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 either about a dataset or a model. In this study, we investigate the symmetry that can be learned from time sequences of lengths 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 has shown the importance of symmetry-aware neural network models [16, 35, 32, 50]. 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 have succeeded in exploiting the knowledge of geometrical symmetries to improve the sample efficiency [49, 52] or to train a model that generalizes to unseen observations [46].

However, building an inductive bias that matches the given dataset of interest is challenging, and recent studies have been exploring ways of learning symmetries itself from observational sequences. Many of these approaches consider settings with relatively restrictive assumption or weak supervision. For example, [44] allows the trainer to use the knowledge of the identities of the actions used in making the transition. Meanwhile, [3, 27, 28] 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 certain stationary property (constant velocity, constant acceleration), we can learn the underlying symmetries by simply training the encoder model to be able to predict the future with linear transition in the latent space. We achieve this feat in the way of meta-learning, which consists of (1) the internal training step to compute the prediction loss of linear transition in the latent space and (2) the external training step to update the encoder. We train our

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

model by splitting each observation sequence into two components that are respectively dedicated to the internal and external step. Our experimental results on synthetic and benchmark dataset indicate that our method can capture the equivariance relation, thereby succeeding to extrapolate the unseen sequences far into the future. We will also show that, in alignment to group representation theory, the learned linear transitions in our framework can be simultaneously block-diagonalized over all sequences in the dataset, 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 data-driven manner. There are rich literatures 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, object centric approaches introduces inductive bias in the form of architecture, and equips the model with pre-defined slots to be allocated to objects [30, 26, 31]. Meanwhile, [19, 1] assumes that the symmetry to be found takes the form of the energy conservation law, and learns each variables in the law as a function of observations. While this approach assumes that some *energy* is preserved in each observation, we assume that velocity/acceleration is preserved within each observation. Other more indirect forms of inductive bias include those relevant to distributional sparseness and symmetry defined through algebraic constraints. [24] for instance assumed that every stationary component of a given time series is generated by a finite and independent latent time series. [33] proposed to model the transition sparsely 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 [34, 20, 2]. However, 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.

Unsupervised learning with algebraic/geometrical constraints Many studies impose algebraic constraints that reflects some form of geometrical assumption. [14] uses a known coordinate map parametrization of a Lie group family to construct a posterior distribution on the manifold. [43] 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, [10, 47, 8, 44, 9] make some Lie group type assumption about the transition. [44] also assume that the identity of the actions in the sequences are known. As for the approaches with less explicitly geometrical touch, [27] uses capsule structure in their probabilistic framework to model a finitely cyclic structure while retaining the computability of posterior distribution. By design, [27] assumes that all sequences in the dataset transitions with the same cyclic velocity. [54] 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 [44, 9, 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 [23] 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 [39]. In response to this work, subsequent works have explored different frameworks such as weakly-/semi-supervised settings [41, 40] and learning on sequential examples [33] to provably learn disentangled representations. ICA [25] and recent works [55, 51] also discuss the identifiebility property of learned representations. Classical methods like [23, 13, 29] take an approach of incorporating the inductive bias in the form of probabilistic model.

Our method achieves disentanglement as a by-product of training a predictor model with linear algebraic constraint. As we mentioned in the introduction, the set of latent linear transformation estimated by our method for different time sequences can be simultaneously block-diagonalized, and

the latent space each block corresponds to a disentangled feature. We achieve this disentanglement with a data assumption that each sequence transitions with constant velocity/acceleration. Our data assumption might be similar in taste to the setting used by [25], in which the observed time series can be split into finite number of stationary components.

3 Learning of Data 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 are enabling such a feat by analyzing a short, past time-frame with a certain stationary property (e.g. constant velocity/accelerataion). Indeed, only a handful of people with good dynamic visual acuity can chase a fast moving object, because this ability allows them to identify such a 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. More precisely, we suggest to learn the structure of a given dataset $\mathcal{X}$ by using a set of time-series in $\mathcal{X}$ that are transitioning in the same way at each time step. We formalize this idea below.

Dataset structure Our dataset $\mathcal{S}$ consists of time series, so that each member $\mathbf{s} \in \mathcal{S}$ takes form $\mathbf{s} = [s_k \in \mathcal{X}; k = 1, ..., T] \in \mathcal{S}$. Because we want $\mathbf{s}$ to be describing a time-series that transitions at constant velocity, we assume that all s_k in a given instance of $\mathbf{s} \in \mathcal{S}$ are related by a fixed transition operator $g : \mathcal{X} \to \mathcal{X}$ so that $s_{t+1} = g \circ s_t$ for all t. We use G to denote the set of transition operators, and we assume that $\mathcal{X}$ is closed under G; that is, $g \circ x \in \mathcal{X}$ for all $x \in \mathcal{X}, g \in G$. We would allow 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 [27] that explores a cyclic structure using the capsules of same size. We emphasize that the transition action g generally differs across the different members of $\mathbf{s}$. For example if G is a set of rotations and $\mathcal{X}$ a set of images, then the rotational speed , direction and the initial image may be all different for two distinct instances of $\mathbf{s}$ 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 inital frame s_1 and transitions with g. We use $\mathcal{G}$ to denote the set of operators that are used to generate $\mathcal{S}$. Summarizing, $\mathcal{S}$ is a subset of $\{[g^k \circ s_1; k = 1, ..., T-1]; s_1 \in \mathcal{X}, g \in \mathcal{G}\}$.

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

$$M_g \Phi(x) = \Phi(g \circ x) \tag{1}$$

for all $x \in \mathcal{X}$ and $g \in \mathcal{G}$. In other words, $\mathcal{X}$ and $\mathbb{R}^{a \times m}$ are invertibly related by an equivariance relation with respect to $\mathcal{G}$, where $\mathcal{G}$ acts on $H \in \mathbb{R}^{a \times s}$ via the action $g \circ_{latent} H = M_g H$. In this framework, we predict the sequence $\mathbf{s}(s_1, g)$ with the relation $\Phi^{-1}(M_g \Phi(s_{t-1})) = \hat{s}_t$. Note that when Φ is a linear function and $\mathcal{G}$ is a group, this implies that $\mathcal{X}$ is a representation space of G and M is a representation of G; in other words, M is a map such that $M_g M_h = M_{gh}$ for all $g, h \in G$ [7, 53, 8]. This relation is automatically forced from (1) because $M_{gh}\Phi(x) = \Phi(gh \circ x) = M_g \Phi(hx) = M_g M_h \Phi(x)$ for all x. We also 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 assume that there are m number of subspaces that react to g in the same way. We assume $m > 1$ in our study, as 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 [7, 17], and is known to be a special case that happens only under a restrictive condition on the dimension of the observations space [36]. A similar idea has been used in model architecture as well [12]. Because we are proposing 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 [27], which encodes a predefined cyclic structure in the model. Figure 1 describes our overall framework.

3.1 Learning equivariance relation from stationary sequential dataset

How does the stationary property (constant velocity assumption) help us find a representation satisfying the equivariance relation (1)? As we elaborate in the next section, we train our Φ by requiring that, for each time sequence $\mathbf{s}(s_1, g)$, there exists some "sequence"-common matrix $M(g, s_1)$ such that $\Phi(s_t) = M(g, s_1)\Phi(s_{t-1})$. On its own, this requirement seems to only enforce that the latent transition M in a given sequence is well defined; that is, $M(g, x) = M(g, g \circ s_1) = M(g, g^2 \circ s_1) \cdots$ and so on. Indeed, this is insufficient for the full equivariance (1), because we want $M(g, x)$ to be independent of x, thereby achieving $\Phi(g \circ x) = M(g, x)\Phi(x) = M_g\Phi(x)$ with well-defined M_g. It turns out that, with continuity assumption on M and an assumption on the choice of G, this properties can be partially satisfied under certain regularity conditions. We say that M satisfies **intra-orbital homogeneity** if $M(g, x)$ is constant on the orbit $G \circ x = \{g \circ x; g \in G\}$ for each x. The next proposition shows that the intra-orbital homogeneity is satisfied if G is sufficiently regular.

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 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 $G \circ x \neq G \circ x'$ can be shown to be at least similar.

Proposition 3.2. *Suppose that $M(g, x)$ satisfies intra-orbital homogeneity, and suppose that 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 homogeneity properties can be satisfied automatically by training the reresentation on the set of stationary sequences. Interestingly, as we experimentally demonstrate later, our training method that we will elabote in the next section successfully learns a fully equivariant Φ without explicitly enforcing the change of basis P to be I, suggesting that our method's scope is beyond these theoretical results.

4 Encoding Algebraic Structure by Meta-optimization

We would 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 $g \in G$, $s_1 \in \mathcal{X}$. We propose to 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 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 for all $\mathbf{s}$ at all time t.

We propose to 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:

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

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 $\mathbf{s}_p$ makes a substantial difference in the training process. This is most likely because the minimization of the validation error of M^* optimized on $\mathbf{s}_c$ would encourage Φ to exclude the $\mathbf{s}_c$-specific information from the transition M^*. We will illustrate this effect in the experimental section. We shall note that we can also parametrize 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 [11] for building Lie group convolutional network and for [47, 9] in predicting a sequence that is not necessarily stationary. In order to train their model, [47] used additional parameters to diagonalize each algebra element as well as hyper parameters to stabilize the training. We also experimented with 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. Our internal optimization procedure is free of such a parameter tuning.

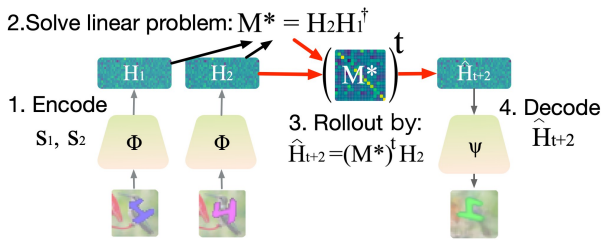


Figure 1: Prediction with our method when $T_{\mathrm{c}} = 2$ and $T_{\mathrm{p}} = t$. After the encoder encodes the observations into tensor representations $H_1, H_2 = \Phi(s_1), \Phi(s_2)$, the method solves the least square problem: $M^* = \arg\min_M \|MH_1 - H_2\|_F^2$. The model then predicts the future observations by $\tilde{s}_{t+2} = \Psi((M^*)^t H_2)$. These processes (including the linear problem) are all differentiable.

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

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

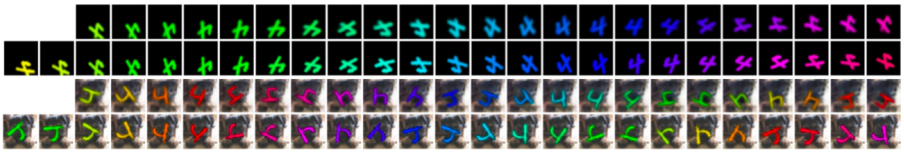
where $H_{+0} = [\Phi(s_1); ...; \Phi(s_{T_{\mathrm{c}}-1})] \in \mathbb{R}^{a \times (T_{\mathrm{c}}-1)s}$ and $H_{+1} = [\Phi(s_2); ...; \Phi(s_{T_{\mathrm{c}}}) \in \mathbb{R}^{a \times (T_{\mathrm{c}}-1)s}$ are the horizontal concatenation of the encoded frames and $H_{+0}^{\dagger}$ is the Moore-Penrose pseudo inverse of H_{+0}. Because $M^*(\mathbf{s}_{\mathrm{c}}|\Phi)$ is a closed form with respect to Φ, the loss (2) can be directly optimized by differentiating with respect to the parameters of both Φ and Ψ. Thus, the training is done in an end-to-end manner. Figure 1 and summarizes the overall procedure to make prediction on a given sequence. 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.

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 the appendix section D. We tested our framework on *Sequential* MNIST, 3DShapes [4], and SmallNORB [38]. Sequential MNIST is created from the MNIST [37] dataset. For all experiments, we used a ResNet [21]-based encoder-decoder architecture and we set $a = 16$ and $m = 256$ so that the latent space lives in $\mathbb{R}^{16 \times 256}$.

For sequential MNIST, we chose our G to be the set of all combinations of three types of transformations: shape rotation, hue rotation, and translation (See Appendix D for the examples of sequences), and randomly sampled one instance of $g \in G$ for each sequence. To create each sequence, we first resized the MNIST image to 24×24, applied repetitions of a randomly sampled, fixed member of $g \in G$ 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 the straight line between the sampled points. We also experimented on sequenaital MNIST with background (We call it 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 [45] dataset, 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 *digit4 only* training.

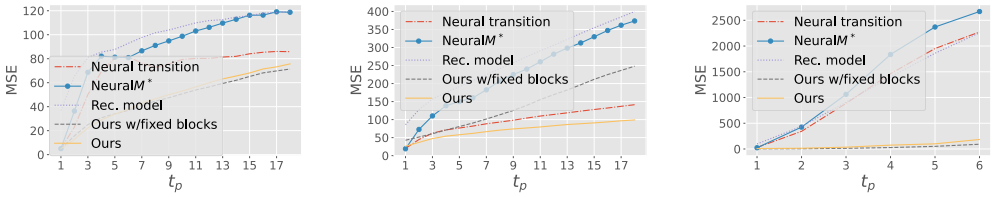
3DShapes and SmallNORB are datasets with multiple factors of variation. We created a set of *constant-velocity* sequences from these datasets by varying a fixed combination of factors for each sequence. That is, on these datasets, we chose our G to be the set of variations of factors, and sampled each 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 i-th factor direction in 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 the sequence, we set $T_{\mathrm{c}} = 2$ and $T_{\mathrm{p}} = 1$ on all of



(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 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 blocked, 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 $T_{\mathrm{p}} = t_{\mathrm{p}}$. During training phase, models are trained to predict the observations only at $t_{\mathrm{p}} = 1$. The prediction errors at $t_{\mathrm{p}} > 1$ indicate extrapolation performance. The results on SmallNORB can be found in the appnedix section A.

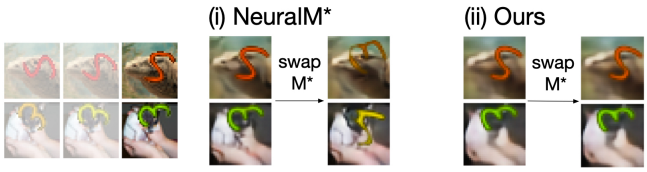
the constant velocity experiments. We also conducted experiments for the sequences with constant acceleration on sequential MNIST. To create a sequence with constant acceleration, we chose a pair $g_a, g_v \in G$ for each sequence, and generated $\mathbf{s}$ by setting $s_{t+1} = g_a^t g_v s_t$. We elaborate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

As ablations, we tested several variants of our method: **Proposed method with fixed 2x2 blocks**(When we perform internal optimization, we seperated the latent tensor $H \in \mathbb{R}^{16\times 256}$ into 8 subtensors $\{H_k \in \mathbb{R}^{2\times m}\}_{k=1}^{8}$ and calculated pseudo inverse for each k), **Reconstruction model**(Training Φ and Ψ with $\mathcal{L}_r$ with $T = 3$), **Neural*M****(Abbreviated as Rec. Model. Uses an additional neural network to map $\mathbf{s}_{\mathrm{c}}$ to M^*), and **Neural transition**(Autoregressive model on $\Phi(s_t)$)[1]. In testing these variants, we used the same pair of encoder and decoder architecture. For NeuralM^*, we trained an additional network M_θ that takes a pair of observations to output the transition matrix, replaced M_θ with M^* in (4), and optimized θ and Φ, Ψ simultaneously. For Neural transition model, we trained 1x1 1D-convolutional networks on the $\Phi(s)$s to produce the latent tensor in the next time step: For instance, $\hat{H}_{t+1} = \mathrm{1DCNN}(\Phi(s_t), \Phi(s_{t-1}))$ when $T_{\mathrm{c}} = 2$. The 1DCNN was applied along the multiplicity dimension (m).

5.1 Qualitative and quantitative results on the prediction

Figure 2 shows the example sequences generated by the proposed model and comparative models. The graph value at t_p represents the prediction performance at $T_{\mathrm{c}} + t_p$ where $T_{\mathrm{c}} = 2$. Thus, during the training, the prediction of the error at $t_p = 1$ is back-propageted to the encoder, and the prediction at $t_p > 1$ is an extrapolation. Our method successfully predicts the images for $t_{\mathrm{p}} \geq 1$. Neural transition and NeuralM^* models had almost the same prediction performance at $t_{\mathrm{p}} = 1$, but they both failed in extrapolation. Also, our method failed in extrapolation when we parametrized M with fixed block structure. This might be because over-regularization hindered with the training of SGD optimization [15]. The training however succeeded when we left more freedom for M.

[1]This model can be seen simplified version of [42]



(a) Dependency of M^* on sequence. The 2x3 tiled images in the left most panel represents two sequences $\mathbf{s}^{(1)}, \mathbf{s}^{(2)}$ with same transition action yielding $M^{*(1)}, M^{*(2)}$ respectively. Neural M^* fails in prediction when $M^{*(2)}$ is used to predict s_1. Our method does not fail by this swap, indicating $M^{*(2)} \cong M^{*(1)}$.

Method	MNIST		MNIST-bg		3DShapes		SmallNORB	
	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$	$\mathcal{L}^p$	$\mathcal{L}^p_{\text{equiv}}$
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

Figure 4: Equivariance performance. Our method trained more equivariant model than all other rivals. We set $T_{\text{c}} = 2, T_{\text{p}} = 1$ for the calculation of $\mathcal{L}^p$(Eq.(2)) and $\mathcal{L}^p_{\text{equiv}}$(Eq.(5)). To evaluate equivariance errors on more difficult settings, we used all of digits in Sequential MNIST-bg for training and test.

To evaluate the wellness of the representation learned by our method, we also predicted the factors of transition from M^* (Figure 8 in the appendix A). SimCLR [6] and Contrastive predictive coding (CPC) [48, 22] are tested as baselines. Please see the appendix D for the detailed experimental settings for SimCLR and CPC. Our method yields better representation than all comparative methods on both MNIST and MNIST-bg datasets.

5.2 Equivariance performance

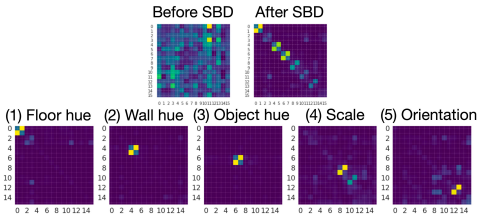
As we mentioned in the 3.1 section, by the construction of our framework, the equivariance is achieved when $M(g, x)$ does not depend on x. We therefore do not want its estimator $M^*(\mathbf{s}(s_1, g)|\Phi)$ to depend on s_1, where we recall that $\mathbf{s}(s_1, g)$ represents the sequence that begins with s_1 and transitions with g. To see how much the trained model is equivariant to the transformation underlying the sequence, we therefore calculated *equivariance error* as the prediction error from applying $M^*(\mathbf{s}|\Phi)$ to $\Phi(s'_2)$ for a pair of $\mathbf{s} \neq \mathbf{s}'$ that transitions with common g. In other words, we compute the following;

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

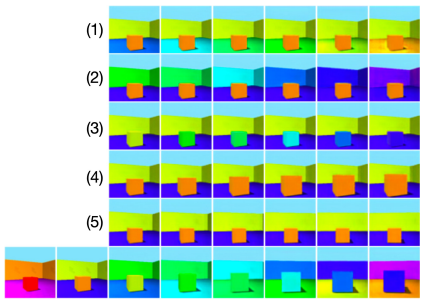
where $\mathcal{S}(g)$ represents the set of all sequences that transitions with g. For each of $\mathbf{s}, \mathbf{s}'$ we set $T_{\text{c}} = 2$ and $T_{\text{p}} = 1$ as done in the experiments in the previous sections. Figure 4 compares the NeuralM^* methods against our method in terms of equivariance 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 swap the images; this implies that M^* learned by NeuralM^* encodes *sequence* specific information together with the transition. On the other hand, swapping of M^* learned by our method does not affect the prediction, 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 inter-homogeneity ($P = I$ in Section 3.1).

5.3 Measuring the block-ness by Simultaneous diagonalization of M^*s

So far, we have observed that our model has an ability to learn G-equivariant model. Representation theory gurantees that, if G is a compact connected group, any representation $D : G \to \mathbb{R}^{a \times a}$ can be simultanesouly block-diagonalized; that is, there is a common change of basis U such that $UD_gU^T = \bigoplus_j V_g^{(j)}$, where $V_g^{(j)}$ is called irreducible representation that cannot be block-diagonalized any further. We may have $V_g^{(j)} = 1 \in \mathbb{R}^{1 \times 1}$ for all g for some (j), because



(a) Simeltaneously block-diagonalized matrices



(b) Disentangled transition representation on 3DShapes

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

$V : G \to \{1\}$ is a valid irreducible representation for any G. 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 $G_1 \times G_2$ is of form $V^{(1)} \otimes V^{(2)}$, where $V^{(k)}$ is an irreducible representation of 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 G_1 or of G_2.

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 $\hat{M}^*(\mathbf{s}) := U^{-1}M^*(\mathbf{s}|\Phi)U$ based on the normalized graph Laplacian operator Δ:

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

where $\hat{M}^*(\mathbf{s}) = abs(\hat{M}^*(\mathbf{s}))abs(\hat{M}^*(\mathbf{s}))^T$ with $abs(\hat{M}^*(\mathbf{s}))$ representing the matrix such that $abs(M^*(\mathbf{s}))_{ij} = |M^*_{ij}|$. 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 graph Laplacian. For the derivation, please see the appendix E. Figure 5a shows the block structure revealed by simultaneous block-diagonalization through U trained by minimizing U with respect to the the average of $\mathcal{L}_{\mathrm{bd}}$ over all $\mathbf{s}$. Figure 5b shows the results of applying just transformation of one block. We can see that each block only alters one factor of variation. Our results suggest that the learned M^* captures the 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, 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 ${}^1M_{+0} = [{}^1M_2; ...; {}^1M_{T_c-1}] \in \mathbb{R}^{a\times(T_c-1)s}$ and ${}^1M_{+1} = [{}^1M_3; ...; {}^1M_{T_c}] \in \mathbb{R}^{a\times(T_c-1)s}$. We can then predict the future representation $\tilde{s}_t$ for $t = T_c + 1, ..., T_p$ by

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

(a) Generated images on *accelerated* Sequencial MNIST. (b) Prediction error.

Figure 6: Results on accelerated sequential MNIST. Every model was trained with $T_{\mathrm{c}} = 5$, $T_{\mathrm{p}} = 5$. Neural transition overfitted and collapsed from $t_p = 5$ (beyond the training horizon).

where $\prod$ represents multiplications from left. We train Ψ and Ψ by minimizing the mean squared error between $\tilde{s}_t$ and s_t for $t = T_{\mathrm{c}} + 1, ..., T_{\mathrm{p}}$ 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 he accelerated sequence, while the 2nd order model succeeded in predicting the sequence even after $t_{\mathrm{p}} > 5$.

6 Discussion & Limitations

How is full equivariance achieved in our method? The theoretical results we provided in section 3.1 only assures that $M(g, x)$ and $M(g, x')$ are similar when the underlying group is commutative, compact and connected. However, as we have shown in experiment, our method seems to be learning Φ for which the estimators of M satisfies $M^*(\mathbf{s}(g, x)|\Phi) \cong M^*(\mathbf{s}(g, x')|\Phi)$. 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 the 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 [28], 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 [5] 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 13 in the appendix B shows that actually the our current method was not able to generate the images on 3D rotation 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.

7 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 dataset 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] 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`.

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

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

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

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

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

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

[12] 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.

[13] U. disentangling in -VAE. C.p.burgess and i.higgins and a.pal and l.matthey and n.watters and g.desjardins and anda.lerchner. *NeurIPS Workshop of Learning Disentangled Features*, 2017.

[14] 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.

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

[16] 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.

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

[18] 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.

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

[20] 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.

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

[22] 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.

[23] 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.

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

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

[26] 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.

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

[28] T. A. Keller and M. Welling. Predictive coding with topographic variational autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1086–1091, 2021.

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

[30] T. Kipf, E. van der Pol, and M. Welling. Contrastive learning of structured world models. In *ICLR*, 2020.
[31] 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.
[32] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
[33] 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.
[34] 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.
[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 25, 2012.
[36] A. S. Leahy. A classification of multiplicity free representations. 8:367–391, 1998.
[37] 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`.
[38] 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/`.
[39] 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.
[40] F. Locatello, B. Poole, G. Rätsch, B. Schölkopf, O. Bachem, and M. Tschannen. Weakly-supervised disentanglement without compromises. In *ICML*, pages 6348–6359, 2020.
[41] F. Locatello, M. Tschannen, S. Bauer, G. Rätsch, B. Schölkopf, and O. Bachem. Disentangling factors of variation using few labels. In *ICLR*, 2020.
[42] A. v. d. 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.
[43] D. Pfau, I. Higgins, A. Botev, and S. Racanière. Disentangling by subspace diffusion. *NeurIPS*, pages 17403–17415, 2020.
[44] R. Quessard, T. Barrett, and W. Clements. Learning disentangled representations and group structure of dynamical environments. *NeurIPS*, pages 19727–19737, 2020.
[45] 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.
[46] 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.
[47] J. Sohl-Dickstein, C. M. Wang, and B. A. Olshausen. An unsupervised algorithm for learning lie group transformations. *arXiv preprint arXiv:1001.1027*, 2010.
[48] A. Van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807, 2018.
[49] 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.
[50] 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.
[51] 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.
[52] D. Wang, R. Walters, and R. Platt. So(2) -equivariant reinforcement learning. *arXiv preprint arXiv:2203.04439*, 2022.
[53] S. H. Weintraub. *Representation Theory of Finite Groups: Algebra and Arithmetic*, volume 59. American Mathematical Society, 2003.
[54] T. Yang, X. Ren, Y. Wang, W. Zeng, and N. Zheng. Towards building a group-based unsupervised representation disentanglement framework. In *ICLR*, 2021.
[55] 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

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or

[N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

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 the section 7
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We
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 hyper parameters, the data splits in the section 5 and the 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 the appendix section A (We omitted them in main submission because of the space limitation) and the linear classification performance of the transition parameters in the 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 [37] for MNIST, [5] for ShapeNet, [29] for 3Dshapes and [38] 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]