

MODEL	BETA ↑	FACTOR ↑	SAP ↑	DCI ↑	MIG ↑	MOD ↑	$\mathcal{D}_{\text{LSBD}}$ ↓
VAE	$1.000_{\pm.001}$	$.947_{\pm.054}$	$.023_{\pm.005}$	$.013_{\pm.005}$	$.020_{\pm.017}$	$.801_{\pm.045}$	$1.342_{\pm.084}$
β-VAE	$1.000_{\pm.001}$	$.997_{\pm.005}$	$.018_{\pm.005}$	$.036_{\pm.012}$	$.028_{\pm.012}$	$.816_{\pm.104}$	$1.481_{\pm.129}$
CC-VAE	$.858_{\pm.194}$	$.646_{\pm.353}$	$.010_{\pm.006}$	$.021_{\pm.011}$	$.018_{\pm.009}$	$.969_{\pm.034}$	$1.481_{\pm.174}$
FACTOR	$1.000_{\pm.000}$	$.984_{\pm.015}$	$.020_{\pm.003}$	$.021_{\pm.008}$	$.026_{\pm.013}$	$.810_{\pm.040}$	$1.382_{\pm.171}$
DIP-I	$1.000_{\pm.000}$	$.994_{\pm.008}$	$.022_{\pm.004}$	$.029_{\pm.012}$	$.026_{\pm.012}$	$.842_{\pm.073}$	$1.289_{\pm.150}$
DIP-II	$.998_{\pm.005}$	$.972_{\pm.031}$	$.021_{\pm.004}$	$.022_{\pm.013}$	$.030_{\pm.019}$	$.780_{\pm.054}$	$1.367_{\pm.129}$
ADAGVAE	$.962_{\pm.120}$	$.892_{\pm.314}$	$.013_{\pm.009}$	$.026_{\pm.016}$	$.010_{\pm.008}$	$.733_{\pm.264}$	$1.029_{\pm.288}$
ADAMLVAE	$1.000_{\pm.000}$	$.995_{\pm.007}$	$.019_{\pm.009}$	$.035_{\pm.011}$	$.017_{\pm.009}$	$.861_{\pm.073}$	$.994_{\pm.275}$
QUESSARD	$.999_{\pm.003}$	$.987_{\pm.026}$	$.018_{\pm.007}$	$.016_{\pm.009}$	$.018_{\pm.005}$	$.795_{\pm.107}$	$.558_{\pm.239}$
LSBD-VAE /0	$.536_{\pm.065}$	$.000_{\pm.000}$	$.002_{\pm.001}$	$.007_{\pm.004}$	$.005_{\pm.003}$	$.956_{\pm.046}$	$1.165_{\pm.180}$
LSBD-VAE /256	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.006}$	$.144_{\pm.011}$	$.023_{\pm.004}$	$.870_{\pm.039}$	$.153_{\pm.021}$
LSBD-VAE /512	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.023_{\pm.008}$	$.151_{\pm.015}$	$.020_{\pm.004}$	$.846_{\pm.032}$	$.168_{\pm.022}$
LSBD-VAE /768	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.004}$	$.140_{\pm.014}$	$.022_{\pm.005}$	$.832_{\pm.034}$	$.180_{\pm.030}$
LSBD-VAE /1024	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.020_{\pm.005}$	$.160_{\pm.015}$	$.022_{\pm.005}$	$.859_{\pm.032}$	$.165_{\pm.021}$
LSBD-VAE /1280	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.024_{\pm.004}$	$.153_{\pm.013}$	$.022_{\pm.003}$	$.876_{\pm.016}$	$.151_{\pm.015}$
LSBD-VAE /1536	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.021_{\pm.005}$	$.160_{\pm.016}$	$.022_{\pm.004}$	$.896_{\pm.025}$	$.140_{\pm.018}$
LSBD-VAE /1792	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.022_{\pm.005}$	$.163_{\pm.022}$	$.023_{\pm.003}$	$.904_{\pm.016}$	$.138_{\pm.010}$
LSBD-VAE /FULL	$1.000_{\pm.000}$	$1.000_{\pm.000}$	$.016_{\pm.008}$	$.161_{\pm.024}$	$.021_{\pm.006}$	$.913_{\pm.018}$	$.132_{\pm.009}$
LSBD-VAE /PATHS							$.185_{\pm.017}$

Table 8: Scores for the Modelnet40 Airplanes dataset.

MODEL	BETA ↑	FACTOR ↑	SAP ↑	DCI ↑	MIG ↑	MOD ↑	$\mathcal{D}_{\text{LSBD}}$ ↓
VAE	.995±.004	.838±.030	.013±.002	.013±.002	.009±.002	.415±.058	.393±.110
β-VAE	.995±.005	.857±.045	.012±.003	.015±.003	.009±.002	.447±.067	.285±.045
CC-VAE	.997±.003	.818±.093	.011±.003	.017±.004	.011±.003	.567±.063	.281±.191
FACTOR	.996±.004	.856±.052	.012±.002	.014±.003	.010±.003	.444±.077	.388±.096
DIP-I	.988±.009	.783±.070	.012±.002	.013±.002	.008±.001	.343±.082	.416±.142
DIP-II	.994±.006	.832±.042	.013±.003	.014±.003	.011±.002	.433±.080	.379±.130
ADAGVAE	.996±.006	.775±.079	.010±.006	.014±.006	.013±.004	.421±.092	.476±.218
ADAMLVAE	.996±.006	.784±.055	.012±.006	.014±.005	.014±.004	.445±.040	.580±.141
QUESSARD	.907±.192	.727±.384	.010±.005	.015±.007	.009±.004	.563±.108	.134±.294
LSBD-VAE /0	.990±.009	.863±.038	.011±.003	.015±.003	.014±.003	.538±.103	.731±.068
LSBD-VAE /FULL	1.000±.000	.990±.004	.012±.005	.052±.009	.020±.006	.947±.007	.041±.007

Table 9: Scores for COIL 100 dataset.

MODEL	BETA ↑	FACTOR ↑	SAP ↑	DCI ↑	MIG ↑	MOD ↑	$\mathcal{D}_{\text{LSBD}}$ ↓
VAE	1.000±.000	.674±.049	.014±.003	.016±.003	.011±.002	.986±.001	.463±.030
β-VAE	1.000±.001	.740±.024	.015±.004	.014±.004	.013±.003	.982±.001	.579±.095
CC-VAE	.999±.003	.723±.026	.013±.005	.014±.003	.013±.004	.985±.001	.406±.057
FACTOR	1.000±.001	.684±.041	.014±.002	.012±.002	.013±.004	.984±.001	.490±.024
DIP-I	.999±.002	.631±.025	.013±.004	.012±.002	.010±.002	.986±.001	.525±.109
DIP-II	1.000±.001	.643±.043	.013±.003	.014±.002	.011±.002	.985±.001	.568±.079
ADAGVAE	1.000±.000	.672±.021	.015±.007	.016±.005	.014±.006	.984±.001	.431±.049
ADAMLVAE	1.000±.000	.688±.027	.011±.003	.015±.006	.018±.009	.984±.002	.400±.076
QUESSARD	1.000±.000	.780±.044	.014±.004	.014±.002	.011±.003	.973±.004	.396±.055
LSBD-VAE /0	1.000±.001	.739±.047	.014±.003	.014±.001	.011±.001	.982±.004	.515±.099
LSBD-VAE /FULL	1.000±.000	.655±.028	.015±.004	.029±.003	.013±.003	.802±.056	.112±.026