Unsupervised Learning of Algebraic Structure from Stationary Time Sequences

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recently, there is a surge of interest in the data-driven learning of underlying simple relations and symmetry in the dataset. However, finding such a symmetry often requires structural assumptions about either a dataset or a model, or both. In this study, we investigate the symmetry that can be learned from time sequences of length at least three. We show that, if each time sequence in the dataset has a certain stationary property (e.g. constant velocity, constant acceleration), the algebraic structure that helps us extrapolate far into the future can be discovered without supervision. We will showcase our method from both empirical and theoretical perspectives. We will also show that, when the representation is trained so that the future observation can be predicted well by a linear transition in the latent space, the hidden disentangled structure of the dataset naturally emerges as a by-product through simultaneous block-diagonalization. Our result suggests that finding a simple structured relation and learning a model with extrapolation capability are two sides of the same coin.

1 Introduction

The recent evolution and successes of neural networks in machine learning fields have shown the importance of symmetry-aware neural network models [17, 39, 33, 54]. In particular, symmetries in the form of geometric/algebraic constraints have been proven useful in various applications involving high-dimensional, highly-structured observations. For example, recent literature of robotics and reinforcement learning has succeeded in exploiting the knowledge of geometrical symmetries to improve the sample efficiency [53, 56] or to train a model that generalizes to unseen observations [49].

However, building an inductive bias that matches the given dataset of interest is challenging, and recent studies have been exploring the ways to learn symmetries itself from observational sequences. Many of these approaches consider settings with relatively restrictive assumptions or weak supervision. For example, [47] allows the trainer to use the knowledge of the identities of the actions used in making the transition. Meanwhile, [3, 31, 30] essentially assume that the transition velocity of all sequences in the datasets are the same.

These studies indicate that there is still much room left for the question of "what is required for the sequential dataset/model to enable the unsupervised learning of the equivariance relation corresponding to the underlying symmetry". This paper advances this investigation by showing that if the sequential dataset consists of time series with a certain stationary property (constant velocity, constant acceleration), we can learn the underlying symmetries by simply training the model to be able to predict the future with linear transition in the latent space. Our theory in Section 3 shows that this strategy can learn a model that is almost equivariant. Moreover, we will experimentally show that, by training an encoder-decoder model in a framework of meta-learning, we can actually learn an equivariant model. In particular, we show that we can learn a hidden equivariance structure in the

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

dataset by splitting (1) the internal training step to compute the prediction loss of linear transition in the latent space from (2) the external training step to update the encoder and decoder. We will also empirically show that, in alignment with group representation theory [37], the learned linear latent transitions in our framework can be simultaneously block-diagonalized, and that each block corresponds to a disentangled factor of variation.

2 Related works

Recently, numerous studies have explored the ways to learn symmetry in a data-driven manner. There is rich literature in unsupervised/weakly supervised approaches that use sequential datasets to exploit the structure that is shared across time, and they all differ by the types of inductive bias. For example, the object-centric approach introduces inductive bias in the form of architecture, and equips the model with pre-defined slots to be allocated to objects [34, 29, 35]. Meanwhile, [20, 1] assumes that the symmetry to be found takes the form of the energy conservation law, and learns each variable in the law as a function of observations. While this approach assumes that some *energy* is preserved in each observation, we assume that the transition parameters like velocity and acceleration are preserved within each observation. Other more indirect forms of inductive bias include those relevant to distributional sparseness and symmetry defined through algebraic constraints. [27] for instance assumed that every stationary component of a given time series is generated by a finite and independent latent time series. [36] proposed to sparsely model the transition with a distribution of large kurtosis. Our work belongs to a family of unsupervised learning that seeks to find the underlying symmetry of the dataset based on an algebraic inductive bias that the transitions can be represented linearly in some latent space. In this sense, our inductive bias also has a connection to Koopman operator [38, 23, 2]. We are different from these studies in that we are aiming to learn a common encoding function (i.e. lifting function) under which the *set* of sequences following different dynamics can be described linearly. Also [59] applied Koopman operator on pedestrian walking sequences, and [21] used Koopman operator to separate the foreground from the background. However, [59] does not set out their model to solve the extrapolation, and neither [59] nor [21] discusses the natural algebraic decomposition of the latent space that results solely from the objective to predict the unseen future.

Unsupervised learning with algebraic/geometrical constraints Many studies impose algebraic constraints that reflect some form of geometrical assumptions. [15] uses a known coordinate map parametrization of a Lie group family to construct a posterior distribution on the manifold. [46] assumes that the observations are dense enough on the data-manifold to describe its tangent space, and exploits a property of random walks on the product manifold to decompose the data space. In the analysis of sequential datasets, [12, 50, 9, 47, 11] make some Lie group type assumptions about the transition. [47] also assumes that the identity of the actions in the sequences is known. As for the approaches with less explicitly geometrical touch, [31] uses capsule structure in their probabilistic framework to model a finitely cyclic structure while retaining the computability of posterior distribution. By design, [31] assumes that all sequences in the dataset transition with the same cyclic velocity. [58] enforces the underlying transition action to be commutative. Some of these studies learn the representation so that the linear transition in the latent space can be explicitly computed [47, 11, 3]. In particular, [3] presents a theory that suggests that a representation without this feature would have topological defects, such as discontinuity. Our approach shares a similar philosophy with these works except that, instead of imposing a strong assumption about the underlying symmetry, we only make a relatively weak stationarity assumption about the dataset; although we assume each sequence to be transitioning with constant velocity/acceleration, we allow the velocity/acceleration to vary across different sequences.

Disentangled Representation Learning Disentangled structure [26] is a form of symmetry that has also been actively studied. It is known that, under the i.i.d assumption of examples, *unsupervised* learning of disentanglement representation is not achievable without some inductive biases encoded in models and datasets [43]. In response to this work, subsequent works have explored different frameworks such as weakly-/semi-supervised settings [45, 44] and learning on sequential examples [36] to learn disentangled representations. For example, PhyDNet [22] disentangles the known physical dynamics from the unknown factors by preparing an explicit module called PhyCell. ICA [28] and recent works [60, 55] also discuss the identifiability property of learned representa-

tions. Classical methods like [26, 5, 32] take an approach of incorporating the inductive bias in the form of a probabilistic model.

We are different from many previous methods in that we do not equip our model with an explicit disentanglement framework. Our method achieves disentanglement as a by-product of training a model that can predict the future linearly in the latent space. The set of latent linear transformations estimated by our method for different time sequences can be simultaneously block-diagonalized, and the latent space each block corresponds to a disentangled feature. Our data assumption about constant velocity/acceleration might be similar in taste to the setting used by [28], in which the observed time series can be split into the finite number of stationary components.

3 Learning of equivariant structure from stationary time sequences

Our goal is to learn the underlying symmetry structure of a dataset in an unsupervised way that helps us predict the future. What do us humans require for the dataset when we are tasked to, for example, predict where a thrown ball would be in the next second? We hypothesize that we solve such a prediction task by analyzing a short, past time-frame with a certain stationary property (e.g., constant velocity/acceleration). Indeed, people with good dynamic visual acuity can chase a fast-moving object, because they can identify such a short stationary time-frame and use it to predict the near future *linearly* in their latent space. Based on this intuition, we propose to provide the trainer with a dataset consisting of constant velocity/acceleration sequences. We formalize this idea below.

Dataset structure Our dataset $\mathcal{S}$ consists of sequences in some ambient space $\mathcal{X}$, so that each member $\mathbf{s} \in \mathcal{S}$ takes the form $\mathbf{s} = [s_t \in \mathcal{X}; t = 1, ..., T] \in \mathcal{S}$. Because we want $\mathbf{s}$ to be describing a sequence that transitions with constant velocity, we assume that all s_t in a given instance of $\mathbf{s} \in \mathcal{S}$ are related by a fixed transition operation $g \in \mathcal{G}$ so that $s_{t+1} = g \circ s_t$ for all t, where $\mathcal{G}$ is the set of transition operators on $\mathcal{X}$ and each $g \in \mathcal{G}$ acts on $x \in \mathcal{X}$ by sending x to $g \circ x$. We assume that $\mathcal{X}$ is closed under $\mathcal{G}$; that is, $g \circ x \in \mathcal{X}$ for all $x \in \mathcal{X}, g \in \mathcal{G}$. We allow $\mathcal{G}$ to be continuous as well, so that g might not have a finite order (For instance, if g is a rotation with speed $2\pi r$ with irrational r, any finite repetition of g would not agree with identity mapping). This way, our setting is different from those used in [31] that explores a cyclic structure using the capsules of same size. We emphasize that the transition action g is generally assumed to differ across the different members of $\mathcal{S}$. For example, if $\mathcal{G}$ is a set of rotations and $\mathcal{X}$ a set of images, then the rotational speed, direction and the initial image may all be different for any two distinct sequences, $\mathbf{s}$ and $\mathbf{s}'$. Because each instance of $\mathcal{S}$ is characterized by s_1 and g, we may write $\mathbf{s}(s_1, g)$ to denote a sequence that begins with initial frame s_1 and transitions with g. Summarizing, $\mathcal{S}$ is a subset of $\{[g^t \circ s_1; t = 0, ..., T-1]; s_1 \in \mathcal{X}, g \in \mathcal{G}\}$.

Prediction framework through equivariance Our strategy is to exploit the stationary property of each $\mathbf{s} \in \mathcal{S}$ to seek an invertible continuous function $\Phi : \mathcal{X} \to \mathbb{R}^{a \times m}$ such that there exists some $M : \mathcal{G} \to \mathbb{R}^{a \times a}$ satisfying

$$M_g \Phi(x) = \Phi(g \circ x) \text{ for all } x \in \mathcal{X} \text{ and } g \in \mathcal{G}. \tag{1}$$

This relation is known as equivariance [10]. In other words, we seek a model in which $\mathcal{X}$ and $\mathbb{R}^{a \times m}$ are invertibly related by an equivariance relation with respect to $\mathcal{G}$, where $g \in \mathcal{G}$ acts on $\Phi(x) \in \mathbb{R}^{a \times m}$ via the map $\Phi(x) \mapsto M_g \Phi(x)$ with $M_g \in \mathbb{R}^{a \times a}$. We assume $m > 1$ in our study[1]. In this framework, we predict the sequence $\mathbf{s}(s_1, g)$ with the relation $\Phi^{-1}(M_g \Phi(s_{t-1})) = \tilde{s}_t$. When $\mathcal{G}$ is a group, (1) would imply $M_{gh}\Phi(x) = \Phi(gh \circ x) = M_g \Phi(h \circ x) = M_g M_h \Phi(x)$ for all x, and the map $g \mapsto M_g$ is called a *representation* of $\mathcal{G}$ [8, 57, 9].

Because we are aiming to establish the framework in which the representation of each action g can be explicitly computed, our philosophy has much in common with the proposition of [3]. This approach is in contrast to [31], which encodes a predefined cyclic structure in the model.

[1] We note that, when $m > 1$, the action of M_g on $\mathbb{R}^{a \times m}$ can be realized by applying the same M_g to m-copies of $\mathbb{R}^a$. In other words, our prediction framework assumes that there are m number of subspaces that react to g in the same way. This $m > 1$ assumption is also considered in [3]. The case in which there are no copies of subspaces that act in the same way is called multiplicity-free in the literature of representation theory [8, 18], and is known to be a special case that happens only under a restrictive condition on the dimension of the observations space [40]. A similar idea has been used in model architecture as well [14].

3.1 Learning equivariance relation from stationary sequential dataset

However, training the model satisfying (1) with just the *constant velocity assumption* is not a trivial task, because this assumption only assures that, for each sequence $\mathbf{s}(s_1, g)$, there is a sequence-specific operator $M(g, s_1)$ that determines the transition. In order to satisfy the *full* equivariance (1), $M(g, s_1)$ shall not depend on s_1 (i.e. homogeneous with respect to s_1). At the same time, because the constant velocity assumption applies to each sequence over all time intervals, it at least assures that the latent transition M is well defined within each sequence; that is, $M(g, x) = M(g, g \circ s_1) = M(g, g^2 \circ s_1) \cdots$ and so on. It turns out that, with some regularity assumptions on the model and the choice of $\mathcal{G}$, we can extend this observation to say that M satisfies **intra-orbital homogeneity**; that is, $M(g, x)$ is constant on the orbit $\mathcal{G} \circ x = \{g \circ x; g \in \mathcal{G}\}$ for each x.

Proposition 3.1. *Suppose that $\Phi(s_t) = M(g, s_1)\Phi(s_{t-1})$ for all $\mathbf{s}$ and t. If $m > a$ and if $\mathcal{G}$ is a compact commutative Lie group, then M satisfies intra-orbital homogeneity.*

Also, if M satisfies intra-orbital homogeneity, $M(g, x)$ and $M(g, x')$ for any pair (x, x') in different orbits $\mathcal{G} \circ x \neq \mathcal{G} \circ x'$ can be shown to be at least similar.

Proposition 3.2. *Suppose that $M(g, x)$ satisfies intra-orbital homogeneity, and suppose that $\mathcal{G}$ is a compact connected group. If $M(g, x)$ is continuous with respect x, then for all (x, x'), there exists some P such that $PM(g, x)P^{-1} = M(g, x')$.*

Thus, much of the equivariance property can be satisfied automatically by training the representation on the set of stationary sequences. Interestingly, as we experimentally demonstrate later, our training method in the next section successfully learns a fully equivariant Φ without explicitly enforcing the change of basis P to be I.

4 Encoding Algebraic Structure by Meta-optimization

We propose a meta-learning way to learn a homeomorphic function $\Phi : \mathcal{X} \to \mathbb{R}^{a \times m}$ with equivariance property by seeking an injective Φ such that $M(g, s_1)\Phi(s_t) = \Phi(s_{t+1})$ for all $g \in \mathcal{G}$, $s_1 \in \mathcal{X}$. We learn such Φ by casting this problem as a meta-learning problem in which $M(g, s_1)$ is to be internally estimated for each $\mathbf{s}$. In other words, we seek a pair of an encoder Φ and a decoder Ψ such that $\mathcal{L}(\Phi, \Psi) = \min_M \|\Psi(M\Phi(s_t)) - s_{t+1}\|_2^2$ is optimized at all time t for each $\mathbf{s}$.

We conduct this optimization by splitting each $\mathbf{s} = \{s_1, ..., s_T\}$ into conditional time sequence $\mathbf{s}_\mathrm{c} = \{s_1, ..., s_{T_\mathrm{c}}\}$ and validation time sequence $\mathbf{s}_\mathrm{p} = \{s_{T_\mathrm{c}+1}, ..., s_T\}$, while using the former for the internal optimization of M to force the linear algebraic relation in the latent space and using the latter for the prediction loss. More precisely, we solve the following optimization problem about Φ and Ψ:

$$\begin{aligned} \mathcal{L}^p(\Phi, \Psi) &:= \textstyle\sum_{\mathbf{s}} \sum_{t=T_\mathrm{c}+1}^{T} \left\| \Psi(M^*(\mathbf{s}_\mathrm{c}|\Phi)^{t-T_\mathrm{c}} \Phi(s_{T_\mathrm{c}})) - s_t \right\|_2^2 \\ &\text{where } M^*(\mathbf{s}_\mathrm{c}|\Phi) = \arg\min_M \textstyle\sum_{t=1}^{T_\mathrm{c}-1} \|M\Phi(s_t) - \Phi(s_{t+1})\|_F^2 . \end{aligned} \tag{2}$$

Figure 1 describes our overall framework when $T_\mathrm{c} = 2$ and $T_\mathrm{p} = 1$. It might appear as if we can also set $\mathbf{s} = \mathbf{s}_\mathrm{c}$ and optimize the following reconstruction version of Eq.(2):

$$\mathcal{L}^r(\Phi, \Psi) := \textstyle\sum_{\mathbf{s}} \sum_{t=2}^{T_\mathrm{c}} \|\Psi(M^*(\mathbf{s}|\Phi)^{t-1}\Phi(s_1)) - s_t\|_2^2. \tag{3}$$

However, as we will see in the experiment section, the use of the validation sequence $\mathbf{s}_\mathrm{p}$ makes a substantial difference in the learned representation. This is most likely because the minimization of the validation error of M^* on $\mathbf{s}_\mathrm{p}$ would encourage Φ to exclude the $\mathbf{s}_\mathrm{c}$-specific information from the transition M^*. We will illustrate this effect in the experiment section. We shall note that we can also parameterize M as $M := \exp(A)$, where $A \in \mathbb{R}^{a \times a}$ is a Lie algebra element to be internally optimized. This type of approach was used in [13] for building Lie group convolutional network and in [50, 11] for predicting a sequence that is not necessarily stationary. In order to train their model, [50] used additional parameters to diagonalize each algebra element as well as hyperparameters to stabilize the training. We also experimented with a Lie-algebra style representation of M^* and used SGD to internally optimize the exponent parameters, but we needed to carefully tune the hyper-parameter for the norm regularization term to stabilize the training and never really succeeded to train the model without collapsing. Our internal optimization procedure is free of such a parameter tuning.

Internal Optimization of M^* Because the internal optimization in Eq.(2) is a linear problem, it can be solved analytically as

$$M^*(\mathbf{s}_c|\Phi) = H_{+1}H_{+0}^{\dagger}, \tag{4}$$

where $H_{+0} = [\Phi(s_1); ...; \Phi(s_{T_c-1})] \in \mathbb{R}^{a\times(T_c-1)m}$ and $H_{+1} = [\Phi(s_2); ...; \Phi(s_{T_c}) \in \mathbb{R}^{a\times(T_c-1)m}$ are the horizontal concatenations of the encoded frames and $H_{+0}^{\dagger}$ is the Moore-Penrose pseudo inverse of H_{+0}. Because $M^*(\mathbf{s}_c|\Phi)$ is a closed form with respect to Φ, the loss (2) can be directly optimized by differentiating it with respect to the parameters of both Φ and Ψ. Thus, the training is done in an end-to-end manner. Figure 1 summarizes the overall procedure to make prediction on a given sequence when $T_c = 2, T_p = t$. We note that, although we have assumed the dataset to consist of constant-velocity sequences, we can readily extend our method to the dataset consisting of the time series with higher-order stationarity, such as constant acceleration. See Section 5.4 for the detailed explanation of the model extension and the experimental results.

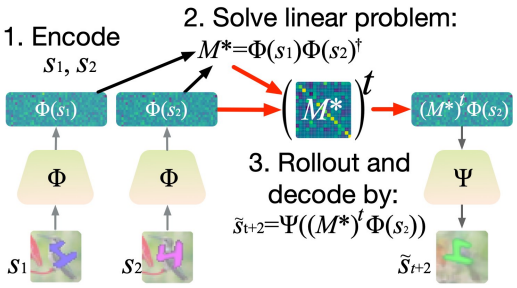


Figure 1: Prediction with our method when $T_c = 2$ and $T_p = t$. After the encoder encodes the observations into tensor representations $\Phi(s_1), \Phi(s_2)$, the method solves the least square problem: $M^* = \arg\min_M \|M\Phi(s_1) - \Phi(s_2)\|_F^2$. The model then predicts the future observations by $\tilde{s}_{t+2} = \Psi((M^*)^t\Phi(s_2))$. These processes (including the linear problem) are all differentiable.

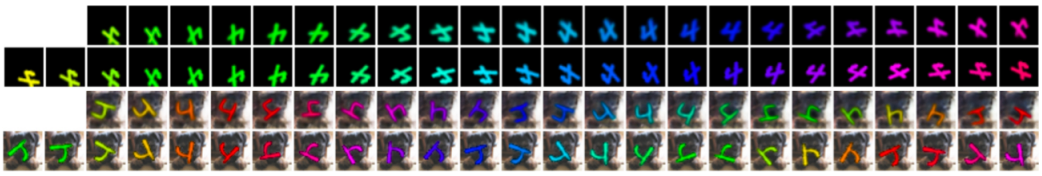
5 Experiments

We conducted several experiments to investigate the efficacy of our framework. In this section, we briefly explain the experimental settings. For more details, please see Appendix D. We tested our framework on *Sequential* MNIST, 3DShapes [4], and SmallNORB [42]. Sequential MNIST is created from MNIST dataset [41]. For all experiments, we used a ResNet [24]-based encoder-decoder architecture and we set $a = 16$ and $m = 256$ so that the latent space lives in $\mathbb{R}^{16\times256}$.

For Sequential MNIST, we chose our $\mathcal{G}$ to be the set of all combinations of three types of transformations: shape rotation, hue rotation, and translation, and randomly sampled a single instance of $g \in \mathcal{G}$ for each sequence(See Appendix D for the examples of sequences). To create each sequence, we first resized the MNIST image to 24×24, applied repetitions of a randomly sampled, fixed member of $g \in \mathcal{G}$ and embedded the results to 32×32 images. For shape and hue rotations, we randomly sampled the velocity of angles from uniform distribution on the interval $[-\pi/2, \pi/2)$ for each sequence. For translation, we randomly sampled the start point and end point in the range of [-10, 10], and then moved the digit images on a straight line between the sampled points. We also experimented on sequential MNIST with background (Sequential MNIST-bg). For Sequantial MNIST-bg, we used the same generation rule as Sequential MNIST but we added background images behind the moving digits. For the background, we used a randomly sampled images from ImageNet [48], which were all resized to 32×32. Also, we only used the images of digit 4 for most of the experiments on Sequential MNIST/MNIST-bg. Unless otherwise noted, all evaluations in this paper for the Sequential MNIST are based on training with only *digit 4*.

3DShapes and SmallNORB are datasets with multiple factors of variation. We created a set of *constant-velocity* sequences from these datasets by varying a fixed combination of factors for each sequence. That is, on these datasets, we chose our $\mathcal{G}$ to be the set of variations of factors, and sampled each g as $\prod_i g_i^{\ell_i}$, where g_i represents the increase of i-th factor by one unit and $\ell_i \in \mathbb{Z}$. Thus, the value of ℓ_i represents the velocity in the direction of the i-th factor on the grid. For 3DShapes, we chose *wall hue*, *floor hue*, *object hue*, *scale*, and *orientation* as the factors to vary. We varied *elevation* and *azimuth* for SmallNORB. For the split of each sequence into $(\mathbf{s}_c, \mathbf{s}_p)$, we set $T_c = 2$ and $T_p = 1$ on all of the constant velocity experiments.

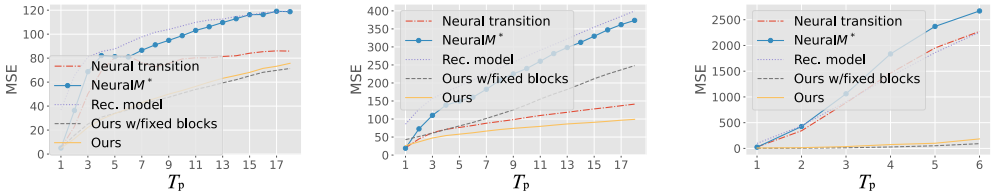
We also conducted experiments for the sequences with constant acceleration on Sequential MNIST. To create a sequence with constant acceleration, we chose a pair $g_a, g_v \in \mathcal{G}$ for each sequence, and generated $\mathbf{s}$ by setting $s_{t+1} = g_a^t g_v s_t$. We elaborate on the detail of this extension in 5.4.



(a) Generated examples with our proposed method on Sequential MNIST and MNIST-bg

(b) Comparison of variant methods

Figure 2: (a): Predictions made by our method on Sequential MNIST and MNIST-bg. The ground truth sequence is placed below the predictions, with the first two images representing $\mathbf{s}_{\mathrm{c}}$. (b): Typical failure examples generated by the comparative methods. (1)(2)(3) and (4) are NeuralM^*, Neural transition, Rec. model and Ours w/ fixed block, respectively. See Appendix B for more examples.



(a) Sequential MNIST (b) Sequential MNIST-bg (c) 3DShapes

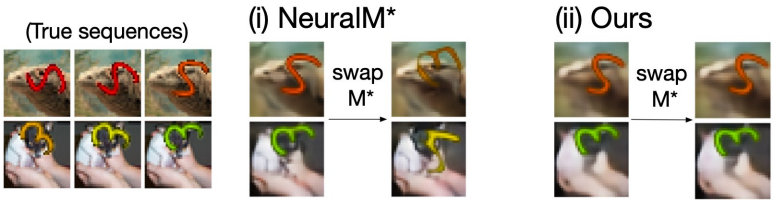
Figure 3: Prediction errors $\mathcal{L}^p$ with $T_{\mathrm{c}} = 2$ and $T_{\mathrm{p}} = 1, \ldots, 18$. During the training phase, models are trained to predict the observations only at $T_{\mathrm{p}} = 1$. The prediction errors at $T_{\mathrm{p}} > 1$ indicate the extrapolation performance. The results on SmallNORB can be found in Appnedix A.

As ablations, we tested several variants of our method: **fixed 2x2 blocks** (abbreviated as fixed blocks), **Neural**M^*, **Reconstruction model** (abbreviated as Rec. Model), and **Neural transition**. For the method of *fixed 2x2 blocks*, we separated the latent tensor $\Phi(s) \in \mathbb{R}^{16\times 256}$ into 8 subtensors $\{\Phi^{(k)}(s) \in \mathbb{R}^{2\times 256}\}_{k=1}^{8}$ and calculated pseudo inverse for each k to compute the transition in each $\mathbb{R}^{2\times 256}$ dimensional space. This variant yields M^* as a direct sum of eight 2×2 matrices. We tested this variant to see the effect of introducing a predetermined representation theoretic structure as in [9]. For Rec. model, we trained Φ and Ψ based on $\mathcal{L}^r$ in Eq. 3 with $T_{\mathrm{c}} = 3$. We tested this variant to see the effect of our use of T_p. For NeuralM^*, we trained an additional network M_θ that maps $\mathbf{s}_{\mathrm{c}}$ to a transition matrix, replaced M^* with M_θ in (4), and optimized θ and (Φ, Ψ) simultaneously. We may see NeuralM^* as a variant in which the *meta* part of the internal and external training is removed from our method. For Neural transition, we trained 1x1 1D-convolutional networks to be applied to latent sequences *in the past* to produce the latent tensor in the next time step; for instance, $\tilde{s}_{t+1} = \Psi(1\mathrm{DCNN}(\Phi(\mathrm{s_t}), \Phi(\mathrm{s_{t-1}})))$ when $T_{\mathrm{c}} = 2$ [2]. The 1DCNN was applied along the multiplicity dimension m. In this variant, the relation between $\Phi(s_t)$ and $\Phi(s_{t+1})$ is not necessarily linear. Section D.1 in Appendix describes each of the comparison methods more in detail. In testing all of these variants, we used the same pair of encoder and decoder architecture as the proposed method.

5.1 Qualitative and quantitative results on the prediction

Figure 2 shows the example sequences generated by the proposed model and comparative models. Figure 3 presents the prediction performance at $T_{\mathrm{c}} + T_{\mathrm{p}}$ when $T_{\mathrm{c}} = 2$. To produce this result, we back-propagated the prediction error at $T_{\mathrm{p}} = 1$ to the encoder during the training, and the prediction at $T_{\mathrm{p}} > 1$ was used to evaluate the extrapolation performance. Our method successfully predicts the images for $T_{\mathrm{p}} \geq 1$. Neural transition and NeuralM^* had almost the same prediction performance at $T_{\mathrm{p}} = 1$, but they both failed in extrapolation. Our *fixed 2×2 blocks* variant failed in extrapolation as

[2] This model can be seen as a simplified version of [51]



(a) Dependency of M^* on sequence. The 2x3 tiled images in the leftmost panel represents two sequences $\mathbf{s}^{(1)}, \mathbf{s}^{(2)}$ with the same transition action g. We consider the effects of $M^{*(1)}$ and $M^{*(2)}$ inferred respectively from $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$. NeuralM^* fails in prediction when $M^{*(2)}$ is used to predict $\mathbf{s}^{(1)}$. Our method does not fail by this swap, indicating $M^{*(2)} \cong M^{*(1)}$.

	MNIST		MNIST-bg		3DShapes		SmallNORB	
Method	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$
Rec. Model	38.98	56.01	86.72	87.98	95.09	175.50	60.41	76.45
NeuralM^*	7.04	62.71	19.80	84.07	22.57	329.08	28.36	49.55
Ours	5.33	**13.36**	26.26	**33.89**	8.12	**8.18**	29.23	**39.88**

(b) Equivariance performance based on $\mathcal{L}^p$(Eq.(2)) and $\mathcal{L}^p_{\text{equiv}}$(Eq.(5)) with $T_{\text{c}} = 2$ and $T_{\text{p}} = 1$. To evaluate equivariance errors on more difficult settings, we used all of digits in Sequential MNIST-bg for both training and test sets.

Figure 4: Quantitative and qualitative evaluation of learned equivariance.

well. This might be because the over-regularized structure of 2x2 block hindered with the training of the SGD optimization [16].

To evaluate how our learned representation relates to the structural features in the dataset, we also regressed the factors of transition from M^* and regressed the class of the digits from $\Phi(s_1)$ (Figures 8 and 9 in Appendix A). SimCLR [7] and contrastive predictive coding (CPC) [52, 25] are tested as baselines. Please see Appendix D for the detailed experimental settings for SimCLR and CPC. Our method yields the representation with better prediction performance than the comparative methods on the test datasets.

5.2 Equivariance performance

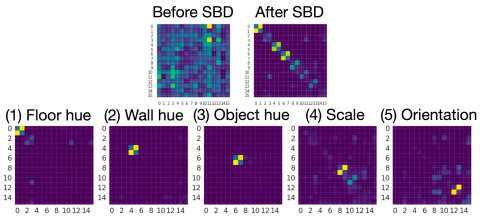
As we have described through Section 3.1 and 4, the equivariance is achieved when $M^*(\mathbf{s}(s_1, g)|\Phi)$ in (2) does not depend on s_1, where we recall that $\mathbf{s}(s_1, g)$ represents the sequence that begins with s_1 and transitions with g. To see how much the trained model is equivariant to the transformations in the sequential dataset, we therefore calculated the *equivariance error*, which is the prediction error from applying $M^*(\mathbf{s}|\Phi)$ to $\Phi(s'_{T_c})$ for a pair $\mathbf{s} \neq \mathbf{s}'$ that transitions with the same g. In other words, when $T_c = 2$, we compute the following;

$$\mathcal{L}^p_{\text{equiv}} := \mathbb{E}_g \mathbb{E}_{\mathbf{s},\mathbf{s}' \in \mathcal{S}(g)} [\|\Psi(M^*(\mathbf{s}|\Phi)\ \Phi(s'_2)) - s'_3\|_2^2] \tag{5}$$

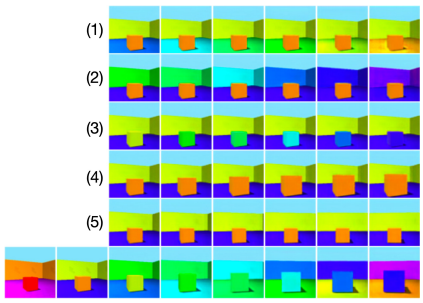
where $\mathcal{S}(g)$ represents the set of all sequences that transition with g. For each pair of $\mathbf{s} \neq \mathbf{s}'$ we set $T_{\text{c}} = 2$ and $T_{\text{p}} = 1$ as done in the experiments in the previous sections. Figure 4b compares the NeuralM^* method against our method in terms of the equivariance error. Figure 4a shows the result of applying $M^*(\mathbf{s}|\Phi)$ on $\Phi(s'_2)$ and applying $M^*(\mathbf{s}'|\Phi)$ on $\Phi(s_2)$. We see that when we swap M^* this way, NeuralM^* also swaps the digits; this implies that M^* learned by NeuralM^* encodes the *sequence* specific information together with the transition. On the other hand, swapping of M^* does not affect the prediction for our method, suggesting that our method is succeeding to learn an equivariant model. This is somewhat surprising, because our model does not have the explicit mechanism to enforce the full equivariance ($P = I$ in Section 3.1).

5.3 Structures found by simultaneous block-diagonalization of M^*s

So far, we have observed that our model has an ability to learn a $\mathcal{G}$-equivariant model. Representation theory guarantees that, if $\mathcal{G}$ is a compact connected group, any representation $D : \mathcal{G} \to \mathbb{R}^{a \times a}$



(a) Simeltaneously block-diagonalized matrices



(b) Disentangled transition representation on 3DShapes

Figure 5: (a) Simultaneous block-diagonalization (SBD) of M^*. The top right matrix is the visualization of $abs(I - \hat{M}^*)$ averaged over all of the training sequences. Each of the five matrices below is the visualization of $abs(I - \hat{M}^*)$ averaged over the set of sequences on which only a single factor was varied. Coordinates are permuted for better visibility. (b) Sequences generated by applying the transformation of just one block. To produce the disentangled sequences in each row from the leftmost two images in the bottom row, we performed the internal optimization of M^* while setting *all but the block positions corresponding to each factor of variation* to be identity. We elaborate this result and the results for Sequential MNIST, MNIST-bg and SmallNORB in Appendix A.

can be simultaneously block-diagonalized; that is, there is a common change of basis U such that $UD_gU^{\mathrm{T}} = \bigoplus_j V_g^{(j)}$, where $V_g^{(j)}$ is called irreducible representation that cannot be block-diagonalized any further [37, 9, 57]. The equivariance of our Φ suggests that $M^*(\mathbf{s}|\Phi)$ may be simultaneously block-diagonalizable as well. This block-diagonalization sometimes reveals disentanglement structure because any irreducible representation of $\mathcal{G}_1 \times \mathcal{G}_2$ is of form $V^{(1)} \otimes V^{(2)}$, where $V^{(k)}$ is an irreducible representation of $\mathcal{G}_k$. In particular, if M's irreducible representations have the form $V^{(1)} \otimes 1$ or $1 \otimes V^{(2)}$, then each block would either corresponds to the action of $\mathcal{G}_1$ or of $\mathcal{G}_2$.[3]

To find U that simultaneously block-diagonalizes all $M^*(\mathbf{s}|\Phi)$, we optimized U based on the following objective function that measures the block-ness of $\hat{M}^*(\mathbf{s}) := UM^*(\mathbf{s}|\Phi)U^{-1}$ based on the normalized graph Laplacian operator Δ:

$$\mathcal{L}_{\mathrm{bd}}(\hat{M}^*(\mathbf{s})) := \|\Delta(A(\hat{M}^*(\mathbf{s})))\|_{\mathrm{trace}} = \textstyle\sum_{d=1}^{a} \sigma_d(A(\hat{M}^*(\mathbf{s}))) \tag{6}$$

where $A(\hat{M}^*(\mathbf{s})) = abs(\hat{M}^*(\mathbf{s}))abs(\hat{M}^*(\mathbf{s}))^{\mathrm{T}}$ with $abs(\hat{M}^*(\mathbf{s}))$ represents the matrix such that $abs(\hat{M}^*(\mathbf{s}))_{ij} = |\hat{M}^*_{ij}|$. Our objective function is based on the fact that, if we are given an adjacency matrix A of a graph, then the number of connected components in the graph can be identified by looking at the rank of the graph Laplacian. For the derivation, please see Appendix E. Figure 5a shows the block structure revealed by simultaneous block-diagonalization through U trained by minimizing the average of $\mathcal{L}_{\mathrm{bd}}$ over all $\mathbf{s}$. Figure 5b shows the results of applying transformation of only one block. We can see that each block only alters one factor of variation. Our results suggest that the learned M^* captures the hidden disentangled structure of the group actions behind the datasets.

5.4 Extension to the sequences with constant acceleration

We have seen that our method successfully learns an equivariant structure from the set of constant-velocity sequences. In this section, we show that we can extend our concept to the set of sequences sharing the stationarity of higher order (constant acceleration). By definition, the pair of $\Phi(s_t)$ and $\Phi(s_{t-1})$ encodes the information about the velocity at t. When the multiplicity m is sufficiently large[4], the velocity can be estimated by: ${}^1M_t = \Phi(s_t)\Phi(s_{t-1})^{\dagger}$. Because this would yield a sequence of velocities, we can simply apply our method again to estimate the constant acceleration by ${}^2M^* = {}^1M_{+1}{}^1M_{+0}^{\dagger}$ where ${}^1M_{+0} = [{}^1M_2; ...; {}^1M_{T_c-1}] \in \mathbb{R}^{a \times (T_c-2)a}$ and

[3] $V : \mathcal{G} \to \{1\}$ is a valid irreducible representation for any $\mathcal{G}$.

[4] If m is less than a, we cannot obtain the pseudo inverse because of the rank deficient in $\Phi(s_{t-1})\Phi(s_{t-1})^{\mathrm{T}}$. Thus m should be at least larger than a.

(a) Generated images on *accelerated* Sequential MNIST.

(b) Prediction error.

Figure 6: Results on accelerated Sequential MNIST. Every model was trained with $T_c = 5$, $T_p = 5$. Neural transition overfitted and collapsed from $T_p = 5$ (beyond the training horizon).

${}^1M_{+1} = [{}^1M_3; ...; {}^1M_{T_c}] \in \mathbb{R}^{a \times (T_c - 2)a}$. We can then predict the future representation $\tilde{s}_t$ for $t = T_c + 1, ..., T_p$ by

$$\tilde{s}_t = \Psi\left(\left(\prod_{t'=T_c+1}^{t} {}^1\hat{M}_{t'}\right)\Phi(s_{T_c})\right) \text{ where } {}^1\hat{M}_{t'} = {}^2M^{*(t'-T_c)}\,{}^1M_{T_c}. \tag{7}$$

where $\prod$ represents multiplications from left. We train Φ and Ψ by minimizing the mean squared error between $\tilde{s}_t$ and s_t for $t = T_c + 1, ..., T$ as in Eq.(2). To create a sequence of constant acceleration from MNIST dataset, we only used shape and color rotations. We chose the initial velocity for these rotations randomly on the interval $[-\pi/5, \pi/5)$ for each sequence, and chose the acceleration on the interval $[-\pi/40, \pi/40)$. The results are shown in Figure 6. The Neural transition and the constant-velocity version of our method failed to predict the accelerated sequence, while the 2nd order model succeeded in predicting the sequence even after $t_p > 5$.

6 Discussion & Limitations

How is full equivariance achieved in our method? The theoretical results we provided in section 3.1 only assure that $M(g, x)$ and $M(g, x')$ are similar when the underlying group is commutative, compact and connected. However, as we have shown experimentally, our method seems to be learning Φ for which the estimators of M satisfy $M^*(\mathbf{s}(g, x)|\Phi) \cong M^*(\mathbf{s}(g, x')|\Phi)$. This can be happening because our framework and the training method based on the internal optimization in the latent space is somehow encouraging M^* to be orthogonal (See the loss curve of orthogonality of M^* in Appendix A). Maybe this is forcing the change of matrix P such that $PM(g, x)P^{-1} = M(g, x')$ to be also rotations as well, which commutes with $M(g, x)$ itself. Also, Figure 2b and 3 show that, as reported in [30], the models trained with reconstruction loss like (3) does not well capture the group transformation behind the sequences: the encoder representation was found to be significantly worse than that of the model trained with (2). We hypothesize that (3) fails to remove the sequence-specific information from M^*, while (3) succeeds to do so by training the model to be able to predict the unseen images.

Towards learning symmetries from more realistic observations As we are making a connection between our prediction framework and group equivariance, we are essentially assuming that the transitions are always invertible, because group is closed under inversion. However, this might not be always the case in real world applications; for instance, if the image sequences are the sequential renderings of a rotating 3D object, the transitions are generally not invertible because only a part of the object is visible at each time step. We experimented Sequential ShapeNet, which is created from ShapeNet [6] dataset. A series of rendered images is generated by sequentially applying 3D rotations of different speeds for each axis. Generated results on Sequential ShapeNet (See 23 in Appendix B show that actually our current method was not able to generate the images on 3D rotated datasets. If the transitions are not invertible, some measures must be taken in order to resolve the indeterminacy, such as probabilistic modeling or additional structural inductive bias.

Broader impact Because our study generally contributes to predictions and extrapolation, it has as much potential to negatively affect the society as most other prediction methods. In particular, applications of our method to image sequence can be potentially integrated into weapon systems, for example. At the same time, our unsupervised learning of the symmetrical structure from sequential datasets may also contribute to new discoveries in the systems of finance, medical science, physics and other fields of ML such as reinforcement learning.

References

[1] F. Alet, D. Doblar, A. Zhou, J. Tenenbaum, K. Kawaguchi, and C. Finn. Noether networks: meta-learning useful conserved quantities. *NeurIPS*, pages 16384–16397, 2021.

[2] P. Bevanda, S. Sosnowski, and S. Hirche. Koopman operator dynamical models: Learning, analysis and control. *Annual Reviews in Control*, 52:197–212, 2021.

[3] D. Bouchacourt, M. Ibrahim, and S. Deny. Addressing the topological defects of disentanglement via distributed operators. *arXiv preprint arXiv:2102.05623*, 2021.

[4] C. Burgess and H. Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

[5] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in β-vae. In *NeurIPS Workshop of Learning Disentangled Features*, 2017.

[6] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. URL `https://shapenet.org/terms`.

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *1597-1607*, 2020.

[8] M. Clausen and U. Baum. *Fast Fourier Transforms*. Wissenschaftsverlag, 1993.

[9] T. Cohen and M. Welling. Learning the irreducible representations of commutative lie groups. In *ICML*, pages 1755–1763, 2014.

[10] T. Cohen and M. Welling. Group equivariant convolutional networks. In *ICML*, pages 2990–2999, 2016.

[11] M. Connor and C. Rozell. Representing closed transformation paths in encoded network latent space. In *AAAI*, pages 3666–3675, 2020.

[12] B. Culpepper and B. Olshausen. Learning transport operators for image manifolds. *NeurIPS*, 22:423–431, 2009.

[13] N. Dehmamy, R. Walters, Y. Liu, D. Wang, and R. Yu. Automatic symmetry discovery with lie algebra convolutional network. *NeurIPS*, pages 2503–2515, 2021.

[14] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas. Vector neurons: A general framework for so(3)-equivariant networks. In *ICCV*, pages 12180–12189, 2021.

[15] L. Falorsi, P. de Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré, and T. S. Cohen. Explorations in homeomorphic variational auto-encoding. *arXiv preprint arXiv:1807.04689*, 2018.

[16] J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.

[17] K. Fukushima and S. Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[18] W. Fulton and J. Harris. *Representation Theory: A First Course*, volume 129. Springer Science & Business Media, 1991.

[19] K. Greff, F. Belletti, L. Beyer, C. Doersch, Y. Du, D. Duckworth, D. J. Fleet, D. Gnanapragasam, F. Golemo, C. Herrmann, T. Kipf, A. Kundu, D. Lagun, I. Laradji, H.-T. D. Liu, H. Meyer, Y. Miao, D. Nowrouzezahrai, C. Oztireli, E. Pot, N. Radwan, D. Rebain, S. Sabour, M. S. M. Sajjadi, M. Sela, V. Sitzmann, A. Stone, D. Sun, S. Vora, Z. Wang, T. Wu, K. M. Yi, F. Zhong, and A. Tagliasacchi. Kubric: a scalable dataset generator. In *CVPR*, 2022.

[20] S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. In *NeurIPS*, pages 15353–15363, 2019.

[21] J. Grosek and J. N. Kutz. Dynamic mode decomposition for real-time background/foreground separation in video. *arXiv preprint arXiv:1404.7592*, 2014.

[22] V. L. Guen and N. Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *CVPR*, pages 11474–11484, 2020.

[23] Y. Han, W. Hao, and U. Vaidya. Deep learning of koopman representation for control. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1890–1895. IEEE, 2020.

[24] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[25] O. J. Hénaff, A. Srinivas, J. D. Fauw, A. Razavi, C. Doersch, S. M. A. Eslami, and A. van den Oord. Data-efficient image recognition with contrastive predictive coding. In *ICML*, pages 4182–4192, 2020.

[26] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[27] A. Hyvarinen and H. Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *NeurIPS*, 2016.

[28] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[29] R. Kabra, D. Zoran, G. Erdogan, L. Matthey, A. Creswell, M. Botvinick, A. Lerchner, and C. Burgess. Simone: View-invariant, temporally-abstracted object representations via unsupervised video decomposition. *NeurIPS*, pages 20146–20159, 2021.

[30] T. Keller and M. Welling. Predictive coding with topographic variational autoencoders. In *ICCV workshop*, pages 1086–1091, 2021.

[31] T. Keller and M. Welling. Topographic vaes learn equivariant capsules. *NeurIPS*, pages 28585–28597, 2021.

[32] H. Kim and A. Mnih. Disentangling by factorising. In *ICML*, pages 2649–2658, 2018.

[33] T. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[34] T. Kipf, E. van der Pol, and M. Welling. Contrastive learning of structured world models. In *ICLR*, 2020.

[35] T. Kipf, G. F. Elsayed, A. Mahendran, A. Stone, S. Sabour, G. Heigold, R. Jonschkowski, A. Dosovitskiy, and K. Greff. Conditional object-centric learning from video. In *ICLR*, 2021.

[36] D. Klindt, L. Schott, Y. Sharma, I. Ustyuzhaninov, W. Brendel, M. Bethge, and D. Paiton. Towards nonlinear disentanglement in natural data with temporal sparse coding. In *ICLR*, 2021.

[37] I. R. Kondor. *Group theoretical methods in machine learning*. Columbia University, 2008.

[38] B. O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the national academy of sciences of the united states of america*, 17(5):315, 1931.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 25, 2012.

[40] A. S. Leahy. A classification of multiplicity free representations. 8:367–391, 1998.

[41] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. URL `http://yann.lecun.com/exdb/mnist/License`.

[42] Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–104. IEEE, 2004. URL `https://cs.nyu.edu/~ylclab/data/norb-v1.0-small/`.

[43] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *ICML*, pages 4114–4124, 2019.

[44] F. Locatello, B. Poole, G. Rätsch, B. Schölkopf, O. Bachem, and M. Tschannen. Weakly-supervised disentanglement without compromises. In *ICML*, pages 6348–6359, 2020.

[45] F. Locatello, M. Tschannen, S. Bauer, G. Rätsch, B. Schölkopf, and O. Bachem. Disentangling factors of variation using few labels. In *ICLR*, 2020.

[46] D. Pfau, I. Higgins, A. Botev, and S. Racanière. Disentangling by subspace diffusion. *NeurIPS*, pages 17403–17415, 2020.

[47] R. Quessard, T. Barrett, and W. Clements. Learning disentangled representations and group structure of dynamical environments. *NeurIPS*, pages 19727–19737, 2020.

[48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[49] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. *arXiv preprint arXiv:2112.05124*, 2021.

[50] J. Sohl-Dickstein, C. M. Wang, and B. A. Olshausen. An unsupervised algorithm for learning lie group transformations. *arXiv preprint arXiv:1001.1027*, 2010.

[51] A. Van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[52] A. Van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807, 2018.

[53] E. van der Pol, D. Worrall, H. van Hoof, F. Oliehoek, and M. Welling. Mdp homomorphic networks: Group symmetries in reinforcement learning. *NeurIPS*, pages 4199–4210, 2020.

[54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017.

[55] J. Von Kügelgen, Y. Sharma, L. Gresele, W. Brendel, B. Schölkopf, M. Besserve, and F. Locatello. Self-supervised learning with data augmentations provably isolates content from style. *NeurIPS*, 34, 2021.

[56] D. Wang, R. Walters, and R. Platt. So(2) -equivariant reinforcement learning. *arXiv preprint arXiv:2203.04439*, 2022.

[57] S. H. Weintraub. *Representation Theory of Finite Groups: Algebra and Arithmetic*, volume 59. American Mathematical Society, 2003.

[58] T. Yang, X. Ren, Y. Wang, W. Zeng, and N. Zheng. Towards building a group-based unsupervised representation disentanglement framework. In *ICLR*, 2021.

[59] S. Zhang, Y. Wang, and A. Li. Cross-view gait recognition with deep universal linear embeddings. In *CVPR*, pages 9095–9104, 2021.

[60] R. S. Zimmermann, Y. Sharma, S. Schneider, M. Bethge, and W. Brendel. Contrastive learning inverts the data generating process. In *ICML*, pages 12979–12990, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The abstract and introduction reflects the contributions of this paper.
   (b) Did you describe the limitations of your work? [Yes] In the Discussion & Limitations section, we discuss the type of dataset to which our work cannot be directly applied, as well as the theoretical problem that has not been solved.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We included the broader impact discussion in Section 6
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We read the guidelines and we confirmed that the contents in our paper conformed to them.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We included the full set of assumptions. We provide the details of the assumptions in the supplementary material.
   (b) Did you include complete proofs of all theoretical results? [Yes] We provide complete proofs of theoretical results in Supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provided the code for reproducing the experimental results as a supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We explained how we chose the hyperparameters, the data splits in section 5 and appendix section D.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We include the standard deviation values of equivariance performance in Appendix A (We omitted them in the main submission because of the space limitation) and the linear classification performance of the transition parameters in Appendix A.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We explained the number of GPUs to train the models and approximated costs for reproducing the full results of experiments in D.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] We cited [41] for MNIST, [6] for ShapeNet, [32] for 3Dshapes and [42] for SmallNORB.

(b) Did you mention the license of the assets? [Yes] For each dataaset we used, we included the URL to the license information in the reference section.

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] For ShapeNet dataset, we are approved to download the assets.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] We don't find any personally identifiable information in the images used in our experiments.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]