Equivariance by Contrast: Identifiable Equivariant Embeddings from Unlabeled Finite Group Actions

Tobias Schmidt[1,2], **Steffen Schneider**[1,2,3*] **and Matthias Bethge**[3*]
[1]Institute of Computational Biology, Helmholtz Munich
[2]Munich Center for Machine Learning (MCML)
[3]Tübingen AI Center

Abstract

We propose Equivariance by Contrast (EbC) to learn equivariant embeddings from observation pairs $(\mathbf{y}, g \cdot \mathbf{y})$, where g is drawn from a finite group acting on the data. Our method jointly learns a latent space and a group representation in which group actions correspond to invertible linear maps—without relying on group-specific inductive biases. We validate our approach on the infinite dSprites dataset with structured transformations defined by the finite group $G := (R_m \times \mathbb{Z}_n \times \mathbb{Z}_n)$, combining discrete rotations and periodic translations. The resulting embeddings exhibit high-fidelity equivariance, with group operations faithfully reproduced in latent space. On synthetic data, we further validate the approach on the non-abelian orthogonal group $O(n)$ and the general linear group $GL(n)$. We also provide a theoretical proof for identifiability. While broad evaluation across diverse group types on real-world data remains future work, our results constitute the first successful demonstration of general-purpose encoder-only equivariant learning from group action observations alone, including non-trivial non-abelian groups and a product group motivated by modeling affine equivariances in computer vision.

1 Introduction

In many real-world inference problems, the relationship between observations is governed by structured transformations. The same sample may be observed before and after an "action" has been applied. In computer vision, an object may be observed in the form of an image before and after rotations, translations, or other types of transformations have been applied [6, 11, 32]. In biology, large-scale single-cell transcriptomic datasets increasingly contain observations of cells before and after perturbations in the form of gene knockouts [17] or pharmacological intervention [41]. In neuroscience, neural activity reflects changing brain states under sensory input or behavioral output [45]. In all these cases, the key to understanding the data lies not only in modeling individual observations but also in capturing the structured relationships between them.

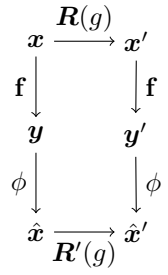

Figure 1: Commutative diagram showing data and model for EbC.

This motivates the goal of learning *equivariant* embeddings. In this embedding space, actions are represented by linear transformations. To address this challenge in a theoretically grounded way, we adopt a perspective rooted in nonlinear Independent Component Analysis (ICA) and group theory. Suppose that each observation $\boldsymbol{y} \in Y$ arises from a latent representation $\boldsymbol{x} \in X$ through an unknown, injective nonlinear mixing function $\mathbf{f}$, i.e., $\boldsymbol{y} = \mathbf{f}(\boldsymbol{x})$. Nonlinear ICA aims to invert this process: to learn an encoder $\phi \approx \boldsymbol{f}^{-1}$ that recovers the latent structure from the observed data.

*Co-corresponding authors: steffen.schneider@helmholtz-munich.de, matthias.bethge@bethgelab.org.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

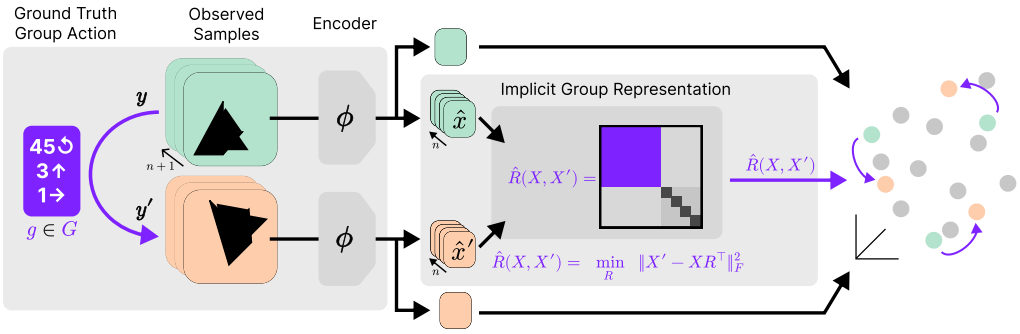

Figure 2: **Overview of the approach**. Left to right: One batch of observed sample consists of $n+1$ paired samples $\{(\boldsymbol{y}_i, \boldsymbol{y}_i')\}_{i=0}^{n+1}$ where $\boldsymbol{y}_i'$ are related to $\boldsymbol{y}_i$ via an unkown group action g such that $\boldsymbol{y}_i' = g \cdot \boldsymbol{y}_i$. An encoder ϕ maps these observations into latent space $\{(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_i')\}_{i=0}^{n+1}$, where n samples are used to estimate a representation $\hat{\boldsymbol{R}}$ of the group.

The nonlinear ICA problem becomes tractable by additional assumptions about the structure of the data-generating process [18, 19, 21, 22, 56]. Here, we leverage the property that many datasets do not consist of isolated samples but of pairs $(\boldsymbol{y}, \boldsymbol{y}')$, where $\boldsymbol{y}'$ is a transformed version of $\boldsymbol{y}$. Group theory provides a formalism to describe these transformations as the elements of a group G.

Each group element $g \in G$ can act on a latent $\boldsymbol{x} \in X$, yielding a transformed latent $g\boldsymbol{x}$. This operation is known as a *group action*. In observation space, we introduce the shorthand $\boldsymbol{y}' := g \cdot \boldsymbol{y}$ to denote the respective relation between $\boldsymbol{y}' = \mathbf{f}(g\boldsymbol{x})$ and $\boldsymbol{y} = \mathbf{f}(\boldsymbol{x})$. In the latent space, these group actions have a particularly simple form: they correspond to linear maps. Formally, a *group representation* is a homomorphism $\boldsymbol{R} : G \mapsto GL(X)$ which maps each group element to an invertible matrix acting on the vector space X. Thus, while the transformation $g \cdot \boldsymbol{y}$ may have complicated non-linear effects in observation space, in a suitable latent space it can be reduced to a linear relationship of the form:

$$\boldsymbol{x}' = g\boldsymbol{x} = \boldsymbol{R}(g)\boldsymbol{x}. \tag{1}$$

Our goal is to infer this relation directly from pairs $(\boldsymbol{y}, g \cdot \boldsymbol{y})$ of observable data and learn an encoder $\phi : Y \mapsto X'$ and a representation $\boldsymbol{R}'$ of the group such that

$$\phi(g \cdot \boldsymbol{y}) = \boldsymbol{R}'(g)\phi(\boldsymbol{y}), \tag{2}$$

and $\boldsymbol{R}' : G \mapsto GL(X')$ is a representation of G (potentially different from $\boldsymbol{R}$) on the vector space X'. Learning such representations directly from data is difficult. A growing body of work has approached this problem by leveraging pairs $(\boldsymbol{y}, g \cdot \boldsymbol{y})$ even when the specific group element g is unknown [15, 28, 52], with varying constraints: CARE [15] restricts itself to orthogonal representations on the hypersphere, STL [52] allows nonlinear equivariant relations in latent space, and the neural fourier transform [NFT; 28] requires to learn a generative model of the data.

Here, we propose *Equivariance by Contrast (EbC)*, to the best of our knowledge the first *encoder-only* method that learns *general linear* group representations from group action observations with a formal identifiability guarantee. In § 2, we present the algorithm which jointly learns the encoder ϕ and an implicit group representation via contrastive learning, without generative modeling or group-specific architectural biases. In § 3, we show that EbC recovers the true latent space and the underlying group representation up to a linear transformation. In §§ 4–6, we evaluate EbC on synthetic and structured vision datasets, including finite product groups $G := (R_m \times \mathbb{Z}_n \times \mathbb{Z}_n)$ and non-abelian groups such as $O(n)$ and $GL(n)$. EbC achieves high-fidelity equivariant embeddings across diverse settings.

2 Learning group-equivariant representations with contrastive learning

Figure 2 outlines our approach: Similar to previous work [15, 28, 36, 52], we assume that data come in the form of batches, which are grouped into $n+1$ pairs undergoing the same action g. This form of data is common in various scientific fields, for example, in neuroscience and biology. Pairs can also be derived from time-series data under the assumption that nearby points are governed by a shared action ["slowness prior"; 26].

The intuition behind our objective function is depicted in Figure 3: The model has access to a set of mixed samples related via the group action ($\boldsymbol{Y}$ and $\boldsymbol{Y}'$), along with a query sample $\boldsymbol{y}$. The objective

is to infer the group action from the examples $\boldsymbol{Y}$ and $\boldsymbol{Y}'$, apply it to the query, and select the correct answer $\boldsymbol{y}'$ among a set of options that include the correct answer alongside negative samples $\boldsymbol{y}'' \in S$. In our case, the set S contains negative samples randomly selected from the dataset, which cover all potential mismatches (different content, different group action, etc.) alongside the positive sample. We use contrastive learning to encode this objective in the likelihood

$$p_\phi(\boldsymbol{y}' \mid \boldsymbol{y}, \boldsymbol{Y}, \boldsymbol{Y}', S) = \frac{\exp\big(-\|\boldsymbol{u}_\phi(\boldsymbol{y}, \boldsymbol{Y}, \boldsymbol{Y}') - \phi(\boldsymbol{y}')\|^2\big)}{\sum_{\boldsymbol{y}'' \in S} \exp\big(-\|\boldsymbol{u}_\phi(\boldsymbol{y}, \boldsymbol{Y}, \boldsymbol{Y}') - \phi(\boldsymbol{y}'')\|^2\big)}. \tag{3}$$

The shorthand $\boldsymbol{u}_\phi$ denotes the operation of inferring the linear representation of the group element, $\hat{\boldsymbol{R}}(\phi(\boldsymbol{Y}), \phi(\boldsymbol{Y}'))$, and then applying it to the feature vector of the reference sample $\phi(\boldsymbol{y})$:

$$\boldsymbol{u}_\phi(\boldsymbol{y}, \boldsymbol{Y}, \boldsymbol{Y}') = \hat{\boldsymbol{R}}(\phi(\boldsymbol{Y}), \phi(\boldsymbol{Y}'))\phi(\boldsymbol{y}) \tag{4}$$

$$\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = \min_{\boldsymbol{R} \in \mathrm{GL}(d)} \|\boldsymbol{X}' - \boldsymbol{X}\boldsymbol{R}^\top\|_F^2. \tag{5}$$

To find the optimal feature encoder ϕ, we optimize the likelihood across all pairs of samples and uniformly sampled negative examples,

$$\min_\phi \mathcal{L}[\phi] = -\mathbb{E}_{\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{Y}, \boldsymbol{Y}', S}\left[\log p_\phi(\boldsymbol{y}' \mid \boldsymbol{y}, \boldsymbol{Y}, \boldsymbol{Y}', S)\right], \tag{6}$$

which is related to the InfoNCE loss [37] with the additional structure required for group learning.

Separating content and style In many cases, we want to produce embedding spaces in which we can separate *what* is transformed from *how* it is transformed. Intuitively, we consider these aspects of the latent representation to encode *content* and *style*. The content is the part of the representation that is expected to be invariant w.r.t. the group action g, whereas the style is expected to be equivariant w.r.t. the group action. In practice, it is therefore useful to split the vector space induced by ϕ into an equivariant and invariant part. Algorithmically, this is done by imposing additional structure on the matrix in Eq. (5), constraining the minimization across matrices of the form $\mathrm{diag}(GL_n, I_m) \subset GL(m+n)$. The representation we learn then has the form

$$\hat{\boldsymbol{R}}_{n+m}(\boldsymbol{X}, \boldsymbol{X}')' = \begin{pmatrix} \hat{\boldsymbol{R}}_n(\boldsymbol{X}, \boldsymbol{X}') & \boldsymbol{0}_{n \times m} \\ \boldsymbol{0}_{m \times n} & \boldsymbol{I}_m \end{pmatrix}. \tag{7}$$

This results in an encoder ϕ with a n-dimensional equivariant subspace, and a m-dimensional invariant subspace. We discuss the theoretical properties of this parametrization below. Note the conceptual similarity to subspace contrastive learning discussed in [44]. The difference here is that we do not use multiple instances of the contrastive loss but a single contrastive loss that directly learns a structured feature space.

3 Equivariance by Contrast is identifiable

By formalizing assumptions about the data-generating process, we can derive identifiability guarantees for the algorithm described in the previous section.

Dataset. We define the dataset in terms of the underlying data-generating process: Let $\boldsymbol{x} \in V \subseteq \mathbb{R}^d$ be a vector describing the ground truth latent components, let g be an element of a group G, and let $\mathbf{f} : V \to \mathbb{R}^D$ be an injective map. Assume that for each group element g, we have at least M pairs of samples $(\boldsymbol{y}_j, \boldsymbol{y}'_j)$ with

$$\boldsymbol{y}_i = \mathbf{f}(\boldsymbol{x}_i), \qquad \boldsymbol{y}'_i = \mathbf{f}(\boldsymbol{R}(g)\boldsymbol{x}_i) \qquad i = 1, \dots, M \tag{8}$$

where $\boldsymbol{R} : G \to \mathrm{GL}(d, \mathbb{R})$ is a representation of G on V. If $\boldsymbol{R}$ is structured accordingly (Eq. 7), $\boldsymbol{x}$ can be decomposed into an equivariant *style* and an invariant *content*, which is denoted as $\boldsymbol{c}$ further below in the experimental sections.

Implicit group representations. We model the group representation using the non-parametric approach outlined in Eq. 5. Assume that for each group element $g \in G$, we are given two matrices $\boldsymbol{X}, \boldsymbol{X}' \in \mathbb{R}^{M \times d}$, $M > d$, where the row vectors $(\boldsymbol{x}_i, \boldsymbol{x}'_i)$ are related via the group element as $\boldsymbol{x}'_i = g\boldsymbol{x}_i$, $i \in \{1, \dots, M\}$. As a shorthand, we write $\boldsymbol{X}' = g\boldsymbol{X}$. Then, the expression

$$\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = \min_{\boldsymbol{R} \in \mathrm{GL}(d)} \|\boldsymbol{X}' - \boldsymbol{X}\boldsymbol{R}^\top\|_F^2 \Leftrightarrow \hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}(\boldsymbol{X}^\top\boldsymbol{X}') \tag{9}$$

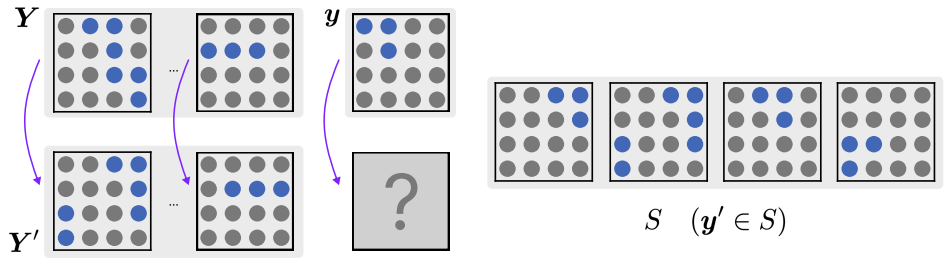

Figure 3: **Intuitive explanation of the training objective**. Given pairs of samples $(\boldsymbol{Y}, \boldsymbol{Y}')$ connected by an action g, infer the most likely action, and apply it to a novel sample $\boldsymbol{y}$ with different content. The training objective is contrastive, and aims to select the correct matching sample $\boldsymbol{y}'$ from a selection of positive/negative samples S.

is a representation of G with $\boldsymbol{R}(g) = \hat{\boldsymbol{R}}(\boldsymbol{X}, g\boldsymbol{X})$ for each $g \in G$. In practice, we do not have access to $(\boldsymbol{X}, \boldsymbol{X}')$ directly, but to a nonlinear projection of these points via the mixing function $\mathbf{f}$, denoted as $(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}')) = (\boldsymbol{Y}, \boldsymbol{Y}')$. We map these mixed samples to a feature space using a learnable encoder $\phi : \mathbb{R}^D \to \mathbb{R}^d$ and insert the resulting matrices into Eq. 9. Our goal is to optimize ϕ such that $\hat{\boldsymbol{R}}(\phi(\boldsymbol{Y}), \phi(\boldsymbol{Y}'))$ becomes a representation of G. An advantage of this approach is that both the feature space and the group representation are fully defined via the feature encoder ϕ.

Equivariance by Contrast learns group representations. Our theory builds on the canonical discriminative form discussed by Roeder et al. [40], which requires conditions on the diversity of the dataset to be fulfilled. Specifically, the latents and feature spaces of ϕ need to "sufficiently vary" across the points in S. In addition, the vectors of differences $\phi(\boldsymbol{y}_A) - \phi(\boldsymbol{y}_B)$ with $\boldsymbol{y}_A, \boldsymbol{y}_B$ columns of $\boldsymbol{S}$ need to span a d-dimensional feature space. The conditions are discussed in detail in Appendix A. This gives the following main results for our model:

Theorem 1 (Identitiability of the Group Representation; informal)**.** *Assume that $p_\phi = p_{\mathbf{f}^{-1}}$, and let the dataset satisfy the diversity conditions mentioned above. Define $\mathbf{h} := \phi \circ \mathbf{f}$. Then, for all points $\boldsymbol{x}$ in the support of the dataset:*

(a) We recover the original vector space $\mathbf{h}(\boldsymbol{x}) = \boldsymbol{L}\boldsymbol{x}$, up to an ambiguity $\boldsymbol{L} \in \mathrm{GL}(d)$.
(b) We recover a representation of the group, $\hat{\boldsymbol{R}}(\mathbf{h}(\boldsymbol{X}), \mathbf{h}(g\boldsymbol{X})) = \boldsymbol{L}\boldsymbol{R}(g)\boldsymbol{L}^{-1}$.

Proof sketch. *With a minor modification, Eq. 3 follows the canonical discriminative form for which Roeder et al. [40] proved linear identifiability. Due to the use of the more flexible Euclidean loss, we instead obtain affine identifiability for $\mathbf{h}(\boldsymbol{x}) = \boldsymbol{L}\boldsymbol{x} + \boldsymbol{b}$. The second condition requires $\hat{\boldsymbol{R}}\phi(\boldsymbol{x}) = \boldsymbol{L}\boldsymbol{R}\boldsymbol{x}$ and implies $\boldsymbol{b} = 0$. From these two conditions and the specific definition of our model, we can derive the results (a) and (b). The full proof is given in Appendix A.*

Corollary 1 (Equivariance)**.** *We have $\boldsymbol{x} \in V$, $\boldsymbol{R}(g)$ is a representation of the group in V; we have $\mathbf{h} : V \to W$ where we define $\mathbf{h} := \phi \circ \mathbf{f}$, and $\boldsymbol{R}'(g) = \hat{\boldsymbol{R}}(\mathbf{h}(\boldsymbol{X}), \mathbf{h}(g\boldsymbol{X}))$ is the representation on W. Then, we have*

$$\mathbf{h}(g\boldsymbol{x}) = g\mathbf{h}(\boldsymbol{x}) \tag{10}$$

Proof. The result follows from Theorem1. For all $\boldsymbol{x}$ we have $\mathbf{h}(g\boldsymbol{x}) = g\mathbf{h}(\boldsymbol{x})$. We insert the representation of g on V on the LHS, and on W on the RHS, to obtain $\mathbf{h}(\boldsymbol{R}(g)\boldsymbol{x}) = \boldsymbol{R}'(g)\mathbf{h}(\boldsymbol{x})$. We insert Thm. 1a to obtain $\boldsymbol{L}\boldsymbol{R}(g)\boldsymbol{x} = \boldsymbol{R}'(g)\boldsymbol{L}\boldsymbol{x}$. From here we obtain $\boldsymbol{L}\boldsymbol{R}(g)\boldsymbol{L}^{-1}\boldsymbol{x} = \boldsymbol{R}'(g)\boldsymbol{x}$ which we showed in Thm. 1b. □

4 Experiment Setup

We validate our proposed model on a set of diverse datasets with different underlying groups.

Synthetic Group Datasets We first consider three types of synthetic datasets following the data-generating process in Eq. 8 matching the assumptions required for Theorem 1. Group elements are taken from a subgroup of special orthogonal group $SO(n)$, the orthogonal group $O(n)$ and the general linear group $GL(n)$. We start by sampling random group elements $g \sim p_G(g)$ from $G \in \{SO(n), O(n), GL(n)\}$. We sample at least m (with $m > d$) random vectors $\boldsymbol{x} \in \mathbb{S}^n$ from the unit sphere and sample a content component $\boldsymbol{c} \in C$ from a finite set of vectors $C \subset S^d$. From

$(\boldsymbol{R}(g), \{\boldsymbol{x}_i\}^m, \boldsymbol{c})$ we generate $\boldsymbol{Y} = \{\boldsymbol{y}_i, \boldsymbol{y}_i'\}^m$, where $\boldsymbol{y}_i = \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{c})$ and $\boldsymbol{y}_i' = \mathbf{f}(\boldsymbol{R}(g)\boldsymbol{x}_i, \boldsymbol{c})$. We repeat this sampling procedure until we reach the desired dataset size. The nonlinear injective mixing function $\mathbf{f}$ is parameterized by a random 3-layer MLP [18] followed by a random linear map to a 50-dimensional space. For the full dataset we sample a maximum of 1000 matrices $\boldsymbol{R}(g)$ and a total of 1M pairs $(\boldsymbol{y}, \boldsymbol{y}')$. Unless stated otherwise, we choose $n = 3$, $d = 3$, $|C| = 100$. In Appendix C.1 we show additional results for $n \in 3, 5, 7, 9$ and in Appendix C.2 we report results on data efficiency. According to the assumptions of Theorem 1, the relationship $\boldsymbol{R}(g)\boldsymbol{x}_i$ in the data-generating process does not include noise. But we show a variation of this experiment In Appendix C.3 that includes noise in the application of the group action.

Infinite dSprites We leverage infinite dSprites [8, idSprites], an extension of dSprites [33] for validation on more diverse visual data. We subsample all images to a resolution of 64×64 The dataset allows for rich variations in the content and ensures that the resulting objects do not have any symmetries. By default, we use the same number of varying factors as in dSprites, namely 3 random shapes of 6 different sizes. For our purposes, we consider the combination of shape and size to be the content. Each of these objects may then be oriented in 40 different ways or have one of 32×32 x or y position translations. We consider the orientations and translations to be our groups of interest and sample the data such that we obtain closed groups. By default, we sample the dataset such that any paired sample $(\boldsymbol{y}, \boldsymbol{y}')$ undergoes a combination of these three types of transformations.

Training Protocol We use a three layer MLP with 512 hidden units for ϕ. We fit the implicit representation $\hat{\boldsymbol{R}}(\boldsymbol{Y}, \boldsymbol{Y}')$ using the `gels` least squares solver in PyTorch. By default, we use 84 sample pairs for idSprites and 12 for the synthetic data in $\boldsymbol{Y}$. Each batch has 1024 positive and 16k negative samples. We train the model for 20k with Adam [25] with learning rate 10^{-3}. We perform 80/10/10 train/valid/test splits. Metrics are computed out-of-sample, by fitting on one part of the valid split, and reporting on the other. Unless stated otherwise, standard deviations and confidence intervals are reported across three dataset and three model seeds. More details in Appendix B.

Baselines We leverage standard InfoNCE [37] training as a baseline (amounting to setting $\boldsymbol{R} = \boldsymbol{I}$), similar to SimCLR [2]. We also consider dynamics contrastive learning [30] using a linear dynamical system (LDS) or a switching linear dynamical system (SLDS). We use the same training protocol for the baselines as we do for our own EbC model (same encoder, batch size, training iterations, optimizer, and data loader).

Metrics Our metrics aim to quantify the representation of the content, the quality of the group representation, and the linear identifiability of the latent space. To measure the quality of the representation of the content we fit a Logistic Regression $(\boldsymbol{W}, \boldsymbol{b})$ and predict the class label $j \in K$ of content $\boldsymbol{c}_j$:

$$\mathrm{Acc}(C, K) := \text{TopK-Acc}(\sigma(\boldsymbol{W}\mathbf{h}(\boldsymbol{x}) + \boldsymbol{b}) \mid \boldsymbol{Y}') \qquad \sigma(\cdot) := \text{softmax}(\cdot) \tag{11}$$

Likewise, to quantify the linear identifiability of the embedding space (Theorem 1a), we measure:

$$R^2(x) := \max_{\boldsymbol{L} \in GL(n)} R^2_{\boldsymbol{x}}(\boldsymbol{L}\hat{\mathbf{h}}(\boldsymbol{x}), \boldsymbol{x}). \tag{12}$$

To measure the quality of the group representation we introduce two different metrics. First we introduce an R^2 based metric which we derive from Theorem 1b:

$$R^2(G) := R^2_{\boldsymbol{x}}[\boldsymbol{L}^*\hat{\boldsymbol{R}}\mathbf{h}(\boldsymbol{x}), \boldsymbol{R}\boldsymbol{x}], \quad \boldsymbol{L}^* = \arg\max_{\boldsymbol{L}} \|\boldsymbol{X} - h(\boldsymbol{X})\boldsymbol{L}^\top\|_F^2 \tag{13}$$

To measure the group representation without access to the ground truth latent representation $\boldsymbol{x}$, we introduce an Accuracy over the KNN lookup of the transformed input vector $\boldsymbol{y}'$:

$$\mathrm{Acc}(G, K) := \text{TopK-Acc}(\text{kNN}(\boldsymbol{Y}) | \boldsymbol{Y}') \qquad \text{kNN}(\boldsymbol{y}) := \arg\max_{\boldsymbol{j} \in \boldsymbol{J}} ||\phi(\boldsymbol{j}) - \hat{\boldsymbol{R}}\phi(\boldsymbol{y})||^2 \tag{14}$$

To compute this metric, we sample 20k random pairs $\boldsymbol{y}, \boldsymbol{y}'$ to solve a 20k-class classification problem. We compute the accuracy cross-validation across every of the 20k samples.

The metrics defined above can be split into two categories: $R^2(x)$ and $R^2(G)$ require access to the ground truth latents and hence can only be computed on synthetic toy datasets for which we have full knowledge about the data-generating process. In practice, this is not the case. Therefore, we defined a second group of metrics, including $\mathrm{Acc}(G, K)$ and $\mathrm{Acc}(C)$, that are applicable in the case where there is no access to the ground truth latents. We refer to appendix C.4 for a study on the relationship of $R^2(G)$ and $\mathrm{Acc}(G, K)$, which suggests we can use $\mathrm{Acc}(G, K)$ as a proxy of $R^2(G)$.

Table 1: Overview. We vary the group G used for the data-generating process, and consider a non-linear mixing through a neural network ("non-linear") as well as the infinite dSprites dataset. We report the R^2 on the equivariant part of the embedding, and the Accuracy (in %) on the invariant part of the embedding for identifying the content information.

Group (G)	SO_3			O_3			GL_3			$R_m \times \mathbb{Z}_n \times \mathbb{Z}_n$	
*Obs. (**f**)*	non-linear			non-linear			non-linear			indSprites	
Metric	$R^2(x)$	$R^2(G)$	Acc(C)	$R^2(x)$	$R^2(G)$	Acc(C)	$R^2(x)$	$R^2(G)$	Acc(C)	Acc(G, 5)	Acc(C)
InfoNCE	$0.0_{\pm 0.02}$	$0.0_{\pm 0.00}$	$98.9_{\pm 0.83}$	$0.0_{\pm 0.01}$	$0.0_{\pm 0.01}$	$99.1_{\pm 0.55}$	$0.1_{\pm 0.15}$	$0.0_{\pm 0.00}$	$98.5_{\pm 0.56}$	$0.36_{\pm 0.08}$	$99.97_{\pm 0.03}$
+LDS	$0.0_{\pm 0.03}$	$0.0_{\pm 0.00}$	$99.0_{\pm 0.51}$	$0.0_{\pm 0.11}$	$0.0_{\pm 0.00}$	$99.0_{\pm 0.65}$	$0.1_{\pm 0.06}$	$0.0_{\pm 0.00}$	$98.4_{\pm 0.59}$	$0.31_{\pm 0.03}$	$99.96_{\pm 0.04}$
+SLDS	$0.0_{\pm 0.01}$	$0.0_{\pm 0.00}$	$98.9_{\pm 0.79}$	$0.0_{\pm 0.03}$	$0.0_{\pm 0.01}$	$98.7_{\pm 0.96}$	$0.2_{\pm 0.24}$	$0.0_{\pm 0.00}$	$97.7_{\pm 0.93}$	$0.28_{\pm 0.02}$	$99.81_{\pm 0.27}$
EbC (lin.)	$70.9_{\pm 2.60}$	$54.1_{\pm 4.63}$	$20.2_{\pm 1.94}$	$70.8_{\pm 2.62}$	$54.0_{\pm 4.66}$	$19.8_{\pm 2.14}$	$59.7_{\pm 8.10}$	$39.8_{\pm 13.66}$	$22.3_{\pm 4.63}$	–	–
EbC	$99.7_{\pm 0.22}$	$99.7_{\pm 0.25}$	$99.1_{\pm 0.72}$	$99.8_{\pm 0.05}$	$99.7_{\pm 0.04}$	$99.2_{\pm 0.69}$	$99.8_{\pm 0.03}$	$99.7_{\pm 0.06}$	$98.5_{\pm 0.69}$	$99.91_{\pm 0.05}$	$74.04_{\pm 1.91}$

5 Empirical Results

We apply EbC on a variety of group learning settings, summarized in Table 1. EbC is effective at recovering group structure both in a setting where we have precise access of the latent space (for verification), and scales to benchmarking datasets used in objective-centric learning. Across all baselines, only EbC is able to recover the structure with high fidelity: We reach an R^2 of $> 99\%$ in recovering the ground truth latent space on synthetic data, validating Thm 1a and substantially outperforming a linear baseline (60–70%). This result verifies that indeed $\mathbf{h}(\boldsymbol{x}) = \boldsymbol{L}\boldsymbol{x}$ up to a linear indeterminacy in practice. We verify the group structure through an R^2 between the projected sample and the ground-truth for synthetic data with perfect recovery of >98% (Thm 1b). As a verification, we compare to existing contrastive learning baselines which are well established at learning *invariant* representations, i.e., the content. Indeed, all considered baselines are effective at recovering the content information with accuracies typically >98%. On synthetic data, EbC matches this performance (>98%); on idSprites we encounter a trade-off and accuracy drops to 75%.

EbC learns a representation of ($R_m \times \mathbb{Z}_n \times \mathbb{Z}_n$) from image data. We next evaluate EbC on idSprites. An optimal embedding space for the $R_m \times \mathbb{Z}_n \times \mathbb{Z}_n$ is a 3-torus, with the three individual groups represented along the three circular coordinates. Fig 4a depicts this (assumed) underlying structure of the latent space (which is not enforced in the real dataset or during training), along with the reconstructed embedding space of EbC. Note that the dimensions are related to each other; in Fig. 4b we show the dependency based on the y-, x- coordinate for a fixed angle. Qualitatively, this embedding space was stable also under variations of the content (shape and size of the object).

EbC learns faithful representations of content and style under diverse conditions. Next, we perform a fine-grained quantitative evaluation of the embedding space. In particular, we vary the properties of the indSprites dataset by varying the number of options for the content (shape and size) and the number of variations to learn from (translation and rotation) while keeping the overall dataset fixed. In Figure 4c we outline these options, reporting both the classification accuracy of the content, and the quality of the representation using our kNN metric. EbC recovers the group structure with high Top-5 accuracy close to 100% when performing an 1-over-20k kNN lookup for a range of number of contents. However, we notice that the content classification declines from around 80% to 20%. We report qualitative results of the KNN prediction in Appendix C.4.

EbC is robust to over-parameterization In Figure 4d, we vary the output dimension of the encoder (divided into 2 dimensions for the content, and 4–7 for the group). We observe that the group structure preserves a high predictive kNN above 99% and a stable content classification performance above 80%. However, this requires a sufficient number of samples for estimating the implicit representation: For $4\times$ the group dimensionality is required when single transitions are observed, and $8\times$ is required in the case of compound actions. Note that in practice, it is possible to even inform model hyperparameters on this metric, as it is available without any knowledge of the underlying ground truth structure of the data.

EbC identifies group representations of $O(n), GL(n)$, and $SO(n)$. Next, we investigate the applicability to more general group structures. For this, we generate additional synthetic data from more complex groups. $SO(n)$ is close to our example on idSprites, which is extended by $O(n)$ and $GL(n, R)$ as a fairly general and challenging example (Figure 5). Here we report results for $n = 3$,

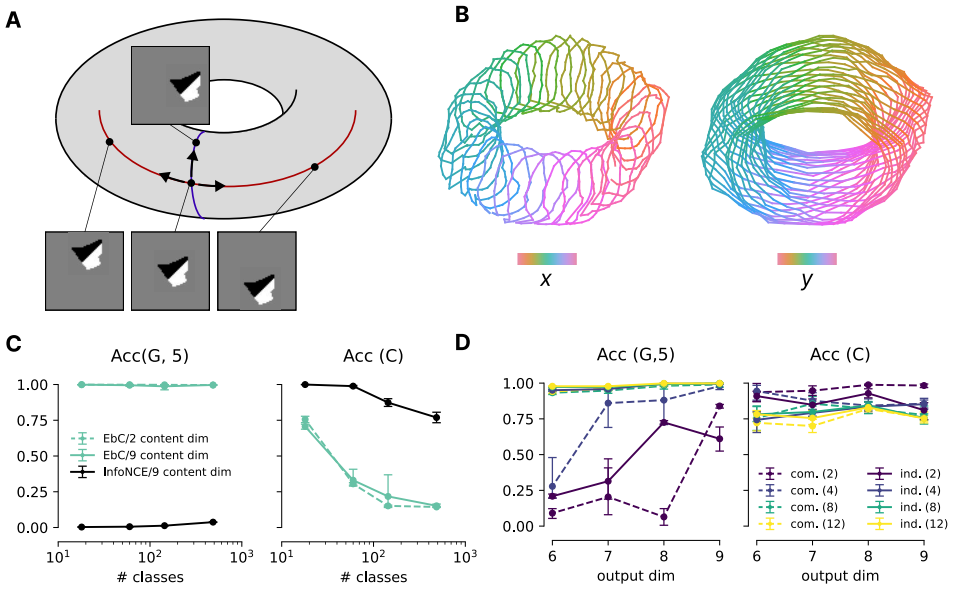

Figure 4: **EbC learns faithful representations of group actions**. A, a possible ground-truth latent space for idSprites. Depicted is $\mathbb{Z}_n \times \mathbb{Z}_n$. B, the actual embedding obtained by EbC, plotted for a fixed value of the orientation angle and colored in two views for variations in x- and y- direction. C, evaluation of representation quality and content accuracy across an increasing amount of classes. D, impact of observing the single groups (ind.) vs. a combination at each step in training (com.) across different output dimensions. Number in brackets is the number of samples in $\boldsymbol{Y}$ per output dim.

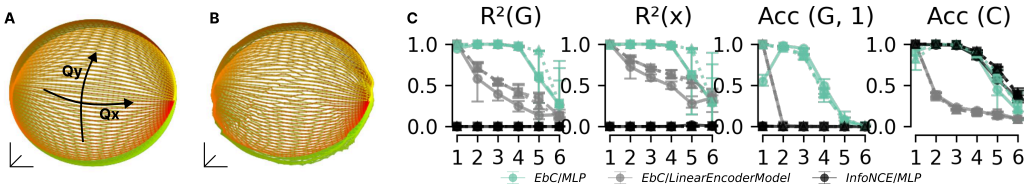

Figure 5: **Verification on simulated data**. A, traversals through the ground-truth latent space of our synthetic datasets. We manually constructed two orthogonal transformations $\boldsymbol{Q}_x, \boldsymbol{Q}_y$ perpendicular to each other to obtain a sphere. B, recovered latent space after non-linear demixing and optimal linear alignment. C, comparisons of EbC (green) against a linear baseline (gray) and an InfoNCE baseline (black) across an increasing number of mixing layers.

in Appendix C.1 we show additional results for higher dimensions. The complexity of the problem is additionally varied by generating increasingly non-linear datasets. This is expected to eventually degrade performance when a fixed-size dataset is used. We vary the number of mixing layers from n=2 to higher dimensions.

EbC is able to recover both a suitable vector space and an implicit group representation on these datasets with high fidelity. Figure 5a shows the ground truth latents, and Figure 5b shows a qualitative impression how the spherical latent space is recovered after unmixing. Quantatively, the R2(x) metric in Fig. 5c confirms the quality of this mapping up to n=4 mixing layers before we obtain degradation. Likewise, forward prediction through the group representation obtains high R2(G) up to 4 mixing layers, substantially outperforming the linear baseline. A relevant metric in practice is Acc(G) measuring forward prediction through a kNN based metric. In comparision to the InfoNCE baseline, we perform comparatively in identifying the class content across all group types (Acc(C)).

6 Further Analysis, Ablations, and Modeling Choices

Following experimental validation of our model and applicability to image data, we are interested in analyzing the empirical behavior of the loss function in more detail. In practice, an important

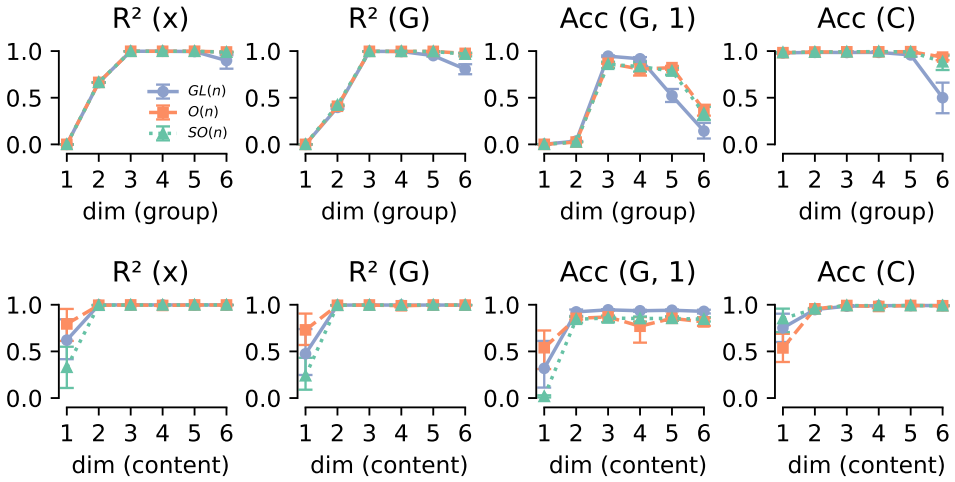

Figure 6: Robustness to model misspecification. In the ground-truth process, we allocate 3 dimensions to coordinates affected by the group action, and 3 dimensions to to content. Note, R^2 metrics reported here are not observable on real data, while Acc(G) is observable and Acc(C) is observable if a labeled test set is given.

hyperparameter is the parameterization of the embedding space, as well as the setup of the loss function. Further validation and ablation experiments can also be found in Appendix C.

Identification of the correct latent dimensionality In practice, the true dimensionality of the latent vector space is not known, or even ambiguous. Algorithms for learning group structure should be ideally (1) robust to model misspecification and (2) provide a clear hyperparameter validation protocol for choosing the correct dimensionality. In Fig. 6, we report the properties of EbC under model misspecification. Both for the content and groups, we consider a 3d ground truth latent space, resulting in a total of six latent factors. In Figure 6, second row, we show performance variations as we specify the wrong content dimension. Except for a degenerative case for d=1, we obtain close to perfect recovery of both group structure and content.

In the first row of Figure 6 we misspecify the group dimensionality, which is more critical. We see a clear peak in 1-NN lookup performance (Acc(G)) as well as saturation for R2(x)/R2(G) prediction performance for d>3. Very crucially, the practically *observable* metric for hyperparameter selection, Acc(G), shows a clear peak at the correct dimensionality (d=3), making it a prime target for model selection (which we indeed did in the experiments showed so far, cf. the discussion in Appendix B). As before, with the correctly selected dimension, we obtain a close-to-perfect performance in both recovery of the latent space and the representation of the group.

Analysis of optimal model parameters and hyperparameter selection With our fixed selection protocol, we finally consider model choices for the implementation of the algorithm outlined in Sec. 2. In particular, the choice of negatives for the set S is relevant for satisfying the diversity conditions required for Theorem 1. Furthermore, we can extend the loss function to consider the property of an inverse element, i.e., that both the action $\boldsymbol{R}(\boldsymbol{X}, \boldsymbol{X}')$ and $\boldsymbol{R}(\boldsymbol{X}', \boldsymbol{X})$ are valid representations. Note, we do not explicitly force the constraint that the respective matrices should be inverses of each other, making our approach still potentially applicable to structures not fulfilling all group axioms.

Table 2 depicts the different choices for both a symmetric loss and different negative distributions. On simpler groups, SO(n) and O(n), when our input vectors are normalized, the co-domain of the action will still be the sphere, making it less crucial to consider both $\boldsymbol{y}$ and $\boldsymbol{y}'$ as negative samples. In contrast, as we move to $GL(n)$, it is crucial to use both samples for training to avoid a performance drop. On O(n), GL(n), a symmetric loss additionally improves the performance.

7 Limitations

Accuracy trade-off for many classes While EbC is able to convincingly estimate both the latent space and the group actions, we could construct a failure case in idSprites when class dimensions

Table 2: Model variations across different loss functions, negative samples, and groups.

Sym.	S	SO_n		O_n		GL_n	
		Acc(G, 1)	Acc(G, 5)	Acc(G, 1)	Acc(G, 5)	Acc(G, 1)	Acc(G, 5)
✗	both	56.23 ±2.62	77.38 ±24.35	86.53 ±1.33	89.75 ±1.65	93.14 ±19.89	99.98 ±0.01
✗	$\boldsymbol{y}$	28.77 ±21.88	83.65 ±6.07	85.49 ±1.39	53.23 ±38.76	99.91 ±0.18	99.98 ±0.01
✗	$\boldsymbol{y}'$	73.44 ±4.40	79.52 ±18.17	87.08 ±1.49	95.91 ±1.49	96.83 ±9.15	99.98 ±0.01
✓	both	94.70 ±1.23	86.95 ±1.06	85.59 ±6.04	99.83 ±0.11	99.98 ±0.01	99.93 ±0.16
✓	$\boldsymbol{y}$	58.55 ±20.11	85.54 ±2.54	85.70 ±4.88	87.77 ±6.73	99.97 ±0.02	99.95 ±0.10
✓	$\boldsymbol{y}'$	94.44 ±1.06	84.31 ±6.99	83.61 ±7.88	99.84 ±0.09	99.89 ±0.25	99.85 ±0.35

increases to multiple hundreds or more. While we continue to estimate the group representation, the classification performance of the content decreases in these cases, even if the dimensionality of the latent space is matched to our baseline. A possible explanation is that EbC needs to implicitly learn a prototype of each class to estimate the coordinates affected by the group action, which might be harder to embed in the network than a classification objective alone.

Style vs. content subspaces identifiability Our proposed prior to separate content and style also has limitations. A clear separation can only be expected if the group dimensionality is chosen to be minimal. While we demonstrated that such a minimal dimensionality can be computed on real-world datasets, it comes with the additional computational burden of hyperparameter estimation, which might be prohibitive in practice. Future work might consider improved versions of our content/style subspace prior and provide theoretical guarantees on the optimal separation. Within the scope of this work, we showed a practical way to estimate the hyperparameters using the Acc(G) metric in the analysis section.

Real-world data and baselines Although we are the first to show that identifiable group representation learning from unlabeled observational data is feasible at all, our empirical analysis can be substantially extended. Firstly, on selected previous setups, more baseline comparisons could be conducted which is here limited due to the lack of extensive benchmarks. It will be particulary interesting to see how the algorithm scales to to full-scale image datasets like 3DIdent [56]. However, we do report additional experiments on real-world data in Appendix C.5.

Scaling properties, empirical vs. theoretical Our theoretical result mandates the use of $d + 1$ examples for estimation of the embedding space, and currently does not make a claim how convergence is influenced by limited data. In practice we observed that substantially more than $d + 1$ samples are required (about 6-8$\times$ gave good empirical results). In this light, our theoretical results can be extended to include bounds on the behavior with limited samples, although challenging and not every common in existing identifiability work. On the other hand, replacing our out-of-the-box least squares estimator by more stable and more adapted algorithms might further close the gap to the theoretical optimum. We report additional results on data efficiency in Appendix C.2.

8 Related Work

Learning equivariant representations Early work in equivariant representation learning focused on designing neural network architectures that are explicitly equivariant to a predefined class of transformations [3, 10, 27, 31, 42, 53]. These methods were powerful but require expert knowledge to design an inductive bias specific to particular groups. Inspired by the success of invariant self-supervised learning (I-SSL) methods [2, 13, 37, 54], a new paradigm emerged that sought to *learn* equivariant representations directly from data. The first wave of SSL-based methods assumed full knowledge of the group actions. Latent representations were learned either via encoder-only frameworks [4, 7, 12, 38] or via autoencoder-based frameworks [23, 24, 39]. Equivariance was encouraged either by predicting the parameters of the group action [4] or enforced by explicitly modeling the effects of group actions in latent space [7, 12, 23, 24, 38, 39]. The most recent advances tackle the more challenging problem of learning equivariance without explicit knowledge of the group actions. Both encoder-only approaches [15, 51, 52] and autoencoder-based approaches [28, 35, 36] have been proposed. The central innovation of these methods is that the representation of group actions $\boldsymbol{R}(g)$ is learned directly from data pairs $(\boldsymbol{y}, g \cdot \boldsymbol{y})$. For example, CARE [15] is an encoder-only, contrastive learning based approach with implicit $\boldsymbol{R}(g) \in O(n)$. STL [52] also uses an encoder-only

approach but enforces non-linear equivariant relationships $\phi(g \cdot \boldsymbol{y}) = \mu(\phi(\boldsymbol{y}), \boldsymbol{R}(g))$. NFT [28] adopts an auto-encoder framework with explicit linear group representations $\boldsymbol{R}(g) \in GL(n)$. While these methods mark important progress, they each leave open critical challenges: some are restricted to special classes of group representations, some require generative modeling of the input space, and none provide formal identifiability guarantees of the applied algorithm. While Koyama et al. [28] are the first to show that the problem itself is identifiable (i.e., there exists a non-trivial G-equivariant map $\phi : Y \mapsto X$ unique up to G-isomorphisms of the embedding space), they do not provide identifiability guarantees for the learning method itself.

Identifiable nonlinear ICA In nonlinear ICA, observed variables $\boldsymbol{y}$ are assumed to be generated from latent variables $\boldsymbol{x}$ via an unknown nonlinear "mixing" function $\mathbf{f}$. The goal is to recover $\mathbf{f}^{-1}$, or equivalently, estimate the latent variables $\boldsymbol{x}$ from the observed variables $\boldsymbol{y}$. Hyvärinen and Pajunen [20] showed that in contrast to linear ICA, the nonlinear ICA problem is, in general, *un*identifiable. Subsequent work has established that identifiability can be recovered by making additional assumptions about the data-generating process [18, 19, 21]. More recently, this theory has been connected to contrastive learning [40, 56], demonstrating that: (a) the choice of the conditional distribution of positive samples encodes the assumptions about the data-generating process, and (b) optimizing the InfoNCE loss [16, 37] recovers a solution to the nonlinear ICA problem. The identified representation is unique up to an indeterminacy that is directly determined by the assumed conditional distribution. Building on these advances, Laiz et al. [30] proposed DCL, which identifies latent representations under the assumption of linear or switching linear dynamical systems. Since G-equivariant representations also rely on a linear relationship between $(\boldsymbol{x}, \boldsymbol{x}')$ or $(\boldsymbol{x}_t, \boldsymbol{x}_{t+1})$) in sequential data, there exists a natural connection between DCL and the equivariant representation learning methods discussed above. Our work leverages this connection, combining the identifiability guarantees of nonlinear ICA with the goals of equivariant representation learning.

9 Conclusion

We proposed the first identifiable learning algorithm for group representations from unlabeled data without relying on generative models. We demonstrated, theoretically and empirically, that latent spaces of datasets with underlying group structure can be estimated from observational data. Crucially, the assumptions of our algorithms match datasets encountered in engineering and science. The supervisory signal merely requires knowledge that "the same" action is applied to a collection of samples. We confirmed that under these settings and mild overall assumptions on sufficient variety in the dataset, few of such pairs are sufficient to estimate the underlying ground truth latent space in which the group actions are faithfully represented as a linear transformation. We see wide applicability of this approach especially in computer vision (e.g., for figure-ground segmentation), robotics (for learning affordances and planning), and biology and biomedicine (for modeling the effects of longitudinal data and perturbation effects), among other fields.

Acknowledgments and Disclosure of Funding

We thank Susanne Keller for in-depth discussions about the model setup and related work. We thank Luisa Eck for discussions on the theory. We thank the anonymous reviewers and the area chair at NeurIPS 2025 for their constructive feedback to improve our manuscript. This work was supported by a Google PhD fellowship to StS. This work was supported by the German Research Foundation (DFG): SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms, project number: 276693517 and by Open Philanthropy Foundation funded by the Good Ventures Foundation. MB is a member of the Machine Learning Cluster of Excellence, funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645. This work was supported by the Helmholtz Association's Initiative and Networking Fund on the HAICORE@KIT and HAICORE@FZJ partitions.

Author contributions

TS: Methodology, Software, Investigation, Writing–Editing; **StS**: Conceptualization, Methodology, Formal analysis, Writing–Original Draft, **MB**: Conceptualization, Writing–Editing.

References

[1] James U. Allingham, Bruno K. Mlodozeniec, Shreyas Padhy, Javier Antorán, David Krueger, Richard E. Turner, Eric Nalisnick, and José M. Hernández-Lobato. A Generative Model of Symmetry Transformations. *Advances in Neural Information Processing Systems*, 37:91091–91130, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/a5a3b1ef79520b7cd122d888673a3ebc-Abstract-Conference.html.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[3] Taco Cohen and M. Welling. Group Equivariant Convolutional Networks. *ArXiv*, February 2016. URL https://www.semanticscholar.org/paper/Group-Equivariant-Convolutional-Networks-Cohen-Welling/fafcaf5ca3fab8dc4fad15c2391c0fdb4a7dc005.

[4] Rumen Dangovski, Li Jing, Charlotte Loh, Seungwook Han, Akash Srivastava, Brian Cheung, Pulkit Agrawal, and Marin Soljacic. Equivariant Self-Supervised Learning: Encouraging Equivariance in Representations. In *International Conference on Learning Representations*, October 2021. URL https://openreview.net/forum?id=gKLAAfiytI.

[5] Nima Dehmamy, Robin Walters, Yanchen Liu, Dashun Wang, and Rose Yu. Automatic Symmetry Discovery with Lie Algebra Convolutional Network. In *Advances in Neural Information Processing Systems*, volume 34, pages 2503–2515. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/148148d62be67e0916a833931bd32b26-Abstract.html.

[6] Emily L Denton and vighnesh Birodkar. Unsupervised learning of disentangled representations from video. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/2d2ca7eedf739ef4c3800713ec482e1a-Paper.pdf.

[7] Alexandre Devillers and Mathieu Lefort. EquiMod: An Equivariance Module to Improve Visual Instance Discrimination. In *The Eleventh International Conference on Learning Representations*, September 2022. URL https://openreview.net/forum?id=eDLwjKmtYFt.

[8] Sebastian Dziadzio, Çağatay Yıldız, Gido M van de Ven, Tomasz Trzciński, Tinne Tuytelaars, and Matthias Bethge. Infinite dsprites for disentangled continual learning: Separating memory edits from generalization. *arXiv preprint arXiv:2312.16731*, 2023.

[9] Cian Eastwood, Julius von Kügelgen, Linus Ericsson, Diane Bouchacourt, Pascal Vincent, Mark Ibrahim, and Bernhard Schölkopf. Self-Supervised Disentanglement by Leveraging Structure in Data Augmentations. In *Causal Representation Learning Workshop at NeurIPS 2023*, October 2023. URL https://openreview.net/forum?id=JoISqbH8vl.

[10] Marc Finzi, S. Stanton, Pavel Izmailov, and A. Wilson. Generalizing Convolutional Neural Networks for Equivariance to Lie Groups on Arbitrary Continuous Data. *ArXiv*, February 2020. URL https://www.semanticscholar.org/paper/Generalizing-Convolutional-Neural-Networks-for-to-Finzi-Stanton/563ca4cda06665f4b90f8fce9bcb28c02e6872b9.

[11] Sanket Gandhi, Atul, Samanyu Mahajan, Vishal Sharma, Rushil Gupta, Arnab Kumar Mondal, and Parag Singla. Learning disentangled representation in object-centric models for visual dynamics prediction via transformers. *CoRR*, abs/2407.03216, 2024. URL https://doi.org/10.48550/arXiv.2407.03216.

[12] Quentin Garrido, Laurent Najman, and Yann LeCun. Self-supervised learning of split invariant equivariant representations. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML'23*, pages 10975–10996, Honolulu, Hawaii, USA, July 2023. JMLR.org.

[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent a new approach to self-supervised learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, pages 21271–21284, Red Hook, NY, USA, December 2020. Curran Associates Inc. ISBN 978-1-71382-954-6.

[14] Andres D Grosmark and György Buzsáki. Diversity in neural firing dynamics supports both rigid and learned hippocampal sequences. *Science*, 351(6280):1440–1443, 2016.

[15] Sharut Gupta, Joshua Robinson, Derek Lim, Soledad Villar, and Stefanie Jegelka. Structuring Representation Geometry with Rotationally Equivariant Contrastive Learning. In *The Twelfth International Conference on Learning Representations*, October 2023. URL https://openreview.net/forum?id=lgaFMvZHSJ.

[16] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, March 2010. URL https://proceedings.mlr.press/v9/gutmann10a.html.

[17] Lukas Heumos, Anna C Schaar, Christopher Lance, Anastasia Litinetskaya, Felix Drost, Luke Zappia, Malte D Lücken, Daniel C Strobl, Juan Henao, Fabiola Curion, et al. Best practices for single-cell analysis across modalities. *Nature Reviews Genetics*, 24(8):550–572, 2023.

[18] Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in neural information processing systems*, 29, 2016.

[19] Aapo Hyvarinen and Hiroshi Morioka. Nonlinear ica of temporally dependent stationary sources. In *Artificial Intelligence and Statistics*, pages 460–469. PMLR, 2017.

[20] Aapo Hyvärinen and Petteri Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks: The Official Journal of the International Neural Network Society*, 12(3):429–439, April 1999. ISSN 1879-2782. doi: 10.1016/s0893-6080(98)00140-3.

[21] Aapo Hyvarinen, Hiroaki Sasaki, and Richard Turner. Nonlinear ica using auxiliary variables and generalized contrastive learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 859–868. PMLR, 2019.

[22] Aapo Hyvärinen, Ilyes Khemakhem, and Hiroshi Morioka. Nonlinear independent component analysis for principled disentanglement in unsupervised deep learning. *Patterns*, 4 (10):100844, October 2023. ISSN 26663899. doi: 10.1016/j.patter.2023.100844. URL https://linkinghub.elsevier.com/retrieve/pii/S2666389923002234.

[23] Yinzhu Jin, Aman Shrivastava, and P. T. Fletcher. Learning Group Actions on Latent Representations. *Advances in Neural Information Processing Systems*, 37:127273–127295, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e63309e532688c722177f81e99f94f32-Abstract-Conference.html.

[24] Hamza Keurti, Hsiao-Ru Pan, Michel Besserve, Benjamin F Grewe, and Bernhard Schölkopf. Homomorphism autoencoder–learning group structured representations from observed transitions. In *International Conference on Machine Learning*, pages 16190–16215. PMLR, 2023.

[25] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[26] David A. Klindt, Lukas Schott, Yash Sharma, Ivan Ustyuzhaninov, Wieland Brendel, Matthias Bethge, and Dylan Paiton. Towards nonlinear disentanglement in natural data with temporal sparse coding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=EbIDjBynYJ8.

[27] R. Kondor and Shubhendu Trivedi. On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups. *ArXiv*, February 2018. URL https://www.semanticscholar.org/paper/On-the-Generalization-of-Equivariance-and-in-Neural-Kondor-Trivedi/84032c19bad3493957d1319babd19bde2821fee3.

[28] Masanori Koyama, Kenji Fukumizu, Kohei Hayashi, and Takeru Miyato. Neural Fourier Transform: A General Approach to Equivariant Representation Learning. In *The Twelfth International Conference on Learning Representations*, October 2023. URL https://openreview.net/forum?id=eOCvA8iwXH.

[29] Lucas Laird, Circe Hsu, Asilata Bapat, and Robin Walters. MatrixNet: Learning over symmetry groups using learned group representations. *Advances in Neural Information Processing Systems*, 37:32512–32535, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/39137b5a573126103c8812dcdb9d0187-Abstract-Conference.html.

[30] Rodrigo González Laiz, Tobias Schmidt, and Steffen Schneider. Self-supervised contrastive learning performs non-linear system identification. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ONfWFluZBI.

[31] Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. In Michael A. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, page 3361. MIT Press, 1995.

[32] Dominik Lorenz, Leonard Bereska, Timo Milbich, and Bjorn Ommer. Unsupervised part-based disentangling of object shape and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

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

[34] Francesco Mezzadri. How to generate random matrices from the classical compact groups, 2007. URL https://arxiv.org/abs/math-ph/0609050.

[35] Thomas W. Mitchel, Michael Taylor, and Vincent Sitzmann. Neural Isometries: Taming Transformations for Equivariant ML. *Advances in Neural Information Processing Systems*, 37:7311–7338, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/0dbc204928f1e111aff6a8cb5b148151-Abstract-Conference.html.

[36] Takeru Miyato, Masanori Koyama, and Kenji Fukumizu. Unsupervised Learning of Equivariant Structure from Sequences. In *Advances in Neural Information Processing Systems*, October 2022. URL https://openreview.net/forum?id=7b7iGkuVqlZ.

[37] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[38] Jung Yeon Park, Ondrej Biza, Linfeng Zhao, Jan-Willem Van De Meent, and Robin Walters. Learning Symmetric Embeddings for Equivariant World Models. In *Proceedings of the 39th International Conference on Machine Learning*, pages 17372–17389. PMLR, June 2022. URL https://proceedings.mlr.press/v162/park22a.html.

[39] Guo-Jun Qi, Liheng Zhang, Feng Lin, and Xiao Wang. Learning Generalized Transformation Equivariant Representations Via AutoEncoding Transformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):2045–2057, April 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2020.3029801. URL https://ieeexplore.ieee.org/document/9219238.

[40] Geoffrey Roeder, Luke Metz, and Durk Kingma. On linear identifiability of learned representations. In *International Conference on Machine Learning*, pages 9030–9039. PMLR, 2021.

[41] Anastasiia V Sadybekov and Vsevolod Katritch. Computational approaches streamlining drug discovery. *Nature*, 616(7958):673–685, 2023.

[42] Victor Garcia Satorras, Emiel Hoogeboom, and M. Welling. E(n) Equivariant Graph Neural Networks. *ArXiv*, February 2021. URL https://www.semanticscholar.org/paper/E(n)-Equivariant-Graph-Neural-Networks-Satorras-Hoogeboom/8ea9cb53779a8c1bb0e53764f88669bd7edf38f0.

[43] Steffen Schneider, Jin Hwa Lee, and Mackenzie Weygandt Mathis. Learnable latent embeddings for joint behavioural and neural analysis. *Nature*, 617(7960):360–368, 2023.

[44] Steffen Schneider, Rodrigo González Laiz, Anastasiia Filippova, Markus Frey, and Mackenzie Weygandt Mathis. Time-series attribution maps with regularized contrastive learning. *arXiv preprint arXiv:2502.12977*, 2025.

[45] Anne E Urai, Brent Doiron, Andrew M Leifer, and Anne K Churchland. Large-scale neural recordings call for new insights to link brain and behavior. *Nature neuroscience*, 25(1):11–19, 2022.

[46] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[47] Julius Von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. *Advances in neural information processing systems*, 34: 16451–16467, 2021.

[48] Robin Winter, Marco Bertolini, Tuan Le, Frank Noé, and Djork-Arné Clevert. Unsupervised learning of group invariant and equivariant representations. *Advances in Neural Information Processing Systems*, 35:31942–31956, 2022.

[49] Tete Xiao, Xiaolong Wang, Alexei A. Efros, and Trevor Darrell. What Should Not Be Contrastive in Contrastive Learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=CZ8Y3NzuVzO.

[50] Jianke Yang, Robin Walters, Nima Dehmamy, and Rose Yu. Generative Adversarial Symmetry Discovery. In *Proceedings of the 40th International Conference on Machine Learning*, pages 39488–39508. PMLR, July 2023. URL https://proceedings.mlr.press/v202/yang23n.html.

[51] Thomas Yerxa, Jenelle Feather, Eero Simoncelli, and SueYeon Chung. Contrastive-Equivariant Self-Supervised Learning Improves Alignment with Primate Visual Area IT. *Advances in Neural Information Processing Systems*, 37:96045–96070, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ae28c7bc9414ffd8ffd2b3d454e6ef3e-Abstract-Conference.html.

[52] Jaemyung Yu, Jaehyun Choi, Dong-Jae Lee, HyeongGwon Hong, and Junmo Kim. Self-supervised Transformation Learning for Equivariant Representations. *Advances in Neural Information Processing Systems*, 37:83068–83090, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/972cd27c994a806e187ef1c2f5254059-Abstract-Conference.html.

[53] Manzil Zaheer, Satwik Kottur, Siamak Ravanbhakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J Smola. Deep sets. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3394–3404, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[54] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. In *Proceedings of the 38th International Conference on Machine Learning*, pages 12310–12320. PMLR, July 2021. URL https://proceedings.mlr.press/v139/zbontar21a.html.

[55] Ding Zhou and Xue-Xin Wei. Learning identifiable and interpretable latent models of high-dimensional neural activity using pi-VAE. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7234–7247. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/510f2318f324cf07fce24c3a4b89c771-Paper.pdf.

[56] Roland S. Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12979–12990. PMLR, 2021.

Supplementary Material

A Proof of Theorem 1

In this chapter, we provide a proof for Theorem 1 in the main paper. We assume the following data generating process:

Definition 1 (Data generating process)**.** Let $\boldsymbol{x} \in V \subseteq \mathbb{R}^d$ be a vector describing the ground truth latent components, let g be an element of a group G, and let $\mathbf{f} : V \to \mathbb{R}^D$ be an injective map. Assume that for each group element g, we have at least M pairs of samples $(\boldsymbol{y}_j, \boldsymbol{y}'_j)$ with

$$\boldsymbol{y}_i = \mathbf{f}(\boldsymbol{x}_i), \qquad \boldsymbol{y}'_i = \mathbf{f}(\boldsymbol{R}(g)\boldsymbol{x}_i) \qquad i = 1, \dots, M \tag{15}$$

where $\boldsymbol{R} : G \to \mathrm{GL}(d, \mathbb{R})$ is a representation of G on V.

If we think of $\boldsymbol{x}$ containing an invariant part (the content), the theory by Von Kügelgen et al. [47] already covers the case for identifying the content component of $\boldsymbol{x}$ when considering the group actions as augmentations. We will therefore not explicitly discuss the discovery of the invariant part of the embedding, but focus our following statements on learning equivariant representations.

We extend the canonical discriminative form introduced by Roeder et al. [40] to arrive at the form

$$p_{\boldsymbol{u},\boldsymbol{v},\alpha,\beta}(\boldsymbol{y} \mid \boldsymbol{x}, S) = \frac{\exp\left(\boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{v}(\boldsymbol{y}) + \alpha(\boldsymbol{x}) + \beta(\boldsymbol{y})\right)}{\sum_{\boldsymbol{y}' \in S} \exp\left(\boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{v}(\boldsymbol{y}') + \alpha(\boldsymbol{x}) + \beta(\boldsymbol{y}')\right)}. \tag{16}$$

The functions $\boldsymbol{u}, \boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^d$, $\alpha, \beta : \mathbb{R}^d \to \mathbb{R}$ can have shared underlying structure (e.g., are parameterized by the same neural network) and are not necessarily independent. We extend the diversity conditions [40]:

Definition 2 (Diversity Conditions, adapted from Roeder et al. [40])**.** Assume a dataset of tuples $(\boldsymbol{x}, \boldsymbol{y}, S)$ and let $p_{\mathrm{data}}(\boldsymbol{x}, \boldsymbol{y}, S)$ be a distribution over this (possibly discrete) dataset. We then require the following properties:

1. For any $\boldsymbol{x}, S$ we can find at least N pairs of points $(\boldsymbol{y}_A, \boldsymbol{y}_B)$ with $p_{\mathrm{data}}(\boldsymbol{x}, \boldsymbol{y}_A, S) > 0$ and $p_{\mathrm{data}}(\boldsymbol{x}, \boldsymbol{y}_B, S) > 0$ such that the finite differences $\boldsymbol{v}(\boldsymbol{y}^i_A) - \boldsymbol{v}(\boldsymbol{y}^i_B)$ are linearly independent.
2. For any $\boldsymbol{y}$, we can find at least N pairs of points $\boldsymbol{x}_A, \boldsymbol{x}_B$ and their sets of negative samples S_A, S_B such that $\boldsymbol{y} \in S_A \cap S_B$ is in the negative sets for both examples, and the finite differences $\boldsymbol{u}(\boldsymbol{x}^i_A) - \boldsymbol{u}(\boldsymbol{x}^i_B)$ are linearly independent.

Both assumptions need to be assured through design of the sampling procedure and initialization of the networks. In particular, the first condition requires that sufficient variation is present among negative examples. Similar to Roeder et al. [40], since $\boldsymbol{v}$ is a randomly initialized neural network, it is expected that the variability is met as long as variability in the ground truth factors is given. The second condition requires that negative samples sufficiently often co-occur with enough variation in the reference examples. In addition, this condition requires $\hat{\boldsymbol{R}}$ to be full-rank, which we ensure by adding at least d samples for estimating a $d \times d$ matrix.

We can now re-state the theorem and discuss the proof strategy. Please note that we relate the embedding space according to the commutative diagram in Figure 1: We consider the relations between samples in the original vector space V vs. the recovered factor space, which is the co-domain of first applying the mixing and then the de-mixing functions, $\mathbf{h} := \phi \circ \mathbf{f}$, $\mathbf{h} : V \to V$. We can state:

Theorem 1 (Identifiability of the Group Representation)**.** *Assume the following:*

(1) $\mathbf{f}$ *is an injective mixing function,* $g \in G$ *is a group element according to Def. 3.*
(2) The model $\phi : \mathbb{R}^D \to V$ *satisfies the diversity conditions in Def. 4.*
(3) $\boldsymbol{R}$ *is a representation of* G*, and* $\hat{\boldsymbol{R}}$ *is the implicit representation of the group (Eq. 2) such that* $\hat{\boldsymbol{R}}(\boldsymbol{X}, g\boldsymbol{X}) = \boldsymbol{R}(g)$ *for pairs of transformed samples* $(\boldsymbol{X}, g\boldsymbol{X})$ *and group actions* $g \in G$*.*

Define $\mathbf{h} := \phi \circ \mathbf{f}$*. Then, for matching conditional distributions* $p_\phi = p_{\mathbf{f}^{-1}}$*, for all points* $\boldsymbol{x}$ *and group actions* g *in the dataset:*

(a) We recover the original vector space $\mathbf{h}(\boldsymbol{x}) = \boldsymbol{L}\boldsymbol{x}$*, up to an ambiguity* $\boldsymbol{L} \in GL(d)$*.*
(b) We recover a representation of the group, $\hat{\boldsymbol{R}}(\mathbf{h}(\boldsymbol{X}), \mathbf{h}(g\boldsymbol{X})) = \boldsymbol{L}\boldsymbol{R}(g)\boldsymbol{L}^{-1}$*.*

As outlined in the main paper, the proof proceeds as follows: First, we extend the canonical discriminative form of Roeder et al. [40] towards a more general setting, giving affine instead of linear identifiability (§A.1). Then, we show that our implicit group representation admits (§A.2). Finally, we leverage these results to arrive at the indeterminacies reported in the theorem (§A.3).

A.1 Extended canonical discriminative form

The following proposition is only a slight modification from Eq. 1 in Roeder et al. [40]. The key difference is the introduction of the scalar potential functions α and β, making the canonical discriminative form more flexible. In particular, it now admits a mean squared error loss function.

Under these assumptions, we can state:

Theorem 2 (Generalized Canonical Discriminative Form)**.** *Let $\boldsymbol{u}, \boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^d$, $\alpha, \beta : \mathbb{R}^d \to \mathbb{R}$ be functions satisfying Def. 2. In particular, $\boldsymbol{u}, \boldsymbol{v}, \alpha, \beta$ can have shared underlying structure and are not necessarily independent. Then, the probabilistic model of the form*

$$p_{\boldsymbol{u},\boldsymbol{v},\alpha,\beta}(\boldsymbol{y} \mid \boldsymbol{x}, S) = \frac{\exp\left(\boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{v}(\boldsymbol{y}) + \alpha(\boldsymbol{x}) + \beta(\boldsymbol{y})\right)}{\sum_{\boldsymbol{y}' \in S} \exp\left(\boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{v}(\boldsymbol{y}') + \alpha(\boldsymbol{x}) + \beta(\boldsymbol{y}')\right)} \tag{17}$$

with $\boldsymbol{y} \in S$ is identifiable up to an affine indeterminacy, i.e., for two models $(\boldsymbol{u}', \boldsymbol{v}', \alpha', \beta')$ and $(\boldsymbol{u}^*, \boldsymbol{v}^*, \alpha^*, \beta^*)$ and their respective distributions (p', p^*), we have for all $\boldsymbol{x}, \boldsymbol{y}$ with $p_{\text{data}}(\boldsymbol{x}, \boldsymbol{y}, S) > 0$,

$$p' = p^* \implies \begin{cases} \boldsymbol{u}'(\boldsymbol{x}) & = \boldsymbol{A}\boldsymbol{u}^*(\boldsymbol{x}) + \boldsymbol{c} \\ \boldsymbol{v}'(\boldsymbol{y}) & = \boldsymbol{B}\boldsymbol{v}^*(\boldsymbol{y}) + \boldsymbol{d} \end{cases} \tag{18}$$

for two invertible $d \times d$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ and vectors $\boldsymbol{c}, \boldsymbol{d} \in \mathbb{R}^d$.

Proof. The proof technique adopts the strategy from Roeder et al. [40] for the indeterminacy of $\boldsymbol{u}$, and re-applies this to the indeterminacy of $\boldsymbol{v}$. The approach considers finite differences on the log probabilities for p, p^* with

$$\log p(\boldsymbol{y}|\boldsymbol{x}, S) = \boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{v}(\boldsymbol{y}) + \alpha(\boldsymbol{x}) + \beta(\boldsymbol{y}) - Z(\boldsymbol{x}, S) \tag{19}$$

$$= \boldsymbol{u}(\boldsymbol{x})^\top \boldsymbol{v}(\boldsymbol{y}) + \tilde{\alpha}(\boldsymbol{x}, S) + \beta(\boldsymbol{y}). \tag{20}$$

Part 1: Affine identifiability of $\boldsymbol{u}$. For any $\boldsymbol{x}$ in the data distribution, and corresponding S, select two points $\boldsymbol{y}_A, \boldsymbol{y}_B \in S$. For the points, it holds that

$$\log p(\boldsymbol{y}_A|\boldsymbol{x}, S) - \log p(\boldsymbol{y}_B|\boldsymbol{x}, S) = \log p^*(\boldsymbol{y}_A|\boldsymbol{x}, S) - \log p^*(\boldsymbol{y}_B|\boldsymbol{x}, S) \tag{21}$$

Let $\Delta\boldsymbol{v}_{AB} := \boldsymbol{v}(\boldsymbol{y}_A) - \boldsymbol{v}(\boldsymbol{y}_B)$, $\Delta\beta_{AB} := \tilde{\beta}(\boldsymbol{y}_A) - \beta(\boldsymbol{y}_B)$ to obtain

$$\Delta\boldsymbol{v}_{AB}^\top \boldsymbol{u}(\boldsymbol{x}) + \Delta\beta_{AB} = (\Delta\boldsymbol{v}_{AB}^*)^\top \boldsymbol{u}^*(\boldsymbol{x}) + \Delta\beta_{AB}^* \tag{22}$$

By assumption (1), we can find N such pairs with linearly independent $\Delta\boldsymbol{v}_{AB}$, $\Delta\boldsymbol{v}_{AB}^*$, which lets us re-write the equation using the full-rank matrices $\boldsymbol{L}, \boldsymbol{L}^*$

$$\boldsymbol{L}^\top \boldsymbol{u}(\boldsymbol{x}) + \boldsymbol{c} = (\boldsymbol{L}^*)^\top \boldsymbol{u}^*(\boldsymbol{x}) + \boldsymbol{c}^* \tag{23}$$

$$\boldsymbol{u}(\boldsymbol{x}) = (\boldsymbol{L}^* \boldsymbol{L}^{-1})^\top \boldsymbol{u}^*(\boldsymbol{x}) + (\boldsymbol{c}^* - \boldsymbol{c}) \tag{24}$$

which requires the map from $\boldsymbol{u}$ to $\boldsymbol{u}^*$ to be affine, concluding the first part of the proof.

Part 2: Affine identifiability of $\boldsymbol{v}$. For a given $\boldsymbol{y}$, find $\boldsymbol{x}_A, \boldsymbol{x}_B, S_A, S_B$ such that $\boldsymbol{y} \in S_A \cap S_B$, and note that $S_A = S_B$ is possible but not required. Then we consider the finite differences:

$$\log p(\boldsymbol{y}|\boldsymbol{x}_A, S_A) - \log p(\boldsymbol{y}|\boldsymbol{x}_B, S_B) = \log p^*(\boldsymbol{y}|\boldsymbol{x}_A, S_A) - \log p^*(\boldsymbol{y}|\boldsymbol{x}_B, S_B) \tag{25}$$

Let $\Delta\boldsymbol{u}_{AB} := \boldsymbol{u}(\boldsymbol{x}_A) - \boldsymbol{u}(\boldsymbol{x}_B)$, $\Delta\tilde{\alpha}_{AB} := \tilde{\alpha}(\boldsymbol{x}_A, S) - \tilde{\alpha}(\boldsymbol{x}_B, S)$ to obtain

$$\Delta\boldsymbol{u}_{AB}^\top \boldsymbol{v}(\boldsymbol{y}) + \Delta\tilde{\alpha}_{AB} = (\Delta\boldsymbol{u}_{AB}^*)^\top \boldsymbol{v}^*(\boldsymbol{y}) + \Delta\tilde{\alpha}_{AB}^* \tag{26}$$

Stacking multiple conditions gives

$$\boldsymbol{L}^\top \boldsymbol{v}(\boldsymbol{y}) + \boldsymbol{c} = (\boldsymbol{L}^*)^\top \boldsymbol{v}^*(\boldsymbol{y}) + \boldsymbol{c}^* \tag{27}$$

$$\boldsymbol{v}(\boldsymbol{y}) = (\boldsymbol{L}^* \boldsymbol{L}^{-1})^\top \boldsymbol{v}^*(\boldsymbol{y}) + (\boldsymbol{c}^* - \boldsymbol{c}) \tag{28}$$

which requires the map from $\boldsymbol{v}$ to $\boldsymbol{v}^*$ to be affine, concluding the proof. □

A.2 Properties of the implicit group representations

We next consider properties of the implicit group representation defined in Sec. 2, given by

$$\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = \min_{\boldsymbol{R} \in \mathrm{GL}(d)} \|\boldsymbol{X}' - \boldsymbol{X}\boldsymbol{R}^\top\|_F^2 \Leftrightarrow \hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = (\boldsymbol{X}^\top \boldsymbol{X})^{-1}(\boldsymbol{X}^\top \boldsymbol{X}'). \tag{29}$$

Lemma 1 (Equivariance of implicit group representations)**.** *Let $\boldsymbol{X}, \boldsymbol{X}' \in \mathbb{R}^{m \times n}$, $m \geq n$ have rank n. Define $\hat{\boldsymbol{R}} : \mathbb{R}^{m \times n} \times \mathbb{R}^{m \times n} \to \mathbb{R}^{n \times n}$ as*

$$\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = \arg\min_{\boldsymbol{L}} \|\boldsymbol{X}' - \boldsymbol{X}\boldsymbol{L}^\top\|_F^2 \tag{30}$$

and let $\mathbf{f}(\boldsymbol{X}) = \boldsymbol{X}\boldsymbol{A}^\top + \mathbf{1}_m \boldsymbol{b}^\top$ be an invertible affine transform applied to each row of $\boldsymbol{X}$, then we have

$$\hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}')) = \boldsymbol{A}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{A}^{-1}. \tag{31}$$

Proof. The solution to the least squares problem can be re-written using the normal equations,

$$\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = \arg\min_{\boldsymbol{L}} \|\boldsymbol{X}' - \boldsymbol{X}\boldsymbol{L}^\top\|_F^2 \quad \Leftrightarrow \quad \left(\boldsymbol{X}^\top \boldsymbol{X}\right) \hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')^\top = \boldsymbol{X}^\top \boldsymbol{X}' \tag{32}$$

Let $\tilde{\boldsymbol{X}} = \boldsymbol{X}\boldsymbol{A}^\top$ and $\tilde{\boldsymbol{X}}' = \boldsymbol{X}'\boldsymbol{A}^\top$. Then, inserting the definition of $\mathbf{f}$ gives

$$\hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}')) = \arg\min_{\boldsymbol{L}} \|\tilde{\boldsymbol{X}}' + \mathbf{1}_m \boldsymbol{b}^\top - \tilde{\boldsymbol{X}}\boldsymbol{L}^\top - \mathbf{1}_m \boldsymbol{b}^\top\|_F^2 \tag{33}$$

$$= \arg\min_{\boldsymbol{L}} \|\tilde{\boldsymbol{X}}' - \tilde{\boldsymbol{X}}\boldsymbol{L}^\top\|_F^2 \tag{34}$$

which is equivalent to solving the normal equations

$$\left(\tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}}\right) \hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}'))^\top = \tilde{\boldsymbol{X}}^\top \tilde{\boldsymbol{X}}' \tag{35}$$

Substituting back in terms of $\boldsymbol{X}$ and $\boldsymbol{A}$:

$$\boldsymbol{A}\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{A}^\top \hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}'))^\top = \boldsymbol{A}\boldsymbol{X}^\top \boldsymbol{X}'\boldsymbol{A}^\top \tag{36}$$

$\boldsymbol{A}$ is invertible by assumption and it follows

$$\boldsymbol{X}^\top \boldsymbol{X} \left[\boldsymbol{A}^\top \hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}'))^\top \boldsymbol{A}^{-\top}\right] = \boldsymbol{X}^\top \boldsymbol{X}' \tag{37}$$

Note that $\boldsymbol{X}^\top \boldsymbol{X} \in \mathbb{R}^{n \times n}$ has full rank by the assumption on $\boldsymbol{X}$. Then, we get

$$\boldsymbol{A}^\top \hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}'))^\top \boldsymbol{A}^{-\top} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{X}' = \hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')^\top \tag{38}$$

Taking the transpose and re-arranging yields

$$\hat{\boldsymbol{R}}(\mathbf{f}(\boldsymbol{X}), \mathbf{f}(\boldsymbol{X}')) = \boldsymbol{A}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{A}^{-1} \tag{39}$$

concluding the proof. □

A.3 Proof of Theorem 1

Proof. We have $p_\phi = p_{\mathbf{f}^{-1}}$. After rewriting Eq. 3 in terms of $\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{X}'$ instead of $\boldsymbol{y} = \mathbf{f}(\boldsymbol{x}), \boldsymbol{Y} = \mathbf{f}(\boldsymbol{X}), \boldsymbol{Y}' = \mathbf{f}(\boldsymbol{X}')$, our model follows the generalized canonical discriminative form (Proposition 2) with the following parametrization for the data likelihood $p_{\mathbf{f}^{-1}}$:

$$\boldsymbol{u}^*(\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{X}') = \hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{x} = \boldsymbol{R}(g)\boldsymbol{x} \tag{40}$$

$$\boldsymbol{v}^*(g\boldsymbol{x}) = g\boldsymbol{x}. \tag{41}$$

To specify the model likelihood p_ϕ, we introduce the shorthand $\mathbf{h} := \phi \circ \mathbf{f}$ which maps samples from the ground truth latent space to the recovered latent space (e.g., the composition of the data generating process and feature encoder) and obtain

$$\boldsymbol{u}'(\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{X}') = \hat{\boldsymbol{R}}(\mathbf{h}(\boldsymbol{X}), \mathbf{h}(\boldsymbol{X}'))\mathbf{h}(\boldsymbol{x}) \tag{42}$$

$$\boldsymbol{v}'(\boldsymbol{x}') = \mathbf{h}(\boldsymbol{x}') \tag{43}$$

By assumption $p_\phi = p_{\mathbf{f}^{-1}}$. From Proposition 2 it then follows that

$$\boldsymbol{u}'(\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{X}') = \boldsymbol{A}\boldsymbol{u}^*(\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{X}') + \boldsymbol{c} \tag{44}$$

$$\boldsymbol{v}'(\boldsymbol{x}') = \boldsymbol{B}\boldsymbol{v}^*(\boldsymbol{x}') + \boldsymbol{d}. \tag{45}$$

where $\boldsymbol{A}, \boldsymbol{B} \in \mathrm{GL}(d)$ and $\boldsymbol{c}, \boldsymbol{d} \in \mathbb{R}^d$. Inserting model and data generating process yields

$$\mathbf{h}(\boldsymbol{x}') = \boldsymbol{B}\boldsymbol{x}' + \boldsymbol{d} \tag{46}$$

$$\hat{\boldsymbol{R}}(\mathbf{h}(\boldsymbol{X}), \mathbf{h}(\boldsymbol{X}'))\mathbf{h}(\boldsymbol{x}) = \boldsymbol{A}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{x} + \boldsymbol{c} \tag{47}$$

Since $\mathbf{h}$ is affine, we insert the first into the second equation and invoke Lemma 1 to arrive at

$$\boldsymbol{B}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{B}^{-1}(\boldsymbol{B}\boldsymbol{x} + \boldsymbol{d}) = \boldsymbol{A}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{x} + \boldsymbol{c} \tag{48}$$

and re-arranging yields

$$\boldsymbol{B}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{x} + \boldsymbol{B}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{d} = \boldsymbol{A}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{x} + \boldsymbol{c} \tag{49}$$

$$(\boldsymbol{B} - \boldsymbol{A})\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{x} = \boldsymbol{c} - \boldsymbol{B}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{d} \tag{50}$$

This is an equation of the form $\boldsymbol{U}\boldsymbol{x} = \boldsymbol{v}$. Assuming that we observe $\boldsymbol{x}$ such that matrix collecting all $\boldsymbol{x}$ has full rank, i.e. all latent dimensions vary, it follows that $\boldsymbol{U} = 0$, $\boldsymbol{v} = 0$, hence

$$(\boldsymbol{B} - \boldsymbol{A})\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = 0 \tag{51}$$

$$\boldsymbol{B}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{d} = \boldsymbol{c} \tag{52}$$

and inserting $\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}') = \boldsymbol{R}(g)$ gives

$$(\boldsymbol{B} - \boldsymbol{A})\boldsymbol{R}(g) = 0 \tag{53}$$

$$\boldsymbol{c} = \boldsymbol{B}\boldsymbol{R}(g)\boldsymbol{d} \tag{54}$$

Since $\boldsymbol{R}(g)$ has full rank, from the first equation it follows that $\boldsymbol{A} = \boldsymbol{B}$. For the second equation, since the left hand side is independent of g and all matrices have full rank, we can only admit the trivial solution where $\boldsymbol{c} = \boldsymbol{d} = 0$. It follows that

$$\mathbf{h}(\boldsymbol{x}) = \boldsymbol{B}\boldsymbol{x} \quad \forall \boldsymbol{x} \in S \tag{55}$$

$$\hat{\boldsymbol{R}}(\mathbf{h}(\boldsymbol{X}), \mathbf{h}(\boldsymbol{X}')) = \boldsymbol{B}\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')\boldsymbol{B}^{-1} \tag{56}$$

for an invertible matrix $\boldsymbol{B}$, which concludes the proof. □

B Additional Experimental Details

In this section, we provide further details on the algorithm and other experimental details. As an extension of this section, we would like to point to our figure and results repository (https://github.com/dynamical-inference/NeurIPS2025_Schmidt) as well as to our code repository (https://github.com/dynamical-inference/ebc) as the reference for all experimental details. The figure repository contains the code to plot the figures, but more importantly, it also contains all the raw results and all training and data generation parameters of all model runs that went into the respective figures. Each of those parameters can easily be traced back to their definition in the main code base repository.

B.1 Experimental protocol

Here we describe the default dataset generation and training protocol of our experiments first, and then provide additional details for any deviation from these default parameters that may exist in any of the figures.

B.1.1 Dataset Generation

Synthetic Group Data We consider three types of synthetic datasets following the data generating process of Theorem 1. Group elements are taken from a subgroup of the special orthogonal group $SO(n)$, the orthogonal group $O(n)$ and the general linear group $GL(n)$:

1. We sample a random group elements $\boldsymbol{R}(g) \sim p_G(g)$ from $G \in \{SO(n), O(n), GL(n)\}$.
2. We sample m (with $m > n$) random vectors $\boldsymbol{x} \in \mathbb{S}^n$ from the n-dimensional unit sphere.
3. We sample the content component $\boldsymbol{c} \in C$ from a finite set of vectors $C \subset S^d$.
4. From $(\boldsymbol{R}(g), \{\boldsymbol{x}_i\}^m, \boldsymbol{c})$ we generate $\boldsymbol{Y} = \{\boldsymbol{y}_i, \boldsymbol{y}_i'\}^m$, where $\boldsymbol{y}_i = \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{c})$ and $\boldsymbol{y}_i' = \mathbf{f}(\boldsymbol{R}(g)\boldsymbol{x}_i, \boldsymbol{c})$.
5. We repeat this sampling procedure until we reach the desired dataset size.

To sample from the $O(n)$ and $SO(n)$ group, we sample from the Haar distribution [34], making use of the SciPy [46] package. To sample from $GL(n)$ we implement a rejection sampling procedure. We first generate a random $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ matrix with values $a_{i,j}$ from a standard normal distribution. Then we compute the determinant $det(\boldsymbol{A})$ & the "identity error" $I_{err} = |\boldsymbol{I} - AA^{-1}|^F$ and we reject A if $|det(A)| < 0.2$ or $det(A) > 10$ or $I_{err} > 10^{-3}$, otherwise we accept. The nonlinear injective mixing function $\mathbf{f}$ is parameterized by a random 3-layer MLP and sampled via rejection sampling to fulfill the injectivity assumption as is done in [18, 30]. The MLP is followed by a random linear map to a 50-dimensional space. For the full dataset, we sample a maximum of 1000 matrices $\boldsymbol{R}(g)$ and a total of 1M pairs $(\boldsymbol{y}, \boldsymbol{y}')$. Unless stated otherwise, we choose $n = 3$, $d = 3$, $|C| = 100$.

Infinite dSprites We leverage infinite dSprites [8, idSprites], an extension of dSprites [33] for validation on more diverse visual data. infinite dSprites is an extension of the original dSprites dataset, which allows to generate variations of dSprites dataset. We subsample all images to a resolution of 64×64. The dataset allows rich variations of the content and ensures that the resulting objects do not have any symmetries. By default we use the same number of varying factors as in dSprites, namely 3 random shapes of 6 different sizes. For our purposes we consider the combination of shape and size the content. Each of these objects may then be oriented in 40 different ways or have one of 32×32 x or y position translations. We consider the orientations and translations to be our groups of interest and sample the data such that we get closed groups. By default, we sample the dataset such that any paired sample $(\boldsymbol{y}, \boldsymbol{y'})$ is undergoing a combination of these three types of transformations.

B.1.2 Dataset Split

Before training on the synthetic group dataset, we perform a hold-out split in terms of the group actions g_i into a training, validation, and test dataset such that we get a 80/10/10 split of the group actions. For idSprites, we randomly sample approx. 20k group actions during training and sample another approx. 90k group actions for validation and test respectively. When computing metrics that require to fit another model, such as a KNN, Linear Regression or Logistic Regression model, we split the validation or test dataset again 50/50 and report the metric on the hold-out set.

B.1.3 Model Training

Estimating $\boldsymbol{R}(g)$ To estimate $\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')$ we use $k \cdot n$ additional samples where n is the assumed group dimensionality of the embedding space and k is the samples per action factor. We fit $\hat{\boldsymbol{R}}(\boldsymbol{X}, \boldsymbol{X}')$ via linear least squares (PyTorch's "gels" solver). By default, we do not compute gradients through the linear least squares estimation to save on compute time. By default for idSprites we use $k = 12$ and for the synthetic group dataset we use $k = 4$.

Encoder Model For the feature encoder ϕ we use an MLP with three layers and hidden unit size of 512 for idSprites and 128 for the synthetic group dataset. The output dimension of the MLP is $n + d$ where n is the assumed group dimensionality of the embedding space and d the assumed content dimensionality. Before passing the idSprites images to the MLP we scale them to $[-1, 1]$ and flatten the image. By default for idSprites we assume group dimensionality $n = 7$ and content dimensionality $d = 2$. For the synthetic group dataset we use the same dimensionalities n and d as are used for the data generation.

Loss Function & Optimizer As loss we minimize the negative mean likelihood (Eq. 3) and for each sample compute the loss in both directions $L(\boldsymbol{y}, \boldsymbol{y}')$ and $L(\boldsymbol{y}', \boldsymbol{y})$. As negatives S, we sample uniformly from all available samples in the dataset. We use 1024 positive pairs and 16k (2^{12}) negative samples in each batch and train each model for 20k iterations. We train the encoder via gradient descent using the Adam optimizer [25] with a learning rate of 1×10^{-3}.

Error bounds For our main results, to compute error bounds or standard deviation across our metrics, we generate each dataset with three different seeds and fit each model three times with different seeds during training. For variations and ablation experiments extending our main results, we only fit a single model across the three different dataset seeds. We refer to the Figure & Results repository for details on the exact number of seeds for every experiment.

B.2 Hyperparameter Selection

We use the $Acc(G, 1)$ metric for any and all hyperparameter selection, since this is an observable metric and a good proxy for the identifiability metrics $R^2(G)$ (see Figure 8) and, by extension, for $R^2(x)$. Additionally, when we perform explicit hyperparameter selection and do not present results from experimental variations or ablation studies, we display the metrics computed on the test set instead of the metrics from the validation set, which were used for hyperparameter selection. Otherwise, we show the metrics on the validation set. Again, we refer to the Figure and Results repository for more details.

B.3 Third-Party License Information

We used the infinite dSprites (idSprites; 8) dataset available at `https://github.com/sbdzdz/idsprites` in the pip-installable version v1.0.1 (MIT License).

B.4 Compute Resources

Experiments were carried out on a compute cluster with A100 cards with 40Gb VRAM. On each card, we ran 2–6 experiments simultaneously, depending on the dataset size. The run time for individual experiments trained for 20k steps varied between approximately 10 minutes and one hour depending on the experiment configuration, the most important factors that impact the train time being batch sizes, specifically the number of negative samples, and the $k \cdot n$ number of samples used to fit the linear least squares estimator.

C Additional Experimental Results

C.1 Recovery of group structure for $n > 3$.

We extend our results from the paper towards higher dimensions, and consider $SO(n), O(n), GL(n)$ for $n = 3, 5, 7, 9$. Results are depicted in Fig. 7. We mirror the main paper results with close to perfect reconstruction scores both for the latent space and the group representation for $SO(n), O(n)$, but notice a drop in performance for $GL(n)$ as dimensionality increases beyond 5. In this case, we can recover the performance for $n = 7$ if we allow to backpropagate through the least-squares solver. As the dimension grows to $n = 9$, the model is no longer able to discover the full group structure, likely due to a mismatch between the complexity of $GL(9)$ and the size of the dataset/variation present.

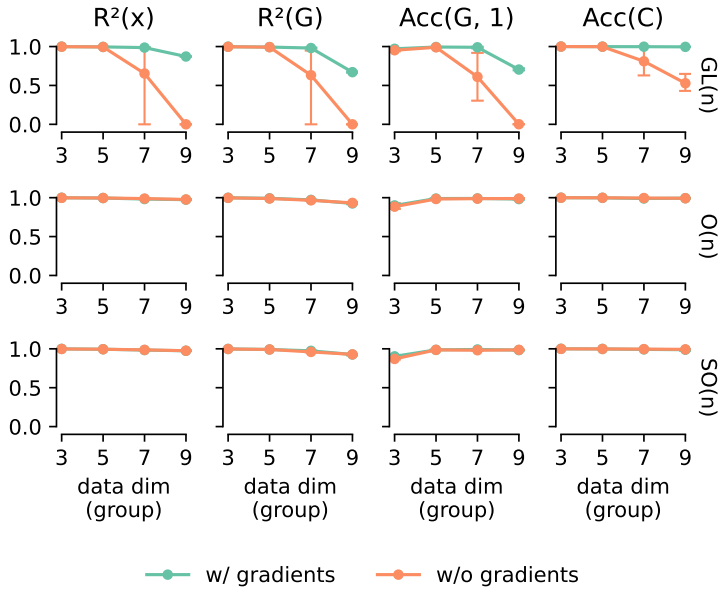

Figure 7: **Higher Latent Dimensions for Synthetic Group Data**. Comparison of EbC across an increasing number of true latent dimensions of the synthetic group dataset. We show results for all group types $GL(n)$, $O(n)$, $SO(n)$ with n being on the x axis and fixed content dimension $d = 5$. Additionally, we indicate whether we compute gradients through the linear least squares estimator or not.

C.2 Data efficiency

We analyze the data efficiency of our method by varying three key hyperparameters while keeping all other settings identical to those for $n = 3$ in Table 1 of the main paper. We test across $SO(3)$, $O(3)$, and $GL(3)$ by:

1. Varying the total dataset size.
2. Varying the number of negative samples used in the contrastive loss.
3. Varying the number of samples per action (k) used for the linear least-squares fit of $\hat{\boldsymbol{R}}(g)$.

Discussion (Dataset Size): The results in Table 3 show that our method is highly robust to smaller dataset sizes. Even with 20x less data (50k vs 1000k), the identifiability of the equivariant representation ($R^2(x)$ and $R^2(G)$) remains nearly perfect, staying above 99.6% for all groups. The primary effect of smaller datasets is a moderate drop in accuracy for the invariant part of the embedding ($Acc(C)$).

Table 3: Data efficiency: Reduced dataset size. We evaluate EbC trained on datasets ranging from 50k to 1M samples (1000k). The 1000k setting replicates the main paper's results. We report mean and standard deviation over 5 runs for $n = 3$.

Group Type	Dataset Size / Metric	1000k	100k	500k	50k
SO(n)	R^2(x)	99.81 ±0.04	99.79 ±0.06	99.75 ±0.22	99.80 ±0.03
	R^2(G)	99.75 ±0.04	99.71 ±0.07	99.66 ±0.28	99.72 ±0.05
	Acc(C)	99.19 ±0.69	97.30 ±2.11	98.97 ±0.72	95.25 ±1.96
O(n)	R^2(x)	99.80 ±0.06	99.81 ±0.03	99.79 ±0.12	99.73 ±0.16
	R^2(G)	99.73 ±0.06	99.74 ±0.03	99.72 ±0.14	99.65 ±0.19
	Acc(C)	99.06 ±0.87	97.90 ±1.46	98.92 ±0.76	94.57 ±1.68
GL(n)	R^2(x)	99.83 ±0.02	99.82 ±0.03	99.82 ±0.02	99.82 ±0.02
	R^2(G)	99.72 ±0.05	99.71 ±0.05	99.71 ±0.04	99.68 ±0.05
	Acc(C)	98.42 ±0.72	96.42 ±1.27	98.37 ±0.68	92.02 ±2.53

Table 4: Data efficiency: Reduced number of negative samples. We vary the number of negative samples per batch from 1024 to 16384. The 16384 setting replicates the results of table 1. We report mean and standard deviation over 5 runs for $n = 3$.

Group Type	Neg. sample batch size / Metric	1024	2048	4096	8192	16384
SO(n)	R^2(x)	99.51 ±0.26	99.66 ±0.09	99.48 ±0.71	99.77 ±0.04	99.81 ±0.04
	R^2(G)	99.28 ±0.38	99.52 ±0.11	99.30 ±0.91	99.67 ±0.04	99.75 ±0.04
	Acc(C)	98.23 ±1.38	98.78 ±0.76	98.60 ±1.32	99.13 ±0.69	99.19 ±0.69
O(n)	R^2(x)	99.55 ±0.07	99.63 ±0.11	99.62 ±0.28	99.75 ±0.10	99.80 ±0.06
	R^2(G)	99.37 ±0.10	99.49 ±0.13	99.49 ±0.36	99.64 ±0.14	99.73 ±0.06
	Acc(C)	98.49 ±0.81	98.90 ±0.59	98.97 ±0.57	99.09 ±0.62	99.06 ±0.87
GL(n)	R^2(x)	99.70 ±0.04	99.72 ±0.06	99.76 ±0.06	99.79 ±0.05	99.83 ±0.02
	R^2(G)	99.53 ±0.07	99.56 ±0.08	99.61 ±0.10	99.65 ±0.08	99.72 ±0.05
	Acc(C)	98.92 ±0.53	99.09 ±0.58	99.11 ±0.53	98.76 ±0.63	98.42 ±0.72

Discussion (Negatives): As shown in Table 4, reducing the number of negative samples has a minimal effect on performance. While there is a slight, consistent downward trend in the equivariant metrics ($R^2(x)$, $R^2(G)$) as negatives decrease, the changes are very small and performance remains high ($> 99.2\%$) even with 16x fewer negatives.

Table 5: Data efficiency: Reduced number of samples per action (k) for fitting $\hat{R}(g)$. We vary k from the theoretical minimum $n = 3$ up to $3n = 9$. Table 1 uses $k = 12$ ($4n$). We report mean and standard deviation over 5 runs for $n = 3$.

Samples per Action / Group Type	Metric	3	4	5	6	7	8	9
SO(n)	R^2(x)	99.71 ±0.06	6.86 ±7.87	89.12 ±32.03	99.85 ±0.01	99.83 ±0.06	99.83 ±0.04	99.77 ±0.11
	R^2(G)	-8.77 ±8.56	-9.59 ±28.67	88.16 ±33.06	99.63 ±0.05	99.66 ±0.13	99.70 ±0.07	99.66 ±0.16
	Acc(C)	99.20 ±0.52	51.17 ±19.08	97.47 ±4.53	99.40 ±0.47	99.30 ±0.74	99.07 ±0.94	99.10 ±0.68
O(n)	R^2(x)	99.65 ±0.10	4.68 ±5.56	99.77 ±0.09	99.84 ±0.03	99.83 ±0.05	99.23 ±1.60	99.39 ±1.21
	R^2(G)	-118.25 ±308.67	-0.03 ±0.09	99.09 ±0.35	99.59 ±0.09	99.67 ±0.10	98.75 ±2.56	98.95 ±2.15
	Acc(C)	99.00 ±0.82	54.32 ±12.03	98.88 ±0.81	99.18 ±0.50	99.13 ±0.90	97.58 ±4.59	97.94 ±3.98
GL(n)	R^2(x)	98.46 ±1.13	5.56 ±9.94	89.63 ±28.86	99.63 ±0.03	99.76 ±0.09	99.80 ±0.04	99.80 ±0.07
	R^2(G)	-0.28 ±0.47	-0.07 ±0.26	77.00 ±28.83	93.25 ±1.49	98.93 ±0.92	99.48 ±0.20	99.56 ±0.18
	Acc(C)	99.11 ±0.32	35.38 ±26.81	98.36 ±0.38	98.70 ±0.57	98.42 ±0.81	98.56 ±0.82	98.37 ±0.51

Discussion (Samples per Action): Table 5 confirms that the theoretical minimum of $k = n = 3$ samples is insufficient in practice. This is expected, as $k = n$ samples must form a full-rank system in latent space to uniquely identify $\hat{R}(g)$, which is unlikely to occur consistently. Using $k = 4$ also leads to instability. However, performance rapidly recovers. With $k = 6$ ($2n$), $R^2(G)$ is $> 99\%$ for $SO(3)$/$O(3)$ and $> 93\%$ for $GL(3)$. Using $k = 9$ ($3n$) yields near-perfect identifiability ($> 99.5\%$) for $R^2(G)$ across all groups, demonstrating practical applicability with a modest number of samples.

C.3 Robustness to noise

To study the effect of noise, we adjust our data-generating process. We introduce a Gaussian noise term $\epsilon \sim N(0, \sigma^2)$ to the group action relationship in the latent space: $x' = R(g_i)x + \epsilon$. We test this for $GL(3)$, varying the noise standard deviation σ from 0.0 to 0.1 and the number of samples per action (k) from 3 to 12. All other parameters follow the setup of Table 1.

Table 6: Robustness to noise for $GL(3)$. We add Gaussian noise $\epsilon \sim N(0, \sigma^2)$ to the latent transformation $x' = R(g)x + \epsilon$. We vary the noise standard deviation σ and the number of samples per action k ($3 = n$, $6 = 2n$, $9 = 3n$, $12 = 4n$) used to fit $\hat{R}(g)$. We report mean and standard deviation over 5 runs.

Noise Std.	Group Type Metric Samples/Action	R²(x)	R²(G)	GL(n) Acc(C)
0.0e+00	3	98.46 ±1.13	-0.28 ±0.47	99.11 ±0.32
	6	99.63 ±0.03	93.25 ±1.49	98.70 ±0.57
	9	99.80 ±0.07	99.56 ±0.18	98.37 ±0.51
	12	99.83 ±0.02	99.72 ±0.05	98.42 ±0.72
1.0e-05	3	97.89 ±1.88	-0.30 ±0.64	98.89 ±0.36
	6	99.62 ±0.06	93.24 ±1.20	98.68 ±0.78
	9	99.82 ±0.04	99.65 ±0.07	98.60 ±0.64
	12	99.82 ±0.03	99.70 ±0.06	98.48 ±0.61
1.0e-04	3	98.14 ±1.17	-28.47 ±65.56	99.00 ±0.36
	6	99.64 ±0.07	93.78 ±2.07	98.73 ±0.68
	9	99.83 ±0.02	99.65 ±0.06	98.45 ±0.47
	12	99.83 ±0.04	99.71 ±0.07	98.60 ±0.60
1.0e-03	3	98.58 ±0.28	-50.29 ±127.95	98.98 ±0.51
	6	99.58 ±0.11	93.33 ±1.76	98.51 ±0.67
	9	99.81 ±0.05	99.61 ±0.09	98.58 ±0.48
	12	99.81 ±0.04	99.68 ±0.08	98.50 ±0.70
1.0e-02	3	88.58 ±30.63	-1.02 ±1.98	98.98 ±0.50
	6	99.58 ±0.05	93.05 ±1.27	98.61 ±0.70
	9	99.81 ±0.04	99.59 ±0.08	98.46 ±0.81
	12	99.83 ±0.04	99.69 ±0.07	98.54 ±0.85
1.0e-01	3	94.56 ±11.34	-27.93 ±55.46	98.97 ±0.37
	6	99.58 ±0.06	91.76 ±0.23	99.04 ±0.56
	9	99.77 ±0.04	96.98 ±0.24	99.05 ±0.40
	12	99.82 ±0.03	97.98 ±0.13	98.94 ±0.74

Discussion (Noise): The results in Table 6 show that the $R^2(x)$ and $Acc(C)$ metrics are highly robust to noise, remaining stable even at $\sigma = 0.1$. As expected, noise primarily affects the identifiability of the group representation ($R^2(G)$). This effect is strongly dependent on the number of samples (k) used for the least-squares fit. With $k = 6$ ($2n$), $R^2(G)$ drops from 93.25% (no noise) to 91.76% ($\sigma = 0.1$). However, increasing the samples completely mitigates this. With $k = 9$ ($3n$), $R^2(G)$ only drops from 99.56% to 96.98%. With $k = 12$ ($4n$), the performance remains excellent, dropping from 99.72% to 97.98%. This demonstrates that the noise in the least-squares problem can be effectively averaged out by using more sample pairs, confirming the method's robustness.

C.4 Infinite dSprites

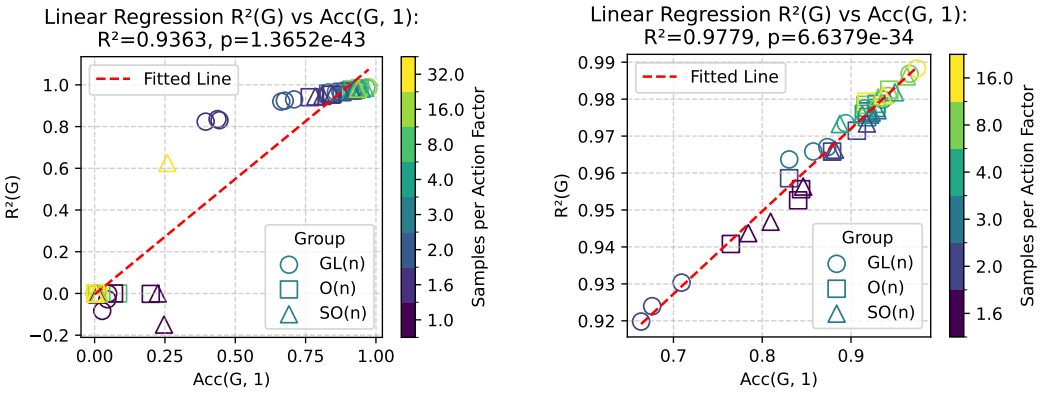

Figure 8: **Acc$(G, 1)$ as observable proxy for $R^2(G)$.** Comparison of our Top-1 NN-Lookup $Acc(G, 1)$ and group action identifiability metric $R^2(G)$ across EbC models trained on synthetic group datasets with varying samples per action factor k. Left: Results for the selected dataset configurations ($n = 5$, $d = 5$, $|C| = 1000$). Right: Filtered results for $Acc(G, 1) > 0.6$.

The Acc(G) metric is suitable for practical hyperparameter selection. In the main paper, we described the R2(G) metric to measure the quality of the group representation with respect to the ground truth group representation and latent factors. On practical experiments, this metric is not observable. Instead, on infinite dSprites but also for hyperparameter selection on the toy dataset, we consider the Acc(G, n) metric which performs a kNN lookup only in the reconstructed embedding space. Intuitively, this metric uses the computed embedding, and is a measure of self-consistency of the representation. As a validation for this choice, we varied the number of samples per action (one hyperparameter in the fitting procedure) and visualize the correleation between the observable Acc(G, 1) metric and the unobservable, but desired, R2(G) metric in Figure 8. On the full hyperparameter sweep, the metrics are significantly correlated with R2=93.6 (p<1e-10; Figure 8 left). When treating unsuccessful hyperparameter configurations as outliers, i.e., discarding models with Acc(G, 1) < 0.6, this correlation gets even more clear at R2=97.8 (p<1e-10; Figure 8 right).

Hyperparameter selection indicates suitable group dimension on idSprites We now leverage the $Acc(G, \cdot)$ metric to perform hyperparameter selection on the idSprites dataset. To provide a full picture, Figure 9 shows 10 variants of the metrics with differently strict criteria, considering top-1 to top-10 accuracies during the NN-lookup.

For 18 and 144 classes, we observe a clear increase in performance in all metrics at d=4 for the least conservative $Acc(G, 10)$ metric, however, the most conservative $Acc(G, 1)$ metric still indicates subpar performance at this dimensionality (below 50%). All metrics start saturating around a dimensionality of d=6, which would allow an embedding of $R_m \times Z_n \times Z_n$ into 3 circular dimensions.

Figure 10 shows example transitions at this hyperparameter point, which show close-to-perfect recovery of the group structure. However, as we overparameterize the model in terms of the group dimensions, the $Acc(G, \cdot)$ metrics improve slightly while the qualitative evaluation shows a noticeable decline in embedding and representation quality (Figure 11). In contrast to the results from Figure 8, this suggests a mismatch of the $Acc(G, \cdot)$ metric and the $R^2(G)$ metric which is unknown for this dataset.

We hypothesize that this is because the $Acc(G, \cdot)$ metrics a) are not able to measure the group action prediction exclusively, but instead also measure misclassification of the content, and b) don't measure the degree of the prediction error. Instead any type of classification error is counted with the same weight, no matter if the prediction was off by a small or large amount.

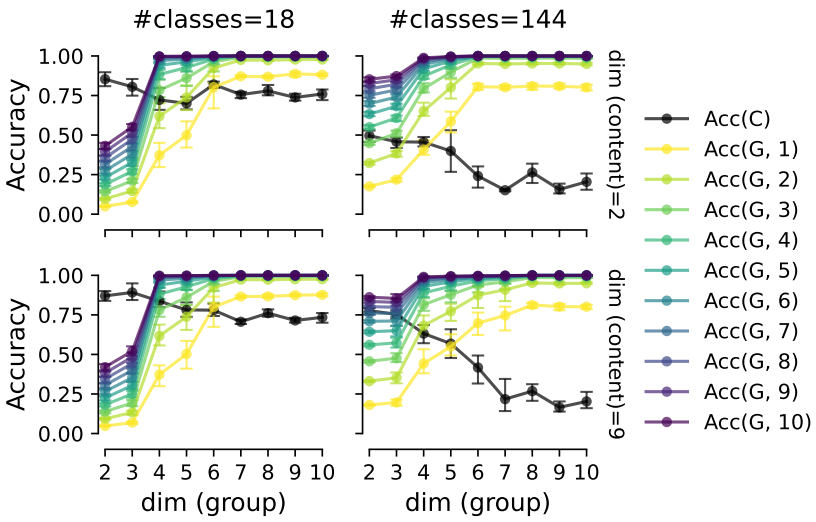

Figure 9: **Embedding Dimension Hyperparameter Sweep for idSprites dataset.** Comparison of EbC for the idSprites dataset across embedding dimensions for the group $n \in [2, .., 10]$ and for the content $d \in [2, 9]$ showing the observable Top-K NN-lookup metric $Acc(G, k)$ for $k \in [1, .., 10]$ as well as unobservable content classification accuracy $Acc(C)$. We show the comparison across two configurations of the idSprites dataset. Left: original dSprites configuration, Right: $16\times$ #shapes, $0.5\times$ #scales #rotations #translationsX #translationsY

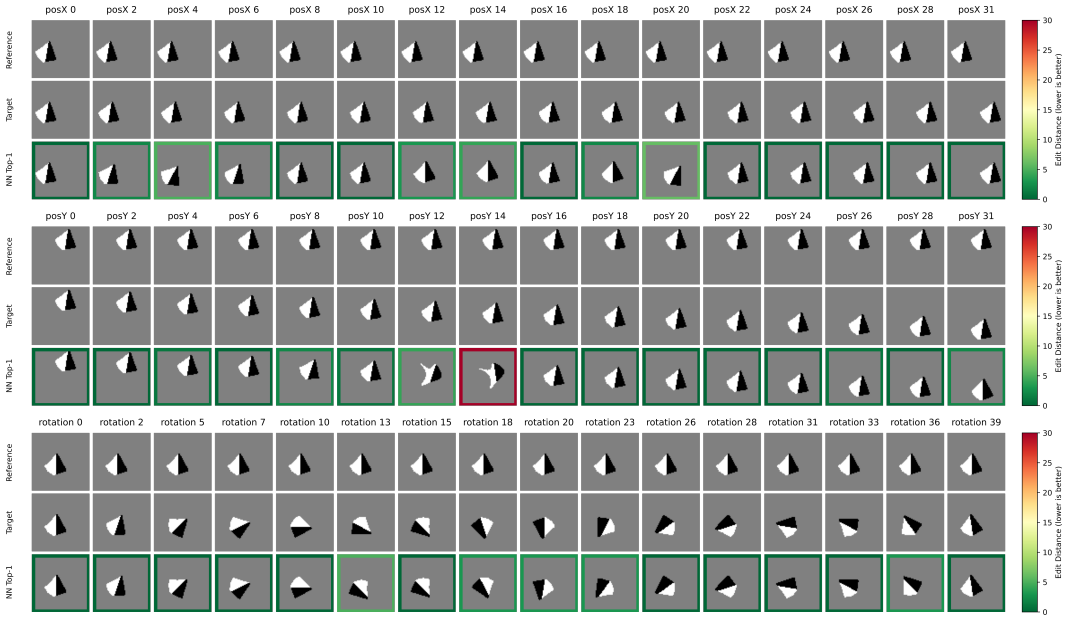

Figure 10: **Top-1 Nearest Neighbor Lookup based on EbC with 6D group dimensions.** For the idSprites dataset configuration with $|C| = 18$, we sample a random reference image $\boldsymbol{y}$ and generate a sequence of transformed images $\boldsymbol{y}'$ (targets) according to the different types of possible group actions. We estimate $\hat{\boldsymbol{R}}$ from the embeddings, compute $\hat{\boldsymbol{R}}\boldsymbol{x}$ and perform a top-1 NN lookup across *all* embeddings to get a prediction in image space. In the three blocks of images we show action traces across posX translation (top), posY translations (middle), and rotations (bottom). Within each block we show the reference image (first row), the target images (second row), the top-1 NN image prediction (third row). We color the images in the third row according to the Manhattan distance between the associated true latent factors of the target and predicted image.

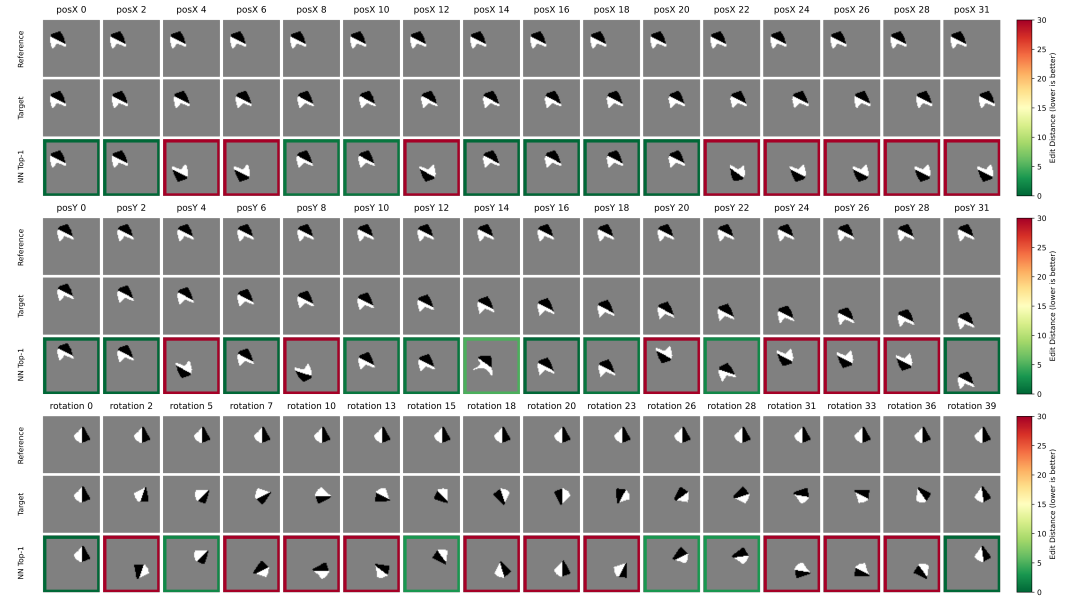

Figure 11: **Top-1 Nearest Neighbor Lookup based on EbC with 8D group dimensions.** Just like Figure 10 but for EbC with $n = 8$.

C.5 Application to real-world data

Here, we demonstrate how EbC can be used on a real-world time series dataset to find group equivariant embedding spaces. We apply EbC to the rat hippocampus dataset recorded by Grosmark and Buzsáki [14]. We use the data as preprocessed by Zhou and Wei [55]. In this dataset, neural activity from CA1 neurons in hippocampus of multiple rats was recorded via implanted silicon probes. During the recordings of each session, the rats moved on a 1.6m long linear track. Neural activity, the position on the track, and the movement direction of the rat was recorded (see Figure 12A). We apply EbC to the data of the following three rats: "Achilles", "Buddy" and "Gatsby".

Using EbC for data analysis requires to specify how positive pairs are sampled and how positive pairs are grouped together into the same group action. This is conceptually close to CEBRA [43], where auxiliary variables are used to resample the dataset to encode particular neuroscientific hypotheses. As in CEBRA, we leverage the behavioral variables of position and movement direction, but discretize the position variable with a bin size of 1cm.

We propose three different sampling strategies, depicted in Figure 12B:

1. Group=Position: We consider all possible pairs of data points to be positive pairs and group positive pairs based on the difference in position regardless of the movement direction.
2. Group=Position & Direction: We consider all possible pairs data points to be positive pairs and group positive pairs based on the difference in position and difference in movement direction.
3. Group=Position, Content=Direction: We consider only those pairs of data points where the movement direction does not change and we group positive pairs based on the difference in position.

We compare our method to CEBRA-behavior [43] which is a state-of-the-art method on this dataset and closely related to EbC in the sense that is also an encoder-only representation learning method with identifiability guarantees. We compare both against CEBRA-Behavior using only the position variable and the variant using both position and movement direction as label information. We follow the same dataset split into train, validation and test used in [43], and leverage consistency across runs for selecting the best hyperparameters for each model. To match the "offset10-model" used by CEBRA as best as possible, we leverage a time-delay embedding for EbC with receptive field 10.

For the CEBRA models, we run a hyperparameter sweep over latent dimensions of $d \in \{6, 8, 12, 16, 32\}$ and train for 10k steps. For the remaining parameters, we use the default parameters specified via CEBRA's sklearn API. In particular, this means using the cosine distance in the loss (restricting embeddings to the hypersphere), using the "offset10-model" (CNN based model with embeddings normalized to the hypersphere), and "time offset"=10 specifying how positive pairs are related in terms of the time offset between them.

For EbC, we run a hyperparameter sweep across group (2 or 3) and content dimensions (0, 1, or 4), as well as over the number of samples per action k for which we run experiments with ($k = 4d$, $k = 8d$). We use a three layer MLP with 128 hidden units per layer for the encoder. We compute the gradients through the linear regression modeling fitting as done in Appendix C.1. As for CEBRA, we train for 10k steps.

We compute the following three types of metrics:

1. **Decoding metrics**: We follow CEBRA's decoding setup using a KNN (with K=3) both for decoding the position and direction from the embeddings.
2. **Consistency metrics**: We use the consistency metrics across runs defined by CEBRA to measure how well different embedding spaces can be aligned to each other. This metric fits an affine transform to map from one embedding space to another and measures the R^2 score of the prediction. We use this on 5 runs of the same model with different seeds and compute the consistency between every resulting embedding space with every other embedding space of those runs.
3. **Linear Group Action (Position)**: To measure how well a given embedding space is linearly structured with regard to the position variable, we collect all embedding points related by a specific difference in the position variable (here: 5cm) and fit a linear model for which

we measure the R^2 score of the prediction. To make this metric fairer, we fit two separate models for pairs based on the direction label. The metric value we report is the average of these two individual R^2 scores.

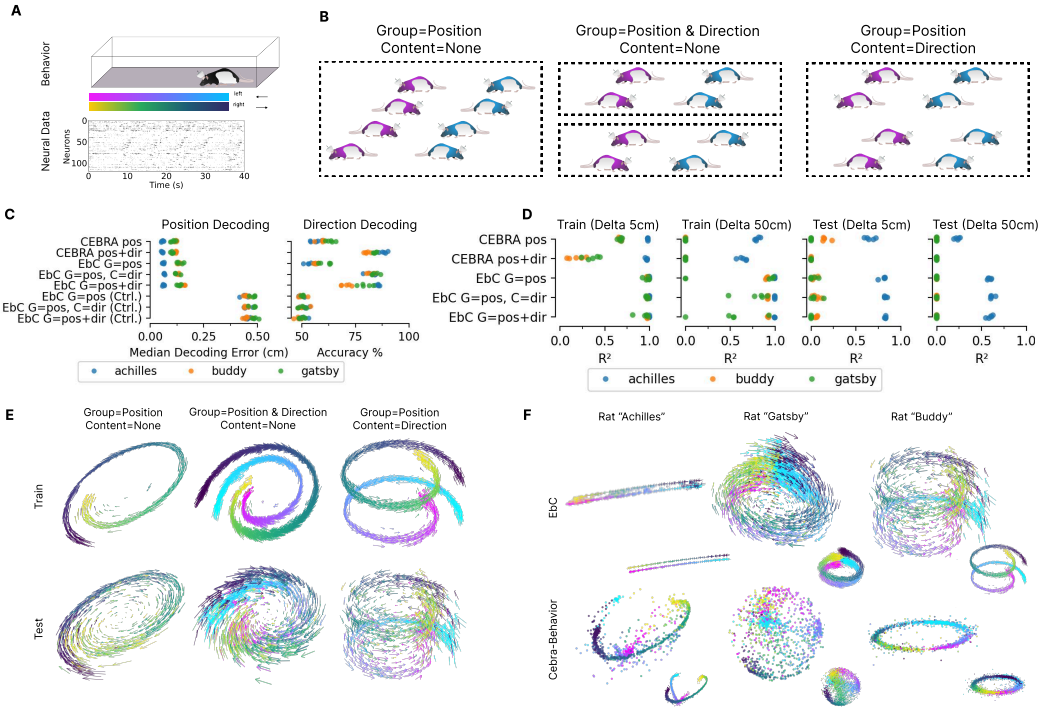

Figure 12: Application to neural time series data. (A) We used recordings from the hippocampus of rats running on a 1.6m long linear track. This yields two time series: neural activity y_t and behavior (position p_t and direction d_t). We use EbC to encode the neural activity with the behavior labels acting as labels determining the positive pairs and thereby the group structure. (B) Illustration of the different sampling strategies employed with EbC. (C) KNN-Decoding performance on the test set for decoding the position and direction label. We compare EbC to CEBRA and include a label shuffle control (D) Measure of linearity (for train and test set) of the embedding spaces conditioned on the difference in the position variable. Reporting the R^2 score of linear regression models trained on predicting the next point in latent space for pairs of data where the difference in position is 5cm. We also report the R^2 of using the same 5cm model on predicting 50cm ahead, i.e. 10 forward predictions in latent space. (E) EbC embeddings of the neural activity of the rat "Buddy" with different variations on how the behavior labels are used to determine positive pairs (F) EbC & CEBRA-Behavior Embeddings on rats Achilles, Gatsby and Buddy. Showing EbC model with group=position and content=direction, and CEBRA model with behavior labels position & direction.

EbC reliably encodes position and direction. Similar to CEBRA-behavior, EbC produces latent spaces from which both the position and movement direction of the rats can be decoded (Figure 12C). For position decoding, EbC and CEBRA-Behavior achieve comparable test performance: median errors of approximately 6 cm for “Achilles”, ≈12–14 cm (EbC) and ≈11 cm (CEBRA-behavior) for “Gatsby”, and ≈13–15 cm (EbC) and ≈12 cm (CEBRA-behavior) for “Buddy”. In contrast, models trained with shuffled labels yield errors of ≈40–50 cm, confirming that both methods leverage meaningful neural–behavioral relationships for decoding.

For direction decoding, the EbC variant trained with direction as part of the content performs on par with CEBRA-behavior. As expected, the EbC model trained with both position and direction as group variables performs worse—particularly for “Buddy”—because this formulation enforces invariance to direction. Consistently, EbC models using only position as the group variable (no content) perform close to the shuffle baseline for direction, reflecting the intended invariance.

EbC recovers latent linear group actions. EbC reveals highly linear relationships in the latent space with respect to position (Figure 12D). At a local scale (5 cm position differences), the recovered embeddings exhibit near-perfect linear structure across all three rats. Notably, a linear model fitted only on 5 cm transitions generalizes well to a more global scale: predicting 50 cm ahead (10 steps) while retaining most of its predictive performance. In contrast, CEBRA-behavior does not consistently capture such linear group structure. At the 5 cm scale, performance is comparable for "Achilles" but weaker for "Buddy" and "Gatsby". At 50 cm, the linear trend largely disappears for "Buddy" and "Gatsby"; for "Achilles", the R^2 drops below 75%, whereas EbC remains at ≈80–100%. However, for both methods, linear models fitted on the train embeddings do not systematically generalize to the test embeddings, potentially due to the limited data size. For "Achilles" which has roughly twice as many neurons recorded as in other sessions, predictive performance generalizes to the test set for EbC.

Sampling strategy induces structured representations. As described above, EbC allows explicit control over the structure of the latent space through the sampling strategy defining positive pairs. For "Buddy", using **Group = Position** (direction ignored) yields embeddings that are equivariant to position and invariant to direction (Figure 12E). When **Group = Position+Direction**, the latent space organizes position as an approximately circular linear trajectory, while direction is encoded as orthogonal movement towards or away from the center of this "position circle". When grouping by position difference while holding direction fixed within a group (**Group = Position, Content = Direction**), EbC separates the two behavioral variables: direction is encoded along the content dimensions, while the group (style) dimensions capture the linear equivariant structure of position.

Limitations and interpretation of model assumptions. Both CEBRA-behavior and EbC provide identifiability guarantees (recovering the latent space up to a linear/affine transformation) under specific assumptions about the data-generating process. Violations of these assumptions break identifiability, and thus each method can be interpreted as testing a specific hypothesis on the structure of the neural–behavioral mapping. For CEBRA-behavior (Pos+Dir), nearby latent points correspond to samples that are close in time, position, and direction, with latent trajectories constrained to the hypersphere. For EbC, pairs of samples are related through a general linear group action in Euclidean space, where the group action is defined by the sampling strategy (as described above).

Three key differences follow: (1) CEBRA-behavior incorporates temporal proximity into the hypothesis; (2) CEBRA-behavior constrains samples to be close in latent space, whereas EbC enforces the existence of linear relations between groups of pairs; and (3) in our experiments, CEBRA-behavior restricts representations to the hypersphere, while EbC does not. In this sense, EbC imposes a stronger structural hypothesis on the data, particularly through the assumption of linear group actions.

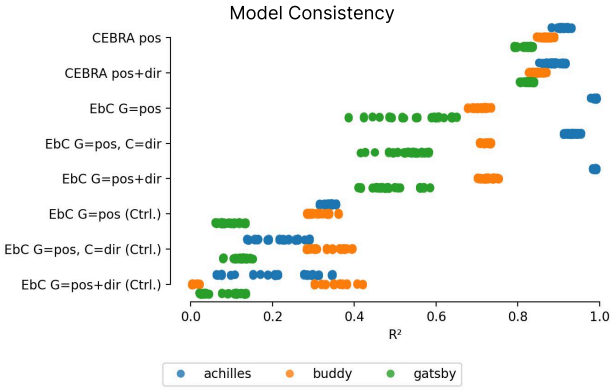

Figure 13: Consistency on the test set across model runs measured via the R^2 as defined by Schneider et al. [43].

Consistency across runs. Within this hypothesis-testing view, we assess whether embeddings from multiple runs (different random seeds) of the same model are linearly related, as predicted by the identifiability guarantees. For both EbC and CEBRA-behavior, we test whether a linear map exists that aligns embeddings across runs.

As shown in Figure 13, the highest consistency is observed for EbC (Group = Position and Group = Position+Direction) on "Achilles" (≈97–100%). For "Buddy", EbC reaches ≈70–75%, and for "Gatsby", ≈40–60%. In contrast, CEBRA-behavior achieves higher average consistency across rats: ≈85–93% (Achilles), ≈82–88% (Buddy), and ≈79–84% (Gatsby), although for Achilles, EbC exceeds CEBRA-behavior.

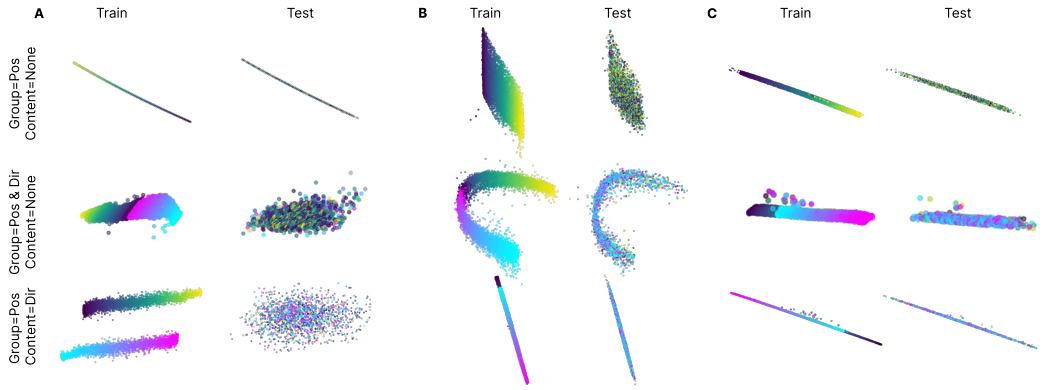

Figure 14: Embeddings of EbC label shuffle control experiments. Shows the same experimental setup as EbC experiments in Figure 12 but with the behavior labels randomly permuted before training.

Shuffle control: importance of held-out evaluation. The shuffled-label control highlights the necessity of train/(val)/test splits and cautions against interpreting structure from train embeddings alone. Even with randomized behavior labels, EbC imposes its specified group structure on the training latent space, producing structured embeddings across all rats (Figure 14). However, this structure differs from the non-shuffled embeddings in its general geometry and does not generalize to the test set when the neural–behavioral relationship is destroyed. When interpreting embeddings, it is hence important to consider that the mere linear arrangement of points in the embedding is consistently visible in the shuffled embeddings, but additional geometry in the non-shuffled embedding (e.g. alignment of positional information across directions; circular structure for some of the rats) is visible and non-trivial. On top, the observed structure in the non-shuffled embeddings transfers to the test set as an additional control to rule out the possibility of overfitting.

D Additional Literature Discussion and Related Work

Equivariant representation learning has been a prominent topic in machine learning research from different angles. Below we denote observed data as $\boldsymbol{y} \in Y$, with $\boldsymbol{y}' = g \cdot \boldsymbol{y}$ as short-hand for the transformed data. Embeddings of the observed data are denoted as $\boldsymbol{x}, \boldsymbol{x}' \in X$, where X is a vector space. Representations of the group are denoted as $R : G \to GL(n, X)$, and $\boldsymbol{x}' = g \cdot \boldsymbol{x} = R(g)x$. We focus on methods for learning vector embeddings $\boldsymbol{x}$ of the data $\boldsymbol{y}$, which are equivariant and/or invariant to group actions g. In contrast, orthogonal related work focuses on learning representations of the group given X, Y [29, 50]. Another rich area of literature focuses on neural network architectures which are invariant or equivariant to specific predefined groups [3, 5, 10, 27, 42].

Non-Invariant Methods. Invariance to all possible group actions (augmentations) may hinder downstream tasks (e.g. color-invariance for flower-type classification). Xiao et al. [49] propose to use a contrastive learning framework to learn representation space in which every subspace is invariant to only one type of augmentation. Eastwood et al. [9] extend this by introducing a second entropy loss to encourage subspaces to become disentangled.

Equivariant and Invariant Representation with known group action g (explicitly observing g in a parameterized form). Several encoder-based algorithms building on contrastive learning exist: E-SSL [4] embeds a reference sample of y and multiple transformed samples $\boldsymbol{y}'_i = g_i \cdot \boldsymbol{y}$ into a latent space $\boldsymbol{x}$, $\boldsymbol{x}'_i$ which is split into an invariant and equivariant subspace. Invariant parts are learned via SimCLR [2] and equivariant parts through an auxiliary task that predicts the parametrized group action g_i from $\boldsymbol{x}'_i$. EquiMod [7] achieves equivariance by embedding pairs of $\boldsymbol{y}$, $g \cdot \boldsymbol{y}$ by explicitly modeling the group action in latent space via a neural network $\boldsymbol{u}_\psi(\boldsymbol{x}, g) = \hat{\boldsymbol{x}}'$ and minimizing the distance between $\hat{\boldsymbol{x}}'$ and $\boldsymbol{x}'$. Park et al. [38] additionally parameterize the encoder and latent group action prediction model as G-equivariant neural networks. Garrido et al. [12] propose a variant using non-contrastive SSL losses.

Beyond contrastive learning, several approaches leverage (variational) autoencoders. Qi et al. [39] encode pairs of $\boldsymbol{y}, \boldsymbol{y}'$ into latent space and decode the group action g from the concatenated embeddings $\boldsymbol{x}, \boldsymbol{x}'$. Jin et al. [23] embed $\boldsymbol{y}$ into latent space, model the latent space group action as $\boldsymbol{x}' = \boldsymbol{R}(g)\boldsymbol{x}$ and decode $\boldsymbol{y}'$ from the predicted $\boldsymbol{x}'$ assuming full knowledge of $\boldsymbol{R}(g)$. Keurti et al. [24] use a linear prediction $\boldsymbol{x}' = R(g)\boldsymbol{x}$ where $\boldsymbol{R}(g)$ is predicted by a neural network from the observed and parametrized g, essentially an auto-encoding variant of Garrido et al. [12].

Equivariant & Invariant Representation with *unknown* group g (implicitly observing g) Although the aforementioned approaches share conceptual similarity in their goal of learning invariant and equivariant embeddings of data by modeling linear relations in latent space, EbC does not assume knowledge of the parametrized form of group actions g. Instead of observing information about g directly, a second class of methods assumes that two or more pairs of data share the same underlying action, $\boldsymbol{y}_i$, $g \cdot \boldsymbol{y}_i$. This is a key assumption we also require for EbC.

Encoder-only methods include CARE [15], a contrastive learning framework to learn invariant an locally equivariant representations. CARE encodes two pairs of observations $(\boldsymbol{y}_1, g \cdot \boldsymbol{y}_1)$ and $(\boldsymbol{y}_2, g \cdot \boldsymbol{y}_2)$ with the same group action g into a latent space such that the embeddings are related by the same matrix $R_g \in O(d)$ through $\boldsymbol{x}'_1 = \boldsymbol{R}_g \boldsymbol{x}_1$ and $\boldsymbol{x}'_2 = \boldsymbol{R}_g \boldsymbol{x}_2$. CARE achieves this by introducing an additional loss term to the InfoNCE loss, which maximizes the similarity between the dot products $\boldsymbol{x}_1^T \boldsymbol{x}'_1$ and $\boldsymbol{x}_2^T \boldsymbol{x}'_2$. Instead of applying this to a subspace of the embedding space, they apply both the invariant loss term of InfoNCE and their new equivariant loss term to the full embedding space and use a weighting factor to control the trade-off between invariant and equivariant representations. Yerxa et al. [51] propose a variation of CARE in which the embedding space is split into an invariant and equivariant subspace. Group actions are encoded as $\boldsymbol{x}'_1 = \boldsymbol{x}_1 + \boldsymbol{b}_g$ and $\boldsymbol{x}'_2 = \boldsymbol{x}_2 + \boldsymbol{b}_g$. STL [52] learns representations of (a) $\boldsymbol{y}, \boldsymbol{y}'$ such that $\boldsymbol{x}, \boldsymbol{x}'$ are equivariant to the group action g and (b) the group action g itself, again from two pairs of data. Similar to EquiMod [7], the latent group action is parameterized by a neural network. However, instead of assuming knowledge about g, they use a learned representation R_g as the second input to the neural network: $\boldsymbol{x}'_1 = u_\psi(\boldsymbol{x}_1, \boldsymbol{R}_g)$ where $\boldsymbol{R}_g$ is predicted by another network $\boldsymbol{R}_\theta(x_2, \boldsymbol{x}'_2) = \boldsymbol{R}_g$ from the latent representations of the second observed pair $(\boldsymbol{y}_2, g \cdot \boldsymbol{y}_2)$. To learn the correct representation of $\boldsymbol{R}_g$, EquiMod is extended by a third loss term that maximizes the similarity of $\boldsymbol{R}_\theta(\boldsymbol{x}_1, \boldsymbol{x}'_1)$ and $\boldsymbol{R}_\theta(\boldsymbol{x}_2, \boldsymbol{x}'_2)$. Like CARE, they don't learn separate subspaces and instead use weighting factors for the invariant and equivariant loss terms to control the trade-off between invariant and equivariant representations in X.

The problem can also be approached with auto-encoding approaches: Winter et al. [48] learn embeddings of $\boldsymbol{y}$ and g in which the factorize $\boldsymbol{y}' = g \cdot \boldsymbol{y}$ into an representation $\hat{\boldsymbol{x}}$ of $\boldsymbol{y}$ which is invariant to g and a representation $\boldsymbol{R}_Y(g)$ that represents the action g in the data space $\boldsymbol{Y}$, such that $\boldsymbol{y}' = \boldsymbol{R}_{\boldsymbol{Y}}(g)\delta(\hat{\boldsymbol{x}})$ with δ being the decoder.

The Unsupervised Neural Fourier Transform (U-NFT) [28, 36] is an auto-encoder framework with a linear group action model in latent space. For their learning setup, Koyama et al. [28] assume access to multiple sequences of data points $\{\boldsymbol{y}^{(i)} := (\boldsymbol{y}_0^{(i)}, ..., \boldsymbol{y}_T^{(i)}\}_{i=0}^N$ with $\boldsymbol{y}_{t+1}^{(i)} = g_i \cdot \boldsymbol{y}_t^{(i)}$, one sequence for each implicitly observed, but unknown group action $g_i \in G$. An autoencoder reconstructs $y_{t+1}^{(i)}$ from the predicted $\hat{\boldsymbol{x}}_{t+1}^{(i)}$, which in turn is predicted from $\hat{\boldsymbol{x}}_{t+1}^{(i)} = \boldsymbol{R}(g_i)\boldsymbol{x}_t^{(i)}$. The representation from these examples is estimated using least squares and a post-hoc basis transformation on the set of $\boldsymbol{R}(g_i)$ to find a block diagonal representation $\boldsymbol{B}(g_i)$ to facilitate disentanglement of the irreducible components of $\boldsymbol{R}(g_i)$. Mitchel et al. [35] propose a variation of NFT, where the learning setup is restricted to directly produce a block-diagonal representation $\boldsymbol{R}(g_i)$, avoiding the need for a post-hoc basis transformation. However, to achieve this, they also restrict $\boldsymbol{R}(g_i)$ to be orthogonal.

Winter et al. [48] and Allingham et al. [1] only recover an invariant representation and model the group action separately in the data space. All other the methods in this section try to solve the same general task of learning equivariant and invariant representations of the data without explicit knowledge of the underlying group actions. CARE (Gupta et al., 2023) is the closest encoder-only (contrastive) approach to EbC, but constrains the embedding to the hypersphere, which imposes additional structure on the learned representation, while EbC can learn different topologies (e.g., torus in Fig. 4, hypersphere in Fig. 5). In terms of function and data requirements, U-NFT (Miyato et al., 2022; Koyama et al., 2023) is the conceptually closest work, but requires to learn a full generative model of the data.

E NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our key theoretical claim is identifiability, which we proof through Theorem 1. Our empirical claim is validated on synthetic datasets as well as infinite dSprites.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 7.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The full proof for Theorem 1 is given in Appendix A. The proof for Corollary 1 is given directly in the main paper. The full definition of assumptions is stated in the Appendix, and informally outlined in the main paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The key components of the algorithm are outlined in Section 2. Experimental details are given in Section 4. Additional experimental details are provided in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code for our paper is available at https://github.com/dynamical-inference/ebc. During the phase of double-blind review, we provided an anonymized version of the codebase. For additional pointers, see Appendix B.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The most important details are outlined in Section 4. The full details are given in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run all experiments across dataset and model seeds, see Section 4 for details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix B contains a list of compute resources used for the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper conforms with the NeurIPS Code of Ethics in every regard.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: As noted in the guidelines, our method is a general-purpose approach for estimating the effect of actions/interventions from observable data. There is broad applicability of such an algorithm, but no specific societal impacts that are particularly tied to the paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We report results on small-scale synthetic datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Appendix B outlines all relevant license information.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.