Université de Montréal

Provably Learning Disentangled & Compositional Models

par

Divyat Mahajan

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Proposition de sujet de thése en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en Informatique

August 2024

Abstract

Despite significant advances in representation learning, success in out-of-distribution (OOD) generalization and reasoning remains elusive. There is growing interest in incorporating causal principles into representation learning as a principled approach to tackle these challenges. Causal factorization helps with efficient OOD generalization as it enables us to model distribution shifts as sparse mechanism shifts, hence we can adapt faster to them. However, a major challenge is that the causal factors responsible for the data generation process are not directly observable. Therefore, it is essential to learn representations that disentangle these latent (causal) factors of variation from high-dimensional observations. This thesis proposal aims to explore the various frameworks that identify latent factors with theoretical guarantees and develop practical approaches that leverage them for constructing reliable machine learning models.

First, we build on existing work that utilizes auxiliary information for disentangled representation learning, focusing on a multi-task supervised learning framework. We demonstrate that it is possible to identify latent variables even when labels do not render the latent variables conditionally independent, challenging this common assumption from previous research. We then shift our focus to unsupervised disentanglement, introducing a class of decoders inspired by object-centric learning methods, which we term as additive decoders. Our findings show that additive decoders can disentangle latent variables under minimal assumptions on their distribution, without the need for any weak supervision. Unlike prior works, we establish a formal connection between disentangled factors and their compositional ability when using additive decoders.

Finally, we propose future directions that extend additive decoders to more flexible additive energy models. Preliminary results suggest that additive energy models can achieve compositional generalization with discrete factors, offering potential benefits for the task of group-robust prediction.

Keywords: Disentangled Representation Learning, Compositional Generalization, Distribution Shifts, Latent Identification, Structural Causal Models

Contents

Introduction

Representation learning [Bengio et al., 2013] aims to extract low dimensional representations from high dimensional complex datasets. The underlying goal is that if these representations effectively capture the factors of variation that describe the high-dimensional data (e.g., features characterizing the shape of an object in an image), they can be utilized to achieve strong performance on new downstream tasks with minimal supervision. The success of large-scale pre-trained models highlights the effectiveness of representation learning approaches [Brown et al., 2020, Wei et al., 2021, Radford et al., 2021]. However, its important to approach these results with caution, as neural networks have been shown to fail often at out-of-distribution generalization [Geirhos et al., 2020, Ibrahim et al., 2022, Hsieh et al., 2024].

To address the out-of-distribution generalization failures, recent works [Schölkopf, 2019, Schölkopf et al., 2021] have advocated for incorporating causal principles into standard training paradigms; supervised [Arjovsky et al., 2019], self-supervised [Von Kügelgen et al., 2021], and unsupervised [Kocaoglu et al., 2018, Pawlowski et al., 2020]. The core issue is that the current deep learning paradigm does not inherently incorporate and leverage key causality principles [Pearl, 2009], such as the invariance principle, the principle of independent causal mechanisms, and causal factorization. This is because the traditional causal inference requires access to structured random variables whose distributions can be decomposed using causal factorization, which is impossible with complex datasets such as images or text.

Therefore, to harness the power of deep learning and causal principles, it is essential to first disentangle the factors of variation from low-level observations to obtain the causal representations that generated the data. Hence, my Ph.D. research is centered on determining the appropriate assumptions required to disentangle latent (causal) variables from data with theoretical guarantees, and on developing practical approaches that utilize these disentangled variables to construct reliable machine learning models with reasoning capabilities.

This thesis proposal first provide reader a background on causality and disentangled representation learning Chapter 1, where we consider the overall research problems and discuss recent trends in this area. Section 1.2 describes a general strategy for latent variable identification with Auto-Encoders by constraining the mixing function and the latent distribution; with linear ICA as an example where latent variables can be "disentangled". Further, given the challenges with unsupervised learning of disentangled representation [Hyvärinen and Pajunen, 1999, Locatello et al., 2019], we discuss in detail the use of (weakly) supervised setups for disentanglement in Section 1.2.3.

In Chapter 2 we build further on the task of disentangled representation learning with auxiliary information, focusing on labeled data as the source of supervision. Prior works for this problem [Hyvarinen et al., 2019, Khemakhem et al., 2020a] operate with the assumption that the latent factors are conditionally independent given the auxiliary information (labels), and they also require a lot of auxiliary information. Hence, we provide identification results for a class of models where auxiliary information does not ensure conditional independence, and also show experimentally that disentanglement (to a large extent) is possible even when the auxiliary information dimension is much less than the dimension of the true latent variables.

Chapter 3 returns back to the question of unsupervised disentangled representation learning, with inspiration from decoders used in object-centric learning [Locatello et al., 2020c, Dittadi et al., 2021]. We provide latent identification guarantees for a class of decoder that we call additive, which decompose an image into a sum of object-specific images. Our identification result does not make any parametric assumption of distribution of latent factors, only minimal assumptions on their support. Unlike prior works on disentangled representation learning, we also prove the compositional abilities of the disentangled factors, termed as Cartesian-product extrapolation. Specifically, we show that the decoder can generate novel images outside the support of training data via novel combinations of the disentangled factors.

A key assumption made for Cartesian-product extrapolation is that the latent factors are continuous variables, while many practical scenarios might require us to be compositional in terms of discrete factors. Further, the extrapolation relies on additive decoders, which are limited in their ability to model complex scenes with occlusions between objects. In Chapter 4, we discuss ongoing and future efforts for compositional generalization (extrapolation) with more expressive additive energy models and discrete factors. We characterize the conditions on the support of training distribution of discrete factors for compositional generalization, and provide both theoretical and empirical evidence that additive energy models can extrapolate to novel compositions. Our current results assume that the latent factors are known as the focus is solely on compositional generalization; and the future goal is to extend this for both disentanglement & compositionality with additive energy models.

Notations. We denote random variables using uppercase letters (X) while their corresponding realizations are denoted with lowercase letters (x). We denote the distribution of the X as $\mathbb{P}_X$ or $p(X)$, and the support of X using the set notation ($\mathcal{X}$).

Another important convention is that we use C^k-function to denote a function that is differentiable k times with continuous derivatives. Similarly, C^k-diffeomorphism is used to denote a C^k-function where the inverse of the function is also C^k. Also, $[m]$ is used to denote the set of positive integers $\{1, \cdots, m\}$.

Finally, Since some notations for the task of disentangled representation learning will be repeated across several chapters in this report, we summarize them in this table below.

$X \in \mathbb{R}^{d_x}$	Observations
$Z \in \mathbb{R}^{d_z}$	Latent Variables
$\mathcal{X} \subseteq \mathbb{R}^{d_x}$	Support of X
$\mathcal{Z} \subseteq \mathbb{R}^{d_z}$	Support of Z
$g : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$	Ground-truth Decoder
$f : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$	Ground-truth Encoder
$\hat{g} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$	Learned Decoder
$\hat{f} : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$	Learned Encoder

Table 1: Recurrent Notation for Disentangled Representation Learning

Chapter 1

Background

1.1 Cause-Effect Models

Several decision-making tasks require us to compute the effect of performing interventions; "how does changing one component of a system affect the other components?". An approach to infer interventional effects is to conduct controlled experiments where we only change the variable of interest, which might not be feasible in several practical scenarios. Hence, learning the causal generative process from observations is a fundamental problem in several scientific domains [Sachs et al., 2005, Foster et al., 2011, Xie et al., 2012], as it offers a comprehensive understanding of the data generation process (DGP), and allows to simulate the effect of controlled experiments/interventions. These desirable properties can even accelerate scientific discoveries by reliably predicting the effects of unseen interventions, eliminating the need for laboratory experiments [Ke et al., 2023, Zhang et al., 2024]. Further, understanding the causal mechanisms behind the data generation process helps in robust representation learning as it provides a principled solution to tackle out-of-distribution (OOD) generalization [Schölkopf et al., 2021], and allow for faster adaptation to distribution shifts [Bengio et al., 2019].

1.1.1 Fundamental Concepts

A popular approach for modeling causal processes is the structural causal model (SCM) framework [Peters et al., 2017] where causal mechanisms are modeled via structured functional relationships. Given a set of random variables $V = \{X_1, \cdots, X_d\}$, an SCM define each causal variable X_i as a function of a subset of the other causal variables $(V/\{X_i\})$

Definition 1.1.1 (Structural Causal Model (SCM))**.** Given a set of endogenous (causal) variables $V = \{X_1, \cdots, X_d\}$ and exogenous (noise) variables $U = \{N_1, \cdots, N_d\}$ sampled from distribution $p(U)$, SCM defines the causal relationship for each X_i as follows:

$$X_i = f_i(PA_i, N_i) \text{ s.t. } PA_i \subset V \ , \ X_i \notin PA_i$$

The distribution over exogenous variables $p(U)$ and the causal mechanisms/relationships $\{f_i\}$ define a unique distribution over the endogenous variables $p(V)$. When we have mutually independent noise variables (U), then SCM is called a markovian SCM. For the rest of the report, we would deal with markovian SCMs as the non-markovian counterparts complicate even basic operations like interventions by violating the independent causal mechanism principle, which will be explained later.

Further, every SCM implies a directed graph (V, E), also known as **causal graph**, where the nodes are the set of causal variables V and there is a directed edge $(j \to i \in E)$ whenever X_j causes X_i, i.e, $X_j \in PA_i$. The causal graph may contain cycles, which happens when SCMs model the equilibrium state of a dynamical system. However, for most problems we assume the causal graph to be directed acyclic graph (DAG), which allows us to connect SCMs with *bayesian networks*. Lets denote the bayesian network associated with the causal graph as a **causal bayesian network**, then the joint distribution over the causal variables $(p(V))$ given by the SCM factorizes as follows:

$$p(X_1, \cdots, X_d) = \prod_{i=1}^{d} p(X_i|PA_i) \quad \text{where } \ p(X_i|PA_i) \ \text{ is specified by } \ f_i(PA_i, N_i) \tag{1.1}$$

Note that a bayesian network implies a family of distributions that factorize in a particular way; hence the above equation states that the distribution (p_V) given by the SCM belongs to the family of distributions associated with the causal bayesian network. Alternatively, an SCM implies a unique causal graph, however, *a causal graph implies multiple SCMs* since it allows a family of distributions.

Independent Causal Mechanisms (ICM). This principle states that each causal mechanism $(f_i(PA_i, N_i))$ is independent of the other causal mechanisms, i.e., knowledge about one mechanism does not provide any information about the other causal mechanisms. This is fundamental assumption for causal models which enables us to perform interventions as described ahead. Note that ICM also implies that the causal graph satisfies the local markov property in bayesian networks, i.e., each causal variable is independent of its non-descendants given its parents, $X_i \perp X_{ND_i/PA_i} \mid PA_i$. Note that for non-markovian SCMs this assumption would not be satisfied, since the noise variables across causal variables are correlated.

Interventions in SCM. Interventions alter the underlying causal relationships of a variable and SCMs allow us to formally characterize them and infer their effect. We say that a variable X_i has been intervened if the corresponding causal mechanism has been changed, i.e, $X_i = \tilde{f}_i(\tilde{PA_i}, \tilde{N}_i)$, represented as $do(X_i = \tilde{f}_i(\tilde{PA_i}, \tilde{N}_i))$. Note that intervening on X_i will not change the other causal mechanisms due to the ICM principle, hence interventions are modular. Interventions that lead to $PA_i = \{\phi\}$ are called *perfect interventions*, and they correspond to removing all the incoming edges to X_i in the causal graph. A special case of perfect interventions are *hard interventions*, where we set the intervened variable to a specific value ($do(X_i = a)$). The effect of an intervention $do(X_i)$ on another variable X_j is defined as $\mathbb{E}[X_j|do(X_i)] = \int X_j \; p(X_j|do(X_i)) \; dX_j$, i.e., the expected value of X_j in intervened SCM. If X_i is a binary variable, then the average treatment (causal) effect of X_i on X_j is defined as the difference in the interventional effects for $do(X_i) = 0$ and $do(X_i) = 1$.

$$\text{Average Treatment Effect (ATE):} \quad \mathbb{E}[X_j|do(X_i = 1)] - \mathbb{E}[X_j|do(X_i = 0)] \tag{1.2}$$

Adapting to distribution shifts. An interesting implication of the ICM principle is that if we model distribution shifts as interventions on a causal variable, then we only need to update the parameters corresponding to the functional mechanism of the intervened node. Hence, if we trained a model with the correct factorization (1.1) on the observational distribution, then we only need to change the mechanism of the intervened node to adapt the model to the interventional distribution. However, if we had an incorrect factorization during training on the observational distribution, then the effect of intervention is not modular and we need to change several mechanisms to adapt the model on the interventional distribution. The following result by [Bengio et al., 2019] provides theoretical evidence for this claim.

Proposition 1.1.2. *Consider $p, \tilde{p}$ as two distributions sampled from an SCM with causal graph G, and they share causal mechanisms for variables, $p(X_i|PA_i) = \tilde{p}(X_i|PA_i) \; \forall i \in C$. Let θ_i denote the parameters for the factor $p(X_i|PA_i)$, then we have $\mathbb{E}_{x \sim \tilde{p}(x)}[\frac{\partial log p(x)}{\partial \theta_i}] = 0 \; \forall i \in C$.*

In many practical scenarios we do not observe the correct causal factorization and only have measure the causal variables themselves. Therefore, inferring the correct causal factorization from data is an important problem, which we discuss in detail in the next subsection.

1.1.2 Causal Discovery

Consider an SCM with causal graph G that entails a distribution $p(x)$ over the causal variables. Then the problem of recovering the SCM from a dataset containing causal variables $\{x_1, \cdots, x_d\} \; \sim p(x)$ is called causal discovery. A common characterization of the SCM is to consider the conditional independence (CI) constraints (d-separation) [Pearl, 2009] entailed by its causal graph $\mathcal{G}$, and solve causal discovery by inferring which SCMs entail the same CI constraints as those in the data distribution $p(x)$. However, such constraint-based methods [Spirtes et al., 2000, Sun et al., 2007, Zhang et al., 2012] rely on the assumption of faithfulness and in principle recover an equivalence class over graphs (Markov equivalence class) that satisfy the same CI constraints [Peters et al., 2017].

To understand this better, lets first assume that as result of some learning algorithm we accurately learnt the joint distribution $p(x)$ and now we look at the different conditional independence (CI) constraints satisfied by $p(x)$. It can be shown for some distributions given by the true SCM there exist CI constraints in $p(x)$ that are not satisfied by the true causal graph (G). In such cases causal discovery is hopeless as we would rule the correct solution G based on all the CI constraints satisfied by $p(x)$. Hence, we make the assumption that we only work with distributions where every CI constraint satisfied by $p(x)$ is also satisfied by the true graph (G), also known as the **faithfulness** assumption. Further, there is no guarantee that there will be a unique graph that can satisfies all the CI constraints entailed by $p(x)$. E.g., consider two causal graphs $A \to B$ and $B \to A$, then the CI constraints satisfied by these graphs is exactly the same.

Hence, the inverse problem is *not identifiable* since there is an equivalence causal graphs ($\mathcal{G}$) that satisfy the same set of CI constraints, known as the **markov equivalence** class The learning problem can be made identifiable under parametric assumptions on the causal mechanisms ($f_i(PA_i, N_i)$) [Peters et al., 2014, Peters and Bühlmann, 2014] or with access to interventional data [Hauser and Bühlmann, 2012, Bengio et al., 2019]. Another class of methods are score-based causal discovery methods [Tsamardinos et al., 2006, Hoyer et al., 2008, Huang et al., 2018], that search over the discrete space of DAGs and return the DAG with the best score, usually computed via variants of maximum likelihood estimation $p(D|\mathcal{G})$. This combinatorial optimization problem is NP-hard [Chickering et al., 2004] due to the super-exponential search space of DAGs. Hence, there have been several methods based on greedy search [Chickering, 2002] and efficient exploration [Deleu et al., 2022, 2024] of the search space for score-based causal discovery. Recent works have also transformed this combinatorial optimization problem into a differentiable continuous optimization problem [Zheng et al., 2018, Lachapelle et al., 2019, Brouillard et al., 2020, Lippe et al., 2021], which allows for more scalable causal discovery algorithms.

1.1.3 Causal Inference with Observational Data

While we could infer the causal effects with the learned SCM (causal discovery) via do-calculus [Pearl, 2009], for certain scenarios we do not need to learn the complete SCM. Consider two random variables; W as the treatment variable that undergoes a *perfect intervention*, and Y as the outcome of interest. If we can measure all the pre-treatment variables (confounders) X that cause both W and Y, then we can estimate the interventional distribution $p(Y|do(W))$ using only observed data.

$$p(Y|do(W)) = \mathbb{E}_X[p(Y|X,W)] = \int p(Y|X,W)p(X)dx \tag{1.3}$$

This is also known as the backdoor estimator [Pearl, 2009], where the confounders X satisfy the backdoor criterion, i.e., X blocks all the paths between W and Y, and X is a non-descendant of W. Note that backdoor criterion is only a sufficient condition, i.e., there could exist another valid adjustment set Z such that $p(Y|do(W)) = \mathbb{E}_Z[p(Y|Z,W)]$.

Hence, with the backdoor estimator we can estimate the causal effect using observational data as $\mathbb{E}[Y|do(W)] = \mathbb{E}_X\mathbb{E}_Y[Y|W,X]$ (proof in Appendix B.1). Similarly, we can estimate the ATE (E.q. 1.2) using observational data as follows:

$$\text{ATE:}\quad \mathbb{E}_X[\,\mathbb{E}_Y[Y|X,W=1] - \mathbb{E}_Y[Y|X,W=0]\,] \tag{1.4}$$

Similar results for ATE estimation with observational data can also be derived using the potential outcomes framework [Rubin, 2005] under ignorability assumptions.

Proposition 1.1.3. *Let Y^0 and Y^1 denote potential outcomes under treatments $do(W=0)$ and $do(W=1)$. Then ATE ($\mathbb{E}[Y^1] - \mathbb{E}[Y^0]$) is identified from observational data (1.4) under the following assumptions.*

- ***Consistency:*** *The observed outcome we observe reflects the true potential outcome under the observed treatment, $Y = W \cdot Y(1) + (1-W) \cdot Y(0)$.*
- ***Exchangeability:*** *We do not have unobserved confounders, $Y(0), Y(1) \perp\!\!\!\perp W \mid X$. Unobserved confounders would lead to open backdoor paths between Y(0), Y(1) and W, hence we would not have conditional independence.*
- ***Overlap:*** *The treatment assignment for each sample is probabilistic, $0 < \pi(x) < 1 \;\; \forall x \in X$. Therefore, each sample has a non-trivial chance of being assigned either to the control group ($W=0$) or the treatment group ($W=1$).*

1.2 Disentangled Representation Learning

In the previous section, we discussed the cause-effect framework in detail and its implication for designing modular systems that adapt efficiently to distribution shifts. However, we made the assumption that the causal variables of interest are observed in the dataset, which is likely to be violated for high-dimensional data like like images and text. In this section, we will discuss the task of disentangling latent causal factors from observations, and provide a thorough overview of the related works.

1.2.1 Problem Setup

We assume the causal variables/factors of variation ($Z \in \mathbb{R}^{d_z}$) are latent and the observations ($X \in \mathbb{R}^{d_x}$) are a function of the latent causal variables.

$$x = g(z) \;\; \forall z \in \mathcal{Z} \;\; \text{where} \;\; z \sim \mathbb{P}_Z \tag{1.5}$$

The task of recovering/disentangling the latent causal variables responsible for generating the data is referred to as causal/disentangled/identifiable representation learning. A common learning objective considered for the same is the reconstruction objective with Auto-Encoders, where the optimal encoder $\hat{f} : \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_z}$ and decoder $\hat{g} : \mathbb{R}^{d_z} \rightarrow \mathbb{R}^{d_x}$ would satisfy the following:

$$\mathbb{E}_{x \sim X} ||x - \hat{g}(\hat{f}(x))||^2 = 0 \tag{1.6}$$

Denoting the learned latent variables as $\hat{z} = \hat{f}(x)$, the optimal solution of the reconstruction objective would imply the following under the assumption that g *is injective* (detailed steps in Appendix B.3).

$$\hat{z} = v(z) \;\; \text{where} \;\; v(z) = \hat{g}^{-1} \circ g(z) \;\; \forall z \in \mathcal{Z} \tag{1.7}$$

Hence, without further constraints, there is an indeterminacy in recovering the latent variables given by the function $v(z) = \hat{g}^{-1} \circ g(z)$. Further, given an optimal solution of the reconstruction objective $(\hat{f}, \hat{g})$, we can create another optimal solution $(a \circ \hat{f}, \hat{g} \circ a^{-1})$ using an invertible function a such that the indeterminacy in latent recovery changes to $a \circ \hat{g}^{-1} \circ g(z)$; making the problem not identifiable. The problem of identifiability is that, given only the distribution over X, it is impossible to distinguish between the two models $(\hat{f}, \hat{g})$ and $(a \circ \hat{f}, \hat{g} \circ a^{-1})$. This is problematic when one wants to discover interpretable factors of variations since $\hat{z}$ and $a \circ \hat{z}$ could be drastically different.

Therefore, we need to constrain the problem to restrict the indeterminacy $v(z)$ to simple functions. E.g., a desirable solution would be to restrict this to linear transformations.

Definition 1.2.1. [Linear Identification] If the learned latent variables $\hat{z}$ are an affine function of the latent variables (z), $\hat{z} = Az + b \ \ \forall z \in \mathcal{Z}$ where $A \in \mathbb{R}^{d_z} \to \mathbb{R}^{d_z}$ is invertible, we achieve linear identification.

Further, an ideal solution from practical perspective would be to restrict the linear transformations to permutation and component-wise scaling, so that each learned latent variable $(\hat{Z})$ corresponds to the latent variables (Z) in a one-to-one manner, i.e, we have *disentangled* latent variables.

Definition 1.2.2. [Permutation & Scaling Identification] If the learned latent variables $\hat{z}$ are related to the latent variables z by a permutation matrix $(\Pi \in \mathbb{R}^{d_z} \to \mathbb{R}^{d_z})$ and an invertible diagonal scaling matrix $(\Lambda \in \mathbb{R}^{d_z} \to \mathbb{R}^{d_z})$, $\hat{z} = \Pi \circ \Lambda(z) + b \ \ \forall z \in \mathcal{Z}$, we achieve permutation & scaling identification.

We can also relax the definition above for a more general definition of disentanglement where we allow for component-wise non-linear functions as opposed to component-wise scaling.

Definition 1.2.3. [Disentanglement] If the learned latent variables $\hat{z}$ are related to the latent variables z by a diffeomorphism v, $\hat{z} = v(z) \ \ \forall z \in \mathcal{Z}$, such that the jacobian of v is a permuted diagonal matrix, then we achieve disentanglement, as we have a one-to-one mapping between components of z and $\hat{z}$.

Remark. In the above definition for disentanglement, the one-to-one mapping between the components of z and $\hat{z}$ may change as we sample different $z \in \mathcal{Z}$. This is typically referred to as *local disentanglement* as the disentanglement pattern might change across samples. If the disentanglement pattern remains the same across samples, we refer to it as *global disentanglement*. Please refer to Appendix B.2 for more details regarding them.

To achieve the identification guarantees mentioned above, broadly we have two options, we can put constraints on the data generation process $(\mathbb{P}(Z), g)$ and the learner $(\hat{f}, \hat{g})$.

1. **Constraints on mixing function.** [Taleb and Jutten, 1999, Gresele et al., 2021, Leeb et al., 2021, Moran et al., 2022, Buchholz et al., 2022, Zheng et al., 2022] Restrict the function class of the true decoder (g) and the learned decoder $(\hat{g})$, as this directly controls the indeterminacy in latent recovery, $v(z) = \hat{g}^{-1} \circ g(z)$.

2. **Constraints on latent distribution.** [Hyvärinen et al., 2019, Locatello et al., 2020b, Lachapelle et al., 2022b, Lippe et al., 2022b, Wang and Jordan, 2022, Roth et al., 2023] Make assumptions on the distribution of latent variables $\mathbb{P}(Z)$ and enforce the learned latent variables $(\hat{z} = v(z))$ to follow these constraints as well. This restricts the indeterminacy as the transformation v has to preserve the distributional constraints.

Note: While we used Auto-Encoders and reconstruction objective in the discussion above, there are alternative frameworks that can be used for disentangling latent variables as well, like Variational Auto-Encoders (VAEs) [Khemakhem et al., 2020a], contrastive learning [Hyvarinen et al., 2019], etc. Auto-Encoders provide an easy way to characterize the indeterminacy in latent recovery, hence being a popular choice for identifiable representation learning. For decoder free learning methods, a useful way to characterize the indeterminacy in latent recovery in such cases is as follows:

$$\hat{z} = \hat{f}(x) = \hat{f} \circ g(z) = v(z) \;\; \forall z \in \mathcal{Z} \tag{1.8}$$

A general recipe would be to restrict $v(z) = \hat{f} \circ g(z)$ by putting constraints on $(\hat{f}, g)$ and other constraints that would follow as per the optimal solution of the learning objective.

Note: Another common assumption in latent variable identification proofs is that *we assume access to infinite data samples when optimizing a learning objective*, so we don't have to worry about finite sample issues with optimization. This is implicitly assumed in all the identification results discussed in this report.

1.2.2 Independent Component Analysis (ICA)

If we restrict the latent distribution to have mutually independent components, then the task of identifiable representation learning is also termed as ICA. We first analyse the case of **Linear ICA**, where the mixing function g is linear and the latent variables (Z) have mutually independent components and *no component is gaussian.* If the learner $\{\hat{f}, \hat{g}\}$ also mimics similar constraints of mutual independence on learned latents $(\hat{z})$ and (invertible) linear decoder $(\hat{g})$; then we achieve permutation & scaling identification (Def. 1.2.2) [Hyvärinen and Oja, 2000].

Note that by restricting the $(g, \hat{g})$ to be linear, we already achieve linear identification as the indeterminacy in latent recovery under optimal reconstruction objective (E.q. 1.7) is linear, $v(z) = \hat{g}^{-1} \circ g(z) = Az$. Then we can utilize the constraint on the latents $(Z, \hat{Z})$ to restrict the A to only permutation & scaling.

Proposition 1.2.4. *Let $Z, \hat{Z} \in \mathbb{R}^{d_z}$ be random variables s.t. $\hat{z} = Az \;\; \forall z \in \mathcal{Z}$. If Z and $\hat{Z}$ have mutually independent components and no component of Z is gaussian, then A is permutation & scaling matrix.*

The proof for the same is provided in Appendix B.4 and it involves the application of Darmois-Skitovitch Theorem [Theis, 2004], which also provides a formal argument for the necessity of non-gaussian components in Z. Intuitively, the issue with gaussian components is that the gaussian distribution with mutually independent components is invariant to rotations, hence it would imply that the indeterminacy ($\hat{z} = Az$) can include rotation matrices.

This also leads to an algorithm for linear ICA; learn $(\hat{f}, \hat{g})$ by solving reconstruction objective (E.q. 1.6) along with minimizing the mutual information between components of $\hat{z} = \hat{f}(X)$, to encourage mutual independence. Since estimation of mutual information is a challenging task, FastICA [Hyvärinen and Oja, 2000] has been proposed that maximizes the non-gaussianity of each component of $\hat{z}$.

Non-linear ICA. Non-linear ICA refers to the case when mixing function g is a general non-linear function and the latent Z have mutually independent components. The complexity of the mixing function g determines the identifiability to a large extent, as evident by the indeterminacy in latent recovery being a function of it. While for the case of linear ICA we were able to achieve identification upto permutation & scaling, Locatello et al. [2019] show that the same cannot be guaranteed for non-linear ICA.

Theorem 1.2.5. *Locatello et al. [2019] Let $Z \in \mathbb{R}^{d_z}$ be a random vector with mutually independent components. Then we can construct an invertible function $v : \mathcal{Z} \rightarrow \mathcal{Z}$ such that $v(z)$ entangles the components of z, $\dfrac{\partial v_i(z)}{\partial z_j} \neq 0 \;\; \forall i, j$, and $v(Z)$ has the same marginal distribution as Z.*

Hence, the above theorem states there exists an entangled solution, $\hat{z} = v(z)$, which satisfies the distributional constraint of mutual independence and still achieves perfect reconstruction with the (encoder, decoder) pair as $\big(\hat{f}(x) = v(z) \;,\; \hat{g}(z) = g \circ v^{-1}(z)\big)$. Note that this does not imply that we can never disentangle the true latents for non-linear ICA, since we may utilize constraints on the mixing function to restrict the function v as we should have $v(z) = \hat{g}^{-1} \circ g(z)$ for optimal solutions to the reconstruction objective. The theorem states that in the *absence of any further assumptions on the mixing function (g)*, the indeterminacy in latent recovery cannot be reduced to simple transformations.

1.2.3 Non-Linear ICA with Auxiliary Information

Given the challenges with unsupervised learning of disentangled representations in Non-Linear ICA, we discuss alternative solutions in the literature that utilize auxiliary information to have more information about the latent variables. The earliest works in this direction used time index as auxiliary information for solving Non-Linear ICA [Hyvarinen and Morioka, 2016], where the authors showed that if each component of the latent vector evolves independently and follows a non-stationary time series without temporal dependence, then latent identification is possible. In contrast, [Hyvarinen and Morioka, 2017] showed that if the latent variables are mutually independent, with each component evolving in time following a stationary time series with temporal dependence, then also identification is possible. Hyvarinen et al. [2019] further generalized the previous results, where instead of using time, the authors require observation of general auxiliary information. We discuss their work in more detail given its importance in presenting a unified framework for utilizing auxiliary information.

Non-Linear ICA using auxiliary variables and contrastive learning. In this work [Hyvarinen et al., 2019], the authors assume that alongside the data ($X \in R^{d_x}$) we also observe auxiliary information ($U \in \mathbb{R}^{d_u}$) which renders the latent variables $Z \in R^{d_z}$ to be conditionally independent, i.e, $Z_i \perp Z_j \mid U \;\; \forall i, j$. The data generation process is described as follows:

$$u \sim p(U) \;\;,\;\; z \sim p(Z|U) \;\;,\;\; x = g(z) \quad \text{where} \quad p(Z|U) = \prod_{i=1}^{d_z} q_i(Z_i, U) \tag{1.9}$$

Note that $q_i(Z_i, U)$ is a function that characterizes the distribution $p(Z_i|U)$ and the above factorization of $p(Z|U)$ is a result of the conditional independence assumption.

For the learning objective, given a dataset $\{x, u\} \sim P(X, U)$ the authors propose to use a variant of contrastive learning where the task is to classify between correctly matched pairs (x, u) and randomly matched pairs (x, u^*) where u^* denotes a random sample from $p(U)$. To solve this binary classification task, they use logistic regression as shown below with the logit function $\psi(x, u)$ and σ representing the sigmoid function.

$$\hat{f}, \hat{h} \leftarrow \min_{f,h} \;\; \mathbb{E}_{x,u}\big[-log(\sigma(\psi(x,u))) + log(\sigma(\psi(x,u^*))\big] \quad \text{where} \quad \psi(x,u) = \sum_{i=1}^{d_z} h_i(f_i(x), u) \tag{1.10}$$

The logit is modeled by the learner as $\psi(x, u) = \sum_{i=1}^{d_z} h_i(f_i(x), u)$ where the encoder $f : \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_z}$ is learning the representation of the data (x) and each h_i is a function that maps $(f_i(x), u)$ to a scalar value. The inductive bias of addition of terms ($h_i(f_i(x), u)$) that depend on a particular component $f_i(x)$ encourages conditional independence. We now state their identification results for the learned latents ($\hat{z} = \hat{f}(x)$) as per the optimal solution to the contrastive learning objective above.

Theorem 1.2.6. *Given the data generation process (E.q. 1.9) and the optimal solution $(\hat{f}, \hat{h})$ under the learning objective (E.q. 1.10), along with the extra assumptions stated below:*

1. *The learned encoder $\hat{f} = (\hat{f}_1, \cdots, \hat{f}_{d_z})$ and the mixing function g are C^2-diffemorphisms.*

2. *The functions $q_i(z_i, u)$ in the log density function $p(Z|U)$ are C^2-functions.*

3. ***Assumption of Sufficient Variability.*** *Denote the diagonal terms of the jacobian and hessian of $q(z, u)$ w.r.t z as $w(z, u) = \big(\frac{\partial q_1(z_1,u)}{\partial z_1}, \cdots, \frac{\partial q_{d_z}(z_{d_z},u)}{\partial z_{d_z}}, \frac{\partial^2 q_1(z_1,u)}{\partial^2 z_1}, \cdots, \frac{\partial^2 q_{d_z}(z_{d_z},u)}{\partial^2 z_{d_z}}\big)$. Then $\forall z \in \mathcal{Z}\ \exists\ 2 * d_z + 1$ values for u such that the following set of vectors are independent: $\Big\{w(z, u_i) - w(z, u_0)\ \mid\ i \in [1, 2 * d_z]\Big\}$.*

Then we achieve disentanglement (Def. 1.2.3) with the learned latent variables $\hat{z}$ where $\hat{z} = \hat{f}(x)$.

To get more intuition behind the identification result, we provide a proof sketch in Appendix B.5. While the paper generalizes the Non-Linear ICA problem beyond mutually independent latent variables, it still does not allow for general SCM in the latent space as the latent variables have to satisfy conditional independence given the auxiliary information. This framework has been extended for VAE [Khemakhem et al., 2020a], energy based models [Khemakhem et al., 2020d], etc., but with similar assumptions of conditional independence in the latent space. Further, this approach has been extended for datasets with sparse temporal dependencies [Hälvä and Hyvarinen, 2020, Yao et al., 2022b,a, Lippe et al., 2022b,a, Lachapelle et al., 2022b, Lachapelle and Lacoste-Julien, 2022].

Weakly-supervised disentanglement without compromises. Another major source of auxiliary information exploited in prior works is sparse data augmentations. This is also referred to as weak supervision in the literature since we do not consider labelled supervised data as auxiliary information. Locatello et al. [2020a] introduced this idea of using weak supervision in the form of data pairs $X \in \mathbb{R}^{d_x}, \tilde{X} \in \mathbb{R}^{d_x}$ under the assumption that the latent factors across these pairs ($Z \in R^{d_z}, \tilde{Z} \in R^{d_z}$) have some shared components, i.e., the change from Z to $\tilde{Z}$ is sparse and not all of the components change. Specifically, we first sample a *non-trivial* set of coordinates S that specify the shared components across pairs, $Z_i = \tilde{Z}_i\ \ \forall i \in S$. Further, they assume both $Z, \tilde{Z}$ are continuous random variables with mutually independent components. The data generation process is described as follows:

$$S \sim p(S) \quad , \quad z \sim p(Z) = \prod_{i=1}^{d_z} p(Z_i) \quad , \quad y \sim p(Y) = \prod_{i=1}^{k} p(Y_i) \quad \text{where } \ |S| = d_z - k > 0$$
$$x = g(z) \ \ , \ \ \tilde{x} = g(\tilde{z}) \quad \text{where } \ \tilde{z} = h(z, y, S) \quad \text{s.t.} \quad \tilde{z}_S = z_S \ \ , \ \ \tilde{z}_{\bar{S}} = y \ \ , \ \ \tilde{z}_{\bar{S}} \neq z_{\bar{S}} \tag{1.11}$$

Note that $y \in \mathbb{R}^k$ denote the latent components in $\tilde{z}$ that are independent of the latent variables z, hence $\tilde{z}$ has same components as z for coordinates in set S and it has same components as y for coordinates in $\bar{S} = [d]/S$. Also, z and $\tilde{z}$ do not have same components for the coordinates in $\bar{S}$.

Consider the Auto-Encoder framework such that the optimal encoder $(\hat{f})$ and decoder $(\hat{g})$ achieve zero reconstruction error for both $X \& \tilde{X}$, while also preserving the constraint that learned latent variables $(\hat{z}, \hat{\tilde{z}})$ have mutually independent components and they share components for coordinates in the set S.

$$\begin{aligned} &\mathbb{E}_{x\sim X}||x - \hat{g}(\hat{f}(x))||^2 = 0 \;\;, \quad \mathbb{E}_{\tilde{x}\sim \tilde{X}}||\tilde{x} - \hat{g}(\hat{f}(\tilde{x}))||^2 = 0 \\ &\hat{z}_i \perp \hat{z}_j \; \forall i,j \;\;, \quad \hat{\tilde{z}}_i \perp \hat{\tilde{z}}_j \; \forall i,j \quad \text{where} \;\; \hat{z} = \hat{f}(x) \;\;, \;\; \hat{\tilde{z}} = \hat{f}(\tilde{x}) \\ &\hat{z}_i = \hat{\tilde{z}}_i \; \forall i \in S \;\;, \quad \hat{z}_j \neq \hat{\tilde{z}}_j \; \forall j \in \bar{S} \end{aligned} \tag{1.12}$$

We now state their identification result for a fixed and known set S. For more details on the proof sketch and extensions for the case of unknown S varying across samples, please check Appendix B.6.

Theorem 1.2.7. *Given the data generation process (E.q. 1.11) and the optimal solution $(\hat{f}, \hat{g})$ under the learning objective (E.q. 1.12), along with the following extra assumptions:*

1. *The mixing function/true decoder (g) and the learned decoder $(\hat{g})$ are diffeomorphisms*

2. *The learner knows the coordinate set S which is fixed across all the data pairs.*

Then we achieve block disentanglement, i.e., $\hat{z} = v(z)$ such that $\hat{z}_S$ depends only on the latent component in S (z_S) and $\hat{z}_{\bar{S}}$ depends only on the latent components in $\bar{S}$ $(z_{\bar{S}})$.

The major limitation with their identification result is the assumption of mutual independence in the latent variables Z. This implies that we can only disentangle causal variables for latent SCMs with the trivial causal graph (no connections between variables). This restrictive assumption in the latent space has been relaxed in follow up works [Von Kügelgen et al., 2021, Brehmer et al., 2022], where they model data augmentations as interventions in the latent space. Ahuja et al. [2022] model data augmentations as perturbations in the latent space ($\tilde{Z} = Z + \delta$ where δ is a sparse vector), and prove identification results without strong assumptions on the latent space.

From counterfactuals to interventions. A key limitation of this perspective of data augmentations as weak supervision is that we need paired data samples $(x, \tilde{x})$, essentially counterfactuals, as the non-intervened latents corresponding to x and $\tilde{x}$ are exactly the same. For certain practical scenarios like genetic perturbations Dixit et al. [2016], we might have access to intervention distribution as result of these perturbations, however, we don't necessarily have access to pre-intervention and post-intervention paired samples. Hence, utilizing intervention distributions as opposed to counterfactual samples as a source of weak supervision is a focus of several recent works [Ahuja et al., 2023, Squires et al., 2023, Buchholz et al., 2024, von Kügelgen et al., 2024, Jiang and Aragam, 2023, Zhang et al., 2024]. Ahuja et al. [2023] were amongst the first works to propose latent identification results that go beyond the assumption of paired counterfactual data and only operate with interventional distributions. They assume that along with observed data $x = g(z)$ where $z \sim \mathcal{Z}$, we also observe data $x^{(i)} = g(z^{(i)})$ where $z^{(i)} \sim \mathcal{Z}^{(i)}$ represents the intervention on the latent component z_i. Therefore, across the environments $\{x\}$ and $\{x^{(i)}\}$ we might not have exact pairs such that the non-intervened variables are the same, however, we know that these data samples differ in terms of intervention on a particular latent target. Under further assumptions on the mixing function $g : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$ and observing perfect interventions on all latent components, they proved that it is possible to achieve permutation & scaling identification Definition 1.2.2. This was extended for soft interventions by Zhang et al. [2024] and for general diffeomorphisms as mixing functions by Buchholz et al. [2024] and von Kügelgen et al. [2024].

1.2.4 Metrics for Disentanglement

Mean Correlation Coefficient (MCC): MCC is a standard metric used in prior works [Hyvarinen and Morioka, 2017, Hyvarinen et al., 2019, Khemakhem et al., 2020a, Zimmermann et al., 2021] to measure permutation and scaling based identification (Definition 1.2.2). The metric is computed by first obtaining the correlation matrix $(\rho(Z, \hat{Z}))$ between the recovered latents $\hat{Z} \in \mathbb{R}^{d_z}$ and the true latents $Z \in \mathbb{R}^{d_z}$. Let's define $|\rho(Z, \hat{Z})|$ as the absolute values of the correlation matrix. Then we find a matching (assign each row to a column in $|\rho(Z, \hat{Z})|$) such that the average absolute correlation is maximized and return the optimal average absolute correlation. Intuitively, we find the optimal way to match the components of the predicted latent representation $(\hat{Z})$ and components of the true representation (Z). Notice that a perfect absolute correlation of one for each matched pair of components would imply identification up to permutations.

$$\text{MCC} := \max_{\pi} \sum_{i=1}^{d_z} |\rho(Z, \pi\hat{Z})|_{i,i} \quad \text{where} \;\; \pi \in \mathbb{R}^{d_z \times d_z} \;\; \text{is a permutation matrix} \tag{1.13}$$

The correlation matrix ρ is typically computed via pearson/spearman correlation, and the optimization problem with searching over the space of permutation matrices is typically solved as a linear sum assignment problem [Crouse, 2016].

Disentanglement, Completeness, and Informativeness (DCI): DCI was introduced by Eastwood and Williams [2018] and aims to infer the relevance of each dimension of learned latent $\hat{Z} \in \mathbb{R}^{d_z}$ in predicting the true latent $Z \in \mathbb{R}^{d_z}$. We learn d_z different regression models $\{f_i\}_{k=1}^{d_z}$ where the regression model $f_i : \mathbb{R}^{d_z} \rightarrow \mathbb{R}$ takes as input the learned latent $\hat{Z}$ and predicts Z_i. Lets denote the coefficient vector learned by the regression model f_i as $w_i \in \mathbb{R}^{d_z}$, and construct a matrix $W = [w_1, \cdots, w_{d_z}] \in \mathbb{R}^{d_z \times d_z}$. Intuitively, W_{ij} denotes the relevance of $\hat{Z}_i$ in predicting Z_j. Using this matrix W, DCI would compute a disentanglement score for each inferred latent $\hat{Z}_i$ and a completeness score for each true latent Z_i.

To get the disentanglement score for $\hat{Z}_i$, we determine whether its informative for predicting only a single true latent component. We first compute the probability of $\hat{Z}_i$ being relevant for predicting Z_j as $p_{ij} = W_{ij} / \sum_j W_{ij}$. Then we measure the entropy of the multinomial distribution with probabilities $p_i = [p_{i1}, \cdots, p_{id_z}]$, as a lower entropy would imply a sparser p_i, essentially more disentangled. Hence, we have $D_i = 1 - \sum_{j=1}^{d_z} p_{ij} \log p_{ij}$, which is aggregated across different components of $\hat{Z}$ to get the final disentanglement score.

$$\text{DCI-Disentanglement} := \sum_{i=1}^{d_z} \rho_i D_i \;\; \text{where} \;\; \rho_i = \sum_j W_{ij} / \sum_{i,j} W_{ij} \tag{1.14}$$

If we only rely on DCI-Disentanglement score then we won't penalize the scenario where some true latent component are not predicted by any component of the predicted latent. Hence, we also compute a completeness score, which is an analogue to the disentangled score and it aims to capture whether each true latent component Z_j is captured by only a single predicted latent component. Hence, we have $C_j = 1 - \sum_{i=1}^{d_z} p_{ij} \log p_{ij}$ where the distribution $p_j = [p_{1j}, \cdots, p_{d_z j}]$ and $p_{ij} = W_{ij} / \sum_i W_{ij}$, that involves aggregation on the predicted latent dimension now.

$$\text{DCI-Completeness} := \sum_{i=1}^{d_z} \rho_j C_j \;\; \text{where} \;\; \rho_j = \sum_i W_{ij} / \sum_{i,j} W_{ij} \tag{1.15}$$

Some papers also combine both the DCI-Disentanglement and DCI-Completeness score into a single score by taking the harmonic mean between them.

Chapter 2

Towards efficient representation identification in supervised learning

This chapter is based on the contents of the paper "Towards efficient representation identification in supervised learning" by Kartik Ahuja*, **Divyat Mahajan***, Vasilis Syrgkanis, and Ioannis Mitliagkas. The paper was accepted at the Conference on Causal Learning and Reasoning [**CLeaR 2022**].

Kartik Ahuja proposed the original project idea and developed the theory for latent identification, with input from Ioannis Mitliagkas. Divyat Mahajan led the experimental design under the guidance of Kartik Ahuja and obtained all the empirical results presented in the paper. Additionally, Divyat Mahajan was actively involved in discussions regarding the theoretical results. Kartik Ahuja led the overall drafting of the paper, while Divyat Mahajan led the writing of the rebuttals and the camera-ready version. Vasilis Syrgkanis and Ioannis Mitliagkas provided supervision throughout the project.

2.1 Introduction

Like we discussed before in Section 1.2, general process of disentanglement is impossible in the absence of side knowledge of the structure of the data generation process [Hyvärinen and Pajunen, 1999, Locatello et al., 2019]. However, under additional structural assumptions on the data generation process, it is possible to invert the data generation process and recover the underlying factors of variation [Hyvarinen and Morioka, 2016]. Recently, there have been works [Hyvarinen et al., 2019, Khemakhem et al., 2020a] which present a general framework that relies on auxiliary information (e.g., labels, timestamps) to disentangle the latents. While they have made remarkable progress in the field of disentanglement, these works make certain key assumptions highlighted below that we significantly depart from.

• **Labels cause the latent variables.** In supervised learning datasets, there are two ways to think about the data generation process—a) labels cause the latent variables and b) latent variables cause the labels. Schölkopf et al. [2012] argue for the former view, i.e., labels generate the latents, while [Arjovsky et al., 2019] argue for the latter view, i.e., latents generate the label (see Figure 2.1). Current non-linear ICA literature [Hyvarinen et al., 2019, Khemakhem et al., 2020a] assumes the label knowledge renders latent factors of variation conditionally independent, hence it is compatible with the former perspective [Schölkopf et al., 2012]. But the latter view might be more natural for the setting where a human assigns labels based on the underlying latent factors. Our goal is to enable disentanglement for this case when the latent variables cause the labels [Arjovsky et al., 2019].

• **Amount of auxiliary information.** Existing works [Hyvarinen et al., 2019, Khemakhem et al., 2020a] require a lot of auxiliary information, e.g., the number of label classes should be twice the total dimension of the latent factors of variation to guarantee disentanglement. We seek to enable disentanglement with lesser auxiliary information.

Contributions. We consider the following data generation process – latent factors generate the observations (raw features) and the labels for multiple tasks, where the latent factors are mutually independent. We study a natural extension of the standard empirical risk minimization (ERM) (Vapnik [1992]) paradigm. The most natural heuristic for learning representations is to train a neural network using ERM and use the output from the representation layer before the final layer. In this work, we propose to add a constraint on ERM to facilitate disentanglement – all the components of the representation layer must be mutually independent. Our main findings for the representations learned by the constrained ERM are summarized below.

• If the number of tasks is at least equal to the dimension of the latent variables, and the latent variables are not Gaussian, then we can recover the latent variables up to permutation and scaling.

• If we only have a single task and the latent variables come from an exponential family whose log-density can be expressed as a polynomial, then under further constraints on both the learner's inductive bias and the function being inverted, we can recover the latent variables up to permutation and scaling.

• To implement constrained ERM, we propose a simple two-step approximation. In the first step, we train a standard ERM based model, and in the subsequent step we carry out linear ICA [Comon, 1994] on the representation extracted from ERM. We carry out experiments with the above procedure for regression and classification. Our experiments

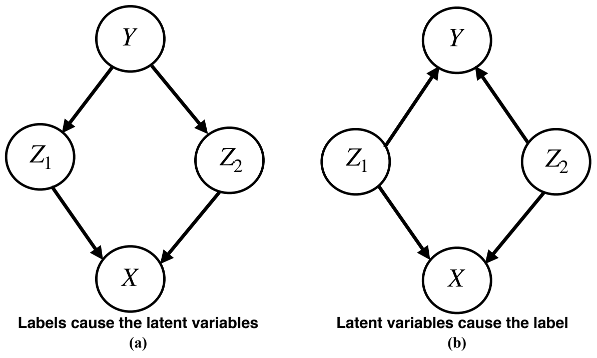

Y
Z1
Z2
X
Labels cause the latent variables
(a)
Y
Z1
Z2
X
Latent variables cause the label
(b)

Figure 2.1: (a) DGP in Hyvarinen et al. [2019]; (b) DGP studied in this work.

show that even with the approximate procedure, it is possible to recover the true latent variables up to permutation and scaling when the number of tasks is smaller than the latent dimension.

2.2 Problem Setup

Assumption 2.2.1 (Data Generation Process)**.** The data generation process for regression is described as

$$Z \leftarrow h(N_Z) \ \ , \ \ X \leftarrow g(Z) \ \ , \ \ Y \leftarrow \Gamma Z + N_Y \tag{2.1}$$

where we have the following:

- $N_Z \in \mathbb{R}^d$ is noise, $h : \mathbb{R}^d \rightarrow \mathbb{R}^d$ generates the latent variables $Z \in \mathbb{R}^d$. The components of Z are mutually independent, non-Gaussian, and have finite second moments.
- $g : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is a bijection that generates the observations X from latent variables Z
- $\Gamma \in \mathbb{R}^{k\times d}$ is a matrix that generates the label $Y \in \mathbb{R}^k$, and $N_Y \in \mathbb{R}^k$ is the noise vector (N_Y is independent of Z and $\mathbb{E}[N_Y] = 0$)

Note that d is the dimension of the latent representation, and k corresponds to the number of tasks. We adapt the data generation process to multi-task classification as follows by changing the label generation process.

$$Y \leftarrow \mathsf{Bernoulli}\Big(\sigma\Big(\Gamma Z\Big)\Big) \tag{2.2}$$

where (σ) corresponds to the sigmoid function applied elementwise to ΓZ and outputs the probabilities of the tasks, and $\mathsf{Bernoulli}$ operates on these probabilities elementwise to generate the label vector $Y \in \{0, 1\}^k$ for the k tasks.

Our objective is to learn the model g^{-1} from the observed data and labels pairs (X, Y), such that for new observations X we can recover the latent variables Z that generated X. Let us denote the model learned as $\hat{g}^{-1}$ and the recovered latents as $\hat{z} = \hat{g}^{-1}(x)$, then our goal is to obtain permutation & scaling identification (Definition 1.2.2).

Remark. Prior works on using auxiliary information (labels) for disentanglement [Hyvarinen et al., 2019, Khemakhem et al., 2020a] consider the data generation process where the labels cause the latent variables (E.q. 1.9). This may be valid for some settings, but it is not perfectly suited for human labelled datasets where a label is assigned based on the underlying latent factors of variations. Hence, we focus on the opposite perspective [Arjovsky et al., 2019] when the latent variables generate the labels. Our setting could be interpreted as multi-task supervised learning, where the downstream task labels serve as the auxiliary information generated from shared latent variables. We also contrast the DAGs of the two data generation processes in Figure 2.1.

2.3 Identifiability Analysis of IC-ERM

The previous section established our objective of learning the model g^{-1} (or the decoder g) from the observed data (X, Y). We first train a supervised learning model to predict the labels Y from X. For the rest of the work, we will assume that the predictor we learn takes the form $\Theta \circ \Phi$, where $\Theta \in \mathbb{R}^{k \times d}$ is a linear predictor that operates on the representation $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^d$. As a result, the hypothesis space of the functions that the learner searches over has two parts: $\Theta \in \mathcal{H}_\Theta$ corresponding to the hypothesis class of linear maps, and $\Phi \in \mathcal{H}_\Phi$, where $\mathcal{H}_\Phi$ corresponds to the hypothesis class over the representations. We measure the performance of the predictor on an instance (X, Y) using the loss $\ell\big(Y, \Theta \circ \Phi(X)\big)$ (mean square error for regression, cross-entropy loss for classification). We define the risk achieved by a predictor $\Theta \circ \Phi$ as $R\big(\Theta \circ \Phi\big) = \mathbb{E}\Big[\ell\big(Y, \Theta \circ \Phi(X)\big)\Big]$, where the expectation is taken with respect to the data (X, Y).

Independence Constrained ERM (IC-ERM): The representations (Θ) learnt by ERM as described above have no guarantee of recovering the true latent variables up to permutations. Hence, we propose a new objective where we want the learner to carry out constrained empirical risk minimization, where the constraint is placed on the representation layer that all components of the representation are mutually independent. We provide the formal definition of mutual independence for the convenience of the reader below.

Definition 2.3.1. Mutual independence. A random vector $V = [V_1, \cdots, V_d]$ is said to be mutually independent if for each subset $\mathcal{M} \subseteq \{1, \cdots, d\}$ we have $p(\{V_i\}_{i \in \mathcal{M}}) = \prod_{i \in \mathcal{M}} p(V_i)$.

We state the proposed independence constrained ERM (IC-ERM) objective formally as follows:

$$\min_{\Theta \in \mathcal{H}_\Theta, \Phi \in \mathcal{H}_\Phi} R(\Theta \circ \Phi) \;\text{ s.t. } \Phi(X) \text{ is mutually independent (Definition 2.3.1)} \tag{2.3}$$

Before we state our main latent identification result, we state some extra assumptions on the hypothesis class of the representations ($\mathcal{H}_\Phi$) and the classifier ($\mathcal{H}_\Theta$).

Assumption 2.3.2. Assumption on $\mathcal{H}_\Phi$ and $\mathcal{H}_\Theta$. For the true solutions (g^{-1}, Γ), we have $g^{-1} \in \mathcal{H}_\Phi$ and $\Gamma \in \mathcal{H}_\Theta$. For the case when $k = d$, the set $\mathcal{H}_\Theta$ corresponds to the set of all invertible matrices.

The above assumption ensures that the true solutions g^{-1} and Γ are in the respective hypothesis classes that the learner searches over. Also, the invertibility assumption on the hypothesis in $\mathcal{H}_\Theta$ is needed to ensure that we do not have redundant tasks for the identification of the latent variables.

We now state our main result that show the IC-ERM learning objective would recover the true latent variables up to permutations & scaling.

Theorem 2.3.3. *Given the data generation process (Assumption 2.2.1) and the optimal solution $\Theta^\dagger \circ \Phi^\dagger$ to IC-ERM (2.3) with ℓ as square loss for regression and cross-entropy loss for classification, along with the exta assumptions stated below:*

- *Assumption 2.3.2 for the true solution g^{-1} and Γ.*
- *The number of tasks k is equal to the dimension of the latent d,*

Then we achieve permutation & scaling identifiability (Definition 1.2.2) with the learned latent variables $\hat{z} = \Phi^\dagger(x)$.

The proof for the same is available in Appendix Section C.2. This implies that for the DAGs in Figure 2.1 (b), it is possible to recover the true latents up to permutation and scaling. This result extends the current disentanglement guarantees [Khemakhem et al., 2020d] that exist for models where labels cause the latent variables (latent variables are conditionally independent) to the settings where latent variables cause the label (latent variables are not conditionally independent). In multi-task learning literature [Caruana, 1997, Zhang and Yang, 2017], it has been argued that learning across multiple tasks with shared layers leads to internal representations that transfer better. The above result states the conditions when the ideal data generating representation shared across tasks can be recovered.

ERM obtains Linear Identification. An important insight from the proof of the above

Theorem is that ERM achieves linear identification (Definition 1.2.1), which we highlight in Appendix C.1. The interesting part about this result is that it does not require us to assume that the latent variables are mutually independent and non-gaussian. Therefore, one can view our result in a modular fashion, ERM reduces the disentanglement problem with non-linear mixing function to disentanglement with a linear mixing function. Then we can use existing techniques to tackle the linear mixing disentanglement problem, which in our current work is Linear ICA (hence the assumption of mutual independence and non-gaussianity).

Identification with fewer tasks than latent dimension. It is intuitive that more auxiliary information/numbers of tasks (k) should help us to identify the latent variables as they are shared across these different tasks. The theorem above states that identification is possible when the number of tasks k is equal to the dimension of the latent variables d. We also analyze the case when the number of tasks are less than the latent dimension, and even handle the extreme case of a single. Please refer to Appendix C.4 for more details.

2.4 Methodology: Proposed Implementation for IC-ERM

In the previous section, we showed the identification guarantees with the IC-ERM objective. However, solving this objective is non-trivial, since we need to enforce independence on the representations learnt. We propose a simple two step procedure as an approximate approach to solve the above problem. The learner first carries out standard ERM stated as

$$\Theta^{\dagger}, \Phi^{\dagger} \in \underset{\Theta \in \mathcal{H}_{\Theta}, \Phi \in \mathcal{H}_{\Phi}}{\arg\min} \; R(\Theta \circ \Phi) \tag{2.4}$$

The learner then searches for a linear transformation Ω that when applied to $\Phi^{\dagger}$ yields a new representation with mutually independent components. We state this as follows. Find an invertible $\Omega \in \mathbb{R}^{d \times d}$ such that

$$Z^{\dagger} = \Omega \circ \Phi^{\dagger}(X) \text{ where the components of } Z^{\dagger} \text{ are mutually independent} \tag{2.5}$$

Note that a solution to the above equation (2.5) does not always exist. However, if we can find a Ω that satisfies the above (2.5), then the classifier $\Theta \circ \Omega^{-1}$ and the representation $\Omega \circ \Phi^{\dagger}(X)$ together solve the IC-ERM (2.3) assuming $\Theta \circ \Omega^{-1} \in \mathcal{H}_{\Theta}$ and $\Omega \circ \Phi^{\dagger} \in \mathcal{H}_{\Phi}$. To find a solution to the equation (2.5) we resort to the approach of linear ICA [Comon, 1994]. The approach has two steps. We first whiten $\Phi^\dagger(X)$, and for simplicity, we assume[*] $\Phi^\dagger(X)$ is zero mean, although our analysis extends to the non-zero mean case as well. Define V to be the covariance matrix of $\Phi^\dagger(X)$. If the covariance V is invertible[†], then the eigendecomposition of V is given as $V = U\Lambda^2 U^\mathsf{T}$. We obtain the whitened data $\Phi^*(X)$ as follows $\Phi^*(X) = \Lambda^{-1}U^\mathsf{T}\Phi^\dagger(X)$. Consider a linear transformation of the whitened data and denote it as $Z^* = \Omega \circ \Phi^*(X)$ and construct another vector Z' such that its individual components are all independent and equal in distribution to the corresponding components in Z^*. Our goal is to find an Ω such that the mutual information between Z^* and Z' is minimized. To make dependence on Ω explicit, we denote the mutual information between Z' and Z^* as $I(\Omega \circ \Phi^*(X))$. We state this as the following optimization

$$\Omega^\dagger \in \underset{\Omega, \Omega \text{ is invertible}}{\arg\min} \quad I(\Omega \circ \Phi^*(X)) \tag{2.6}$$

We denote the above two step approximation method as ERM-ICA and summarize it below:

- **ERM Phase:** Learn $\Theta^\dagger, \Phi^\dagger$ by solving the ERM objective (Eq: 2.4).
- **ICA Phase:** Learn $\Omega^\dagger$ by linear ICA (Eq: 2.6) on the representation from ERM Phase ($\Phi^\dagger$).

The above ERM-ICA procedure outputs a classifier $\Theta^\dagger \circ (\Omega^\dagger)^{-1}$ and representation $\Omega^\dagger \circ \Phi^\dagger$ that is an approximate solution to the IC-ERM problem (2.3). While the proposed ERM-ICA procedure is a simple approximation, we do not know of other works that have investigated this approach theoretically and experimentally for recovering the latents. Despite its simplicity, we can show (similar to Theorem 2.3.3) that ERM-ICA achieves permutation & scaling identification. The proof for the same can be found in Appendix C.3.

Theorem 2.4.1. *If Assumptions 2.2.1, 2.3.2 hold and the number of tasks k is equal to the dimension of the latent d, then the solution $\Omega^\dagger \circ \Phi^\dagger$ to ERM-ICA ((2.4), (2.6)) with ℓ as square loss for regression and cross-entropy loss for classification identifies true Z up to permutation and scaling.*

2.5 Empirical Findings

In this section, we analyse how the practical implementations of our theory holds up for both regression and classification tasks. The code repository to reproduce these experiments can be accessed from the link: `https://github.com/divyat09/ood_identification`.

[*]We also assume $\Phi^\dagger(X)$ has finite second moments.

[†]If it is not invertible, then we first need to project $\Phi^\dagger(X)$ into the range space of V

2.5.1 Experiment Setup

Data Generation Process. We use the data generation process described in Assumption 2.2.1. The components of Z are i.i.d. and follow discrete uniform $\{0, 1\}$ distribution. Each element of the task coefficient matrix Γ is i.i.d. and follows a standard normal distribution. The noise in the label generation is also standard normal. We use a 2-layer invertible MLP to model g and follow the construction used in Zimmermann et al. [2021].[*] We carry out comparisons for three settings, $d = \{16, 24, 50\}$, and vary tasks from $k = \{\frac{d}{2}, \frac{3d}{4}, d\}$. The dataset size used for training and test is 5000 data points, along with a validation set of 1250 data points for hyper parameter tuning. For the classification task, we have an additional step for converting the regression labels to binary labels as $Y \leftarrow \mathsf{Bernoulli}(\sigma(\Gamma Z))$. Also, the noise in the Γ sampling is set to a higher value (10 times that of a standard normal), as otherwise the Bayes optimal accuracy is much smaller.

Baselines. We compare our method (**ERM-ICA**) against two natural baselines. **a) ERM.** In this case, we carry out standard ERM (2.4) and use the representation learned at the layer before the output layer. **b) ERM-PCA.** In this case, we carry out standard ERM (2.4) and extract the representation learned at the layer before the output layer. We then take the extracted representation and transform it using principal component analysis (PCA). In other words, we analyze the representation in the eigenbasis of its covariance matrix. **c) ERM-ICA.** This is the main approach ((2.4), (2.5)) that approximates the IC-ERM objective (2.3). Here we take the representations learnt using ERM (2.4) and transform them using linear independent component analysis (ICA) (2.6). We define mean square error and cross-entropy as our loss objectives for the regression and classification task respectively.

Model parameters and evaluation metrics. We use a two layer fully connected neural network and train the model using stochastic gradient descent. The architecture and the hyperparameter details for the different settings are provided in Appendix C.6. The models are evaluated on the test dataset using the following two metrics.

To measure the label (Y) prediction performance, we use the average R^2 (coefficient of determination) and the average accuracy across tasks in the regression and classification task respectively. We check this metric to ensure that the representations do not lose any information about the label. To measure the latent identification performance, we use the MCC metric (E.q. 1.13), since it is specifically designed to capture the permutation & scaling identifiability.

[*]`https://github.com/brendel-group/cl-ica/blob/master/main_mlp.py`

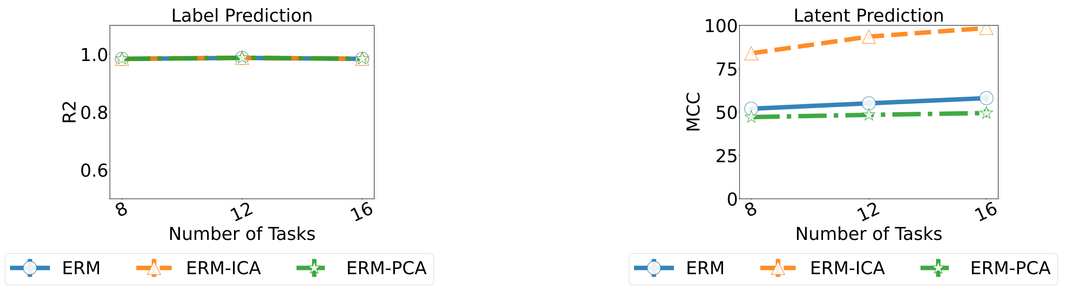


Figure 2.2: Comparison of label and latent prediction performance (regression, $d = 16$).

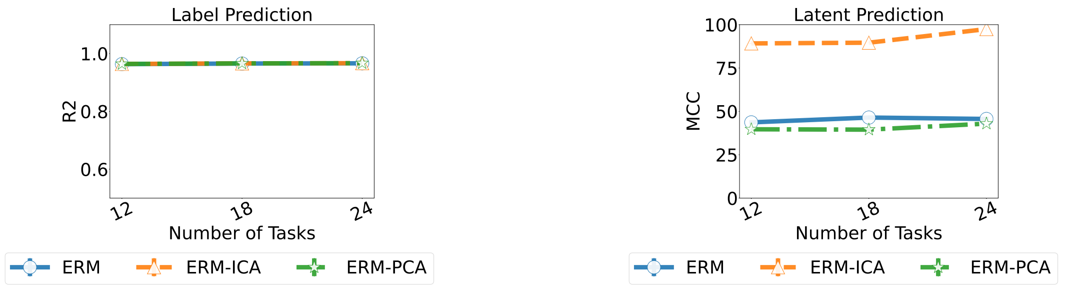


Figure 2.3: Comparison of label and latent prediction performance (regression, $d = 24$).

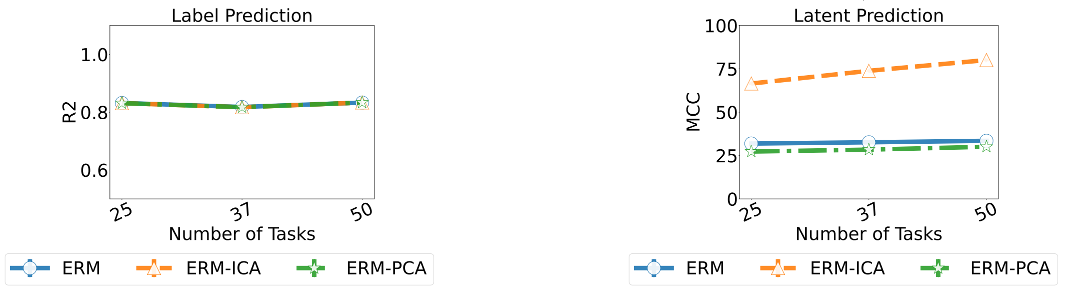


Figure 2.4: Comparison of label and latent prediction performance (regression, $d = 50$).

2.5.2 Results

Regression. Figure 2.2, 2.3 and 2.4 show a comparison of the performance of the three approaches across $d = 16$, $d = 24$, and $d = 50$ for various tasks. In all the cases, we find that the method ERM-ICA is significantly better than the other approaches in terms of guaranteeing permutation and scaling based identification. All three approaches have similar label prediction performance.

Classification. Figure 2.5, 2.6. and 2.7 show a comparison of the performance of the three approaches across $d = 16$, $d = 24$, and $d = 50$ for various tasks. In both cases, we find that the method ERM-ICA is better than the other approaches in terms of guaranteeing permutation and scaling based identification, except in the case of 24 data dimensions with a total of 12 tasks. All three approaches have similar label prediction performance. For classification, unlike regression, the improvements are smaller and we do not see improvement

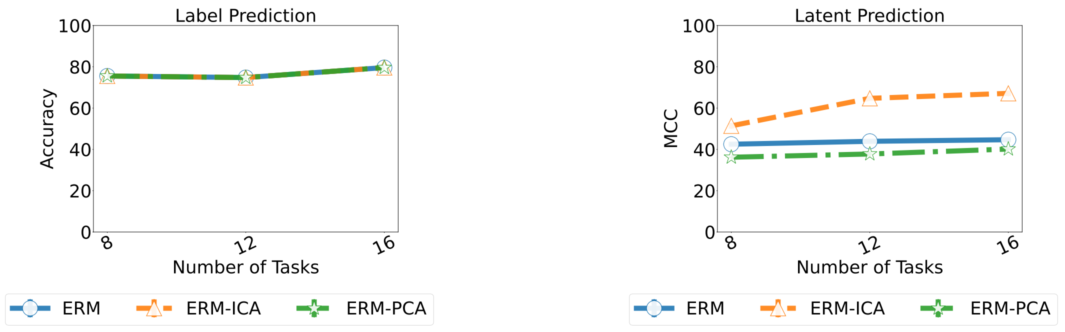


Figure 2.5: Comparison of label and latent prediction performance (classification, $d = 16$)

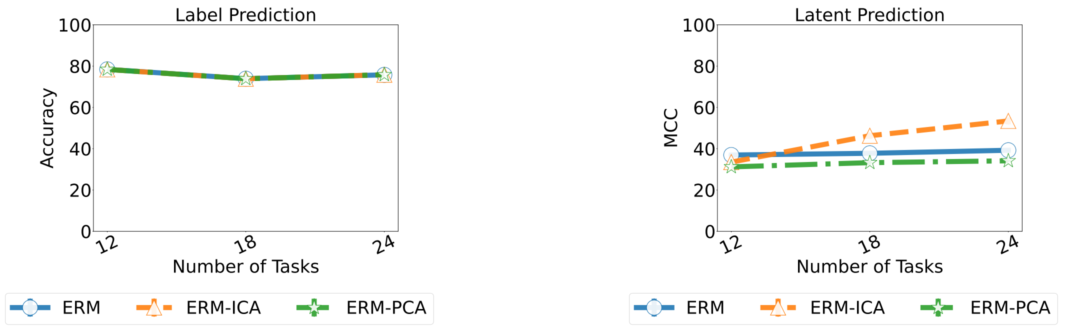


Figure 2.6: Comparison of label and latent prediction performance (classification, $d = 24$)

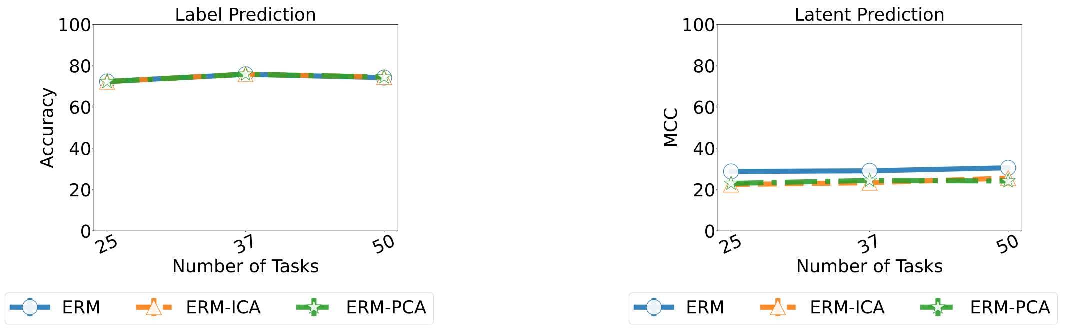


Figure 2.7: Comparison of label and latent prediction performance (classification, $d = 50$)

for $d = 50$. We see a worse performance in the classification setting because the ERM model does not learn the Bayes optimal predictor unlike regression.

Discussion. We have shown that ERM-ICA achieves significant improvement in latent recovery with much fewer tasks (up to $\frac{d}{2}$). Note that we only approximate equation (2.3) with ERM-ICA, and if we build better approximations of the ideal approach (IC-ERM), then we can witness gains with even fewer tasks. We believe that building such approximations is a fruitful future work.

2.6 Conclusion

In this work, we analyzed the problem of disentanglement in a natural setting, where latent factors cause the labels, a setting not well studied in the ICA literature. We show that if ERM is constrained to learn independent representations, then we can have latent recovery from learnt representations even when the number of tasks is small. We propose a simple two step approximate procedure (ERM-ICA) to solve the constrained ERM problem, and show that it is effective in a variety of experiments. Our analysis highlights the importance of learning independent representations and motivates the development of further approaches to achieve the same in practice.

A limitation of our work is the restriction of mutual independence in the latent space, which prevents us from using this approach for more practical scenarios where latent variables maybe correlated. Recent work by Lachapelle et al. [2022a] also deals with the same setup of multi-task supervised learning and made additional assumptions on the sparsity of the labeling function Γ (map from latent vectors Z to labels Y), which allows them to extended latent identification beyond mutually independence latent variables. Further, Fumero et al. [2024] also propose an identifiable sparsity guided multi-task learning method and benchmark it on several real-world datasets for robustness to distribution shifts. Hence, assumptions of sparsity on Γ in our framework is a promising direction for identifiable multi-task learning.

Another way to move beyond mutually independent latents in our setup is to consider the works on factorized support [Wang and Jordan, 2021, Roth et al., 2023]. Factorized/Independent support refers to the case when the DAG describing the latent variables does not contain any edges, however, there is correlation due to the presence of an (unobserved) confounder. Hence, latent variables with factorized support goes one step beyond mutually independent latents for modeling scenarios with correlations. Recent work by [Ahuja et al., 2023] provides latent identification results for latents with factorized support and linear mixing functions to generate observations from them. Since in our framework ERM already obtains linear identification (Appendix C.1), we could use the result for identification of factorized support latent under linear mixing to achieve permutation & scaling identification. The natural extension of our ERM-ICA algorithm would also be to incorporate independence of support penalty (IOSS) [Wang and Jordan, 2021, Ahuja et al., 2023] instead of Linear ICA in the second step.

Chapter 3

Additive Decoders for Latent Variables Identification and Cartesian-Product Extrapolation

This chapter is based on the contents of the paper "Additive decoders for latent variables identification and cartesian-product extrapolation" by Sébastien Lachapelle*, **Divyat Mahajan***, Ioannis Mitliagkas, and Simon Lacoste-Julien. The paper was accepted at the Advances in Neural Information Processing Systems [**NeurIPS 2023**] for an oral presentation.

Sébastien Lachapelle proposed the original project idea and developed the theory for latent identification, with input from Simon Lacoste-Julien. Divyat Mahajan led the experimental design under the guidance of Sébastien Lachapelle and obtained all the empirical results. Additionally, Divyat Mahajan played an active role in discussions regarding the theoretical results on local disentanglement and extrapolation. Sébastien Lachapelle led the writing, with Divyat Mahajan contributing specifically to the experiments section. Ioannis Mitliagkas and Simon Lacoste-Julien provided supervision throughout the project.

3.1 Introduction

In this chapter we focus on the question on unsupervised disentangled representation learning and aim to provide latent identification results without additional weak supervision. As we have seen that without assumptions on the mixing function identification is hopeless for the unsupervised case (Theorem 1.2.5), we focus on decoder architectures from *Object-centric representation learning* (OCRL) as an inspiration. OCRL aims to learn a representation in which the information about different objects are encoded separately [Eslami et al., 2016,

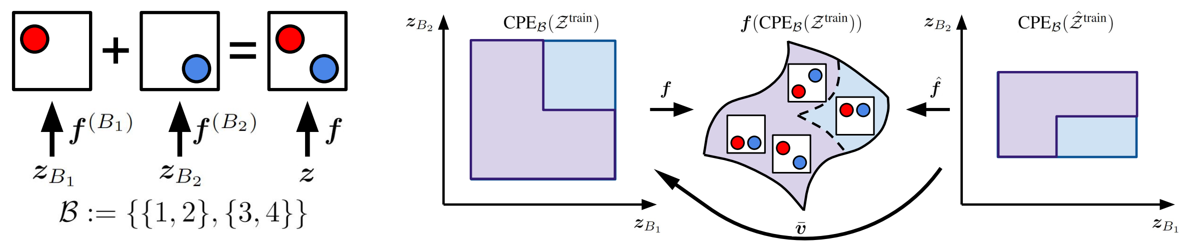

$f^{(B_1)}$
$f^{(B_2)}$
f
z_{B_1}
z_{B_2}
z
$\mathcal{B} := \{\{1,2\},\{3,4\}\}$
z_{B_2}
$\mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}})$
z_{B_1}
f
$f(\mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}}))$
$\hat{f}$
$\tilde{v}$
z_{B_2}
$\mathrm{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\mathrm{train}})$
z_{B_1}

Figure 3.1: **Left:** Additive decoders model the additive structure of scenes composed of multiple objects. **Right:** Additive decoders allow to generate novel images never seen during training via Cartesian-product extrapolation (Corollary 3.4.8). Purple regions correspond to latents/observations seen during training. The blue regions correspond to the Cartesian-product extension. The middle set is the manifold of images of balls. In this example, the learner never saw both balls high, but these can be generated nevertheless thanks to the additive nature of the scene. **Note.** The figure has a small inconsistency that it uses $\boldsymbol{f}$ to denote the decoder, while we use g everywhere in the chapter for the same.

Greff et al., 2016, Burgess et al., 2019, Greff et al., 2019, Engelcke et al., 2020, Locatello et al., 2020c]. These approaches have shown impressive results empirically, but the exact reason why they can perform this form of segmentation without any supervision is poorly understood.

Our first contribution is an analysis of the identifiability of a class of decoders we call *additive* (Definition 3.3.2). Essentially, a decoder $g(z)$ acting on a latent vector $z \in \mathbb{R}^{d_z}$ to produce an observation x is said to be additive if it can be written as $g(z) = \sum_{B \in \mathcal{B}} g^{(B)}(z_B)$ where $\mathcal{B}$ is a partition of $\{1, \dots, d_z\}$, $g^{(B)}(z_B)$ are "block-specific" decoders and the z_B are non-overlapping subvectors of z. This class of decoder is particularly well suited for images x that can be expressed as a sum of images corresponding to different objects (left of Figure 3.1). Unsurprisingly, this class of decoder bears similarity with the decoding architectures used in OCRL (Section 3.2), which already showed important successes at disentangling objects without any supervision. Our identifiability results provide conditions under which exactly solving the reconstruction problem with an additive decoder identifies the latent blocks z_B up to permutation and block-wise transformations (Theorems 3.4.4 & 3.4.6). We believe these results will be of interest to both the OCRL community, as they partly explain the empirical success of these approaches, and to the nonlinear ICA and disentanglement community, as it provides an important special case where identifiability holds. This result relies on the block-specific decoders being "sufficiently nonlinear" (Assumption 3.4.5) and requires only very weak assumptions on the distribution of the ground-truth latent factors of variations. In particular, these factors can be statistically dependent and their support can be (almost) arbitrary.

Our second contribution is to show theoretically that additive decoders can generate images never seen during training by recombining observed factors of variations in novel ways (Corollary 3.4.8). To describe this ability, we coin the term "Cartesian-product extrapolation" (right of Figure 3.1), which is similar to compositional generalization in terms of the latent factors of variation. We believe the type of identifiability analysis laid out in this work to understand "out-of-support" generation is novel and could be applied to other function classes or learning algorithms such as DALLE-2 [Ramesh et al., 2022] and Stable Diffusion [Rombach et al., 2022] to understand their apparent creativity and hopefully improve it.

Both latent variables identification and Cartesian-product extrapolation are validated experimentally[*] on simulated data (Section 3.5). We observe that additivity is crucial for both by comparing against a non-additive decoder which fails to disentangle and extrapolate.

3.2 Background & Literature review

Relation to Non-Linear ICA. In Section 1.2.3, we discussed several works utilizing auxiliary information for solving Non-Linear ICA. In contrast, our approach does not rely on additional information/weak supervision and obtain latent identification via constraints on the mixing function. Our approach departs from the standard nonlinear ICA problem along three axes: (i) we restrict the mixing function to be additive, (ii) the factors do not have to be necessarily independent, and (iii) we can identify only the blocks $\boldsymbol{z}_B$ as opposed to each $\boldsymbol{z}_i$ individually up to element-wise transformations, unless $\mathcal{B} = \{\{1\}, ..., \{d_z\}\}$ (see Section 3.4.1). Our results make mild assumptions about the latent distribution, which can present statistical dependencies and have an almost arbitrarily shaped support. Additionally, none of the prior works on Non-Linear ICA provide extrapolation guarantees as we do in Section 3.4.2.

Object-centric representation learning (OCRL). Lin et al. [2020] classified OCRL methods in two categories: *scene mixture models* [Greff et al., 2016, 2017, 2019, Locatello et al., 2020c] & *spatial-attention models* [Eslami et al., 2016, Crawford and Pineau, 2019, Burgess et al., 2019, Engelcke et al., 2020]. Additive decoders can be seen as an approximation to the decoding architectures used in the former category, which typically consist of an object-specific decoder $g^{(\text{obj})}$ acting on object-specific latent blocks z_B and "mixed" together

[*]The code repository to reproduce the experiments: `https://github.com/divyat09/additive_decoder_extrapolation`

via a masking mechanism $m^{(B)}(z)$ which selects which pixel belongs to which object.

$$g(z) = \sum_{B \in \mathcal{B}} m^{(B)}(z) \odot g^{(\text{obj})}(z_B) \text{ , where } m_k^{(B)}(z) = \frac{\exp(a_k(z_B))}{\sum_{B' \in \mathcal{B}} \exp(a_k(z_{B'}))} \text{ ,} \tag{3.1}$$

where $\mathcal{B}$ is a partition of $[d_z]$ made of equal-size blocks B and $a : \mathbb{R}^{|B|} \to \mathbb{R}^{d_x}$ outputs a score that is normalized via a softmax operation to obtain the masks $m^{(B)}(z)$. Many of these works also present some mechanism to select dynamically how many objects are present in the scene and thus have a variable-size representation z, an important technical aspect we omit in our analysis. Empirically, training these decoders based on some form of reconstruction objective, probabilistic or not, yields latent blocks z_B that represent the information of individual objects separately. We believe our work constitutes a step towards providing a mathematically grounded explanation for why these approaches can perform this form of disentanglement without supervision (Theorems 3.4.4 & 3.4.6). Many architectural innovations in scene mixture models concern the encoder, but our analysis focuses solely on the structure of the decoder $g(z)$, which is a shared aspect across multiple methods. Generalization capabilities of object-centric representations were studied empirically by Dittadi et al. [2021] but did not cover Cartesian-product extrapolation (Corollary 3.4.8).

Differences with OCRL in practice. Although additive decoders make intuitive sense for OCRL, they are not expressive enough to represent the "masked decoders" typically used in practice (Equation (3.1)). The lack of additivity stems from the normalization in the masks $m^{(B)}(z)$. We hypothesize that studying the simpler additive decoders might still reveal interesting phenomena present in modern OCRL approaches due to their resemblance. Another difference is that, in practice, the same object-specific decoder $g^{(\text{obj})}$ is applied to every latent block z_B. Our theory allows for these functions to be different, but also applies when functions are the same. Additionally, this parameter sharing across $g^{(B)}$ enables modern methods to have a variable number of objects across samples, an important practical point our theory does not cover.

Extrapolation. Du and Mordatch [2019] studied empirically how one can combine energy-based models for what they call *compositional generalization*, which is similar to our notion of Cartesian-product extrapolation, but suppose access to datasets in which only one latent factor varies and do not provide any theory. Webb et al. [2020] studied extrapolation empirically and proposed a novel benchmark which does not have an additive structure. Besserve et al. [2021] proposed a theoretical framework in which out-of-distribution samples are obtained by applying a transformation to a single hidden layer inside the decoder network. Krueger et al. [2021] introduced a domain generalization method which is trained to be robust to tasks falling outside the convex hull of training distributions. Extrapolation in text-conditioned image generation was recently discussed by Wang et al. [2023].

3.3 Problem Setup

Assumption 3.3.1 (Data-generating process)**.** The set of possible observations is given by a lower dimensional manifold $g(\mathcal{Z}^{\text{test}})$ embedded in $\mathbb{R}^{d_x}$ where $\mathcal{Z}^{\text{test}}$ is an open set of $\mathbb{R}^{d_z}$ and $g : \mathcal{Z}^{\text{test}} \to \mathbb{R}^{d_x}$ is a C^2-diffeomorphism onto its image. We will refer to g as the *ground-truth decoder*. At training time, the observations are i.i.d. samples given by $x = g(z)$ where $z \sim Z$ is distributed according to the probability measure $\mathbb{P}^{\text{train}}_Z$ with support $\mathcal{Z}^{\text{train}} \subset \mathcal{Z}^{\text{test}}$. Throughout, we assume that $\mathcal{Z}^{\text{train}}$ is regularly closed (Definition D.1.1).

Intuitively, the ground-truth decoder g is effectively relating the "natural factors of variations" z to the observations x in a one-to-one fashion. Mansouri et al. [2022] pointed out that the injectivity of g is violated when images show two objects that are indistinguishable, an important practical case that is not covered by our theory.

We emphasize the distinction between $\mathcal{Z}^{\text{train}}$, which corresponds to the observations seen during training, and $\mathcal{Z}^{\text{test}}$, which corresponds to the set of all possible images. The case where $\mathcal{Z}^{\text{train}} \neq \mathcal{Z}^{\text{test}}$ will be of particular interest when discussing extrapolation in Section 3.4.2. The "regularly closed" condition on $\mathcal{Z}^{\text{train}}$ is mild, as it is satisfied as soon as the distribution of Z has a density w.r.t. the Lebesgue measure on $\mathbb{R}^{d_z}$. It is violated, for example, when Z is a discrete random vector. Figure 3.2 illustrates this assumption with simple examples.

Objective. Our analysis is based on the objective of reconstructing the observations x by learning an encoder $\hat{f} : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$ and a decoder $\hat{g} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$. Note that we assumed implicitly that the dimensionality of the learned representation matches the dimensionality of the ground-truth. We define the set of latent codes the encoder can output when evaluated on the training distribution:

$$\hat{\mathcal{Z}}^{\text{train}} := \hat{f}(g(\mathcal{Z}^{\text{train}})) \tag{3.2}$$

We can characterize the indeterminacy in latent recovery with $v : \hat{\mathcal{Z}}^{\text{train}} \to \mathcal{Z}^{\text{train}}$ when the reconstruction tasks is solved exactly (Appendix B.3) as follows:

$$v := g^{-1} \circ \hat{g} \tag{3.3}$$

Before introducing our formal definition of additive decoders, we introduce the following notation: Given a set $\mathcal{Z} \subset \mathbb{R}^{d_z}$ and a subset of indices $B \subset [d_z]$, let us define $\mathcal{Z}_B$ to be the projection of $\mathcal{Z}$ onto dimensions labelled by the index set B. More formally,

$$\mathcal{Z}_B := \{z_B \mid z \in \mathcal{Z}\} \subseteq \mathbb{R}^{|B|} \, . \tag{3.4}$$

Intuitively, we will say that a decoder is *additive* when its output is the summation of the outputs of "object-specific" decoders that depend only on each latent block z_B. This captures the idea that an image can be seen as the juxatoposition of multiple images which individually correspond to objects in the scene or natural factors of variations (left of Figure 3.1).

Definition 3.3.2 (Additive functions)**.** Let $\mathcal{B}$ be a partition of $[d_z]$[†]. A function $g : \mathcal{Z} \to \mathbb{R}^{d_x}$ is said to be **additive** if there exist functions $g^{(B)} : \mathcal{Z}_B \to \mathbb{R}^{d_x}$ for all $B \in \mathcal{B}$ such that

$$\forall z \in \mathcal{Z}, g(z) = \sum_{B \in \mathcal{B}} g^{(B)}(z_B) \, . \tag{3.5}$$

This additivity property will be central to our analysis as it will be the driving force of identifiability (Theorem 3.4.4 & 3.4.6) and Cartesian-product extrapolation (Corollary 3.4.8).

Remark 3.3.3. Suppose we have $x = \sigma(\sum_{B \in \mathcal{B}} g^{(B)}(z_B))$ where σ is a known bijective function. For example, if $\sigma(y) := \exp(y)$ (component-wise), the decoder can be thought of as being *multiplicative.* Our results still apply since we can simply transform the data doing $\tilde{x} := \sigma^{-1}(x)$ to recover the additive form $\tilde{x} = \sum_{B \in \mathcal{B}} g^{(B)}(z_B)$.

3.4 Identifiability Analysis of Additive Decoders

3.4.1 Latent Identification

We now study the identifiability of additive decoders and show how they can yield disentanglement. Our definition of disentanglement will rely on *partition-respecting permutations*:

Definition 3.4.1 (Partition-respecting permutations)**.** Let $\mathcal{B}$ be a partition of $\{1, ..., d_z\}$. A permutation π over $\{1, ..., d_z\}$ respects $\mathcal{B}$ if, for all $B \in \mathcal{B}, \; \pi(B) \in \mathcal{B}$.

Essentially, a permutation that respects $\mathcal{B}$ is one which can permute blocks of $\mathcal{B}$ and permute elements within a block, but cannot "mix" blocks together. We now introduce $\mathcal{B}$-disentanglement.

Definition 3.4.2 ($\mathcal{B}$-disentanglement)**.** A learned decoder $\hat{g} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$ is said to be **$\mathcal{B}$-disentangled** w.r.t. the ground-truth decoder g when $g(\mathcal{Z}^{\text{train}}) = \hat{g}(\hat{\mathcal{Z}}^{\text{train}})$ and the mapping

[†]Without loss of generality, we assume that the partition $\mathcal{B}$ is contiguous, i.e. each $B \in \mathcal{B}$ can be written as $B = \{i+1, i+2, \ldots, i+|B|\}$.

$v := g^{-1} \circ \hat{g}$ is a diffeomorphism from $\hat{\mathcal{Z}}^{\text{train}}$ to $\mathcal{Z}^{\text{train}}$ satisfying the following property: there exists a permutation π respecting $\mathcal{B}$ such that, for all $B \in \mathcal{B}$, there exists a function $\bar{v}_{\pi(B)} : \hat{\mathcal{Z}}^{\text{train}}_B \to \mathcal{Z}^{\text{train}}_{\pi(B)}$ such that, for all $z \in \hat{\mathcal{Z}}^{\text{train}}$, $v_{\pi(B)}(z) = \bar{v}_{\pi(B)}(z_B)$. In other words, $v_{\pi(B)}(z)$ depends *only* on z_B.

Thus, $\mathcal{B}$-disentanglement means that the blocks of latent dimensions z_B are disentangled from one another, but that variables within a given block might remain entangled. Note that, unless the partition is $\mathcal{B} = \{\{1\}, \cdots, \{d_z\}\}$, this corresponds to a weaker form of disentanglement than what is typically seeked in nonlinear ICA, i.e. recovering each variable individually. To illustrate $\mathcal{B}$-disentanglement, imagine a scene consisting of two balls moving around in 2D where the "ground-truth" representation is given by $z = (x^1, y^1, x^2, y^2)$ where $z_{B_1} = (x^1, y^1)$ and $z_{B_2} = (x^2, y^2)$ are the coordinates of each ball (here, $\mathcal{B} := \{\{1, 2\}, \{3, 4\}\}$). In that case, a learned representation is $\mathcal{B}$-disentangled when the balls are disentangled from one another. However, the basis in which the position of each ball is represented might differ in both representations.

Our first result (Theorem 3.4.4) shows a weaker form of disentanglement we call *local* $\mathcal{B}$-disentanglement. This means the Jacobian matrix of v, Dv, has a "block-permutation" structure everywhere (Appendix B.2).

Definition 3.4.3 (Local $\mathcal{B}$-disentanglement)**.** A learned decoder $\hat{g} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_x}$ is said to be **locally $\mathcal{B}$-disentangled** w.r.t. the ground-truth decoder g when $g(\mathcal{Z}^{\text{train}}) = \hat{g}(\hat{\mathcal{Z}}^{\text{train}})$ and the mapping $v := g^{-1} \circ \hat{g}$ is a diffeomorphism from $\hat{\mathcal{Z}}^{\text{train}}$ to $\mathcal{Z}^{\text{train}}$ satisfying the following property: for all $z \in \hat{\mathcal{Z}}^{\text{train}}$, there exists a permutation π respecting $\mathcal{B}$ such that, for all $B \in \mathcal{B}$, the columns of $Dv_{\pi(B)}(z) \in \mathbb{R}^{|B| \times d_z}$ outside block B are zero.

We now state the main identifiability result of this work which provides conditions to guarantee *local* disentanglement. We will then see how to go from local to *global* disentanglement in the subsequent Theorem 3.4.6. For pedagogical reasons, we delay the formalization of the sufficient nonlinearity Assumption 3.4.5 on which the result crucially relies.

Theorem 3.4.4 (Local disentanglement via additive decoders)**.** *Given the data generation process (Assumption 3.3.1) and the optimal solution $(\hat{f}, \hat{g})$ under the reconstruction loss, along with the extra assumptions stated below:*

- *Both true decoder g and learned $\hat{g}$ are additive functions (Definition 3.3.2)*
- *Learned decoder $\hat{g}$ is a C^2-diffeomorphism, the learned encoder $\hat{f}$ is continuous*
- *True deoder g is sufficiently nonlinear as formalized by Assumption 3.4.5*

Then we have that $\hat{g}$ is locally $\mathcal{B}$-disentangled w.r.t. g (Definition 3.4.3) .

The proof can be found in Appendix D.1.2, which is inspired from Hyvärinen et al. [2019]. The essential differences are that (i) they leverage the additivity of the conditional log-density of z given an auxiliary variable u (i.e. conditional independence) instead of the additivity of the decoder function g, (ii) we extend their proof techniques to allow for "block" disentanglement, i.e. when $\mathcal{B}$ is not the trivial partition $\{\{1\}, \dots, \{d_z\}\}$, (iii) the asssumption "sufficient variability" of the prior $p(z \mid u)$ of Hyvärinen et al. [2019] is replaced by an analogous assumption of "sufficient nonlinearity" of the decoder g (Assumption 3.4.5), and (iv) we consider much more general supports $\mathcal{Z}^{\text{train}}$ which makes the jump from local to global disentanglement less direct in our case.

The identifiability-expressivity trade-off. The level of granularity of the partition $\mathcal{B}$ controls the trade-off between identifiability and expressivity: the finer the partition, the tighter the identifiability guarantee but the less expressive is the function class. The optimal level of granularity is going to dependent on the application at hand. Whether $\mathcal{B}$ could be learned from data is left for future work.

Sufficient nonlinearity. The following assumption is key in proving Theorem 3.4.6, as it requires that the ground-truth decoder is "sufficiently nonlinear".

Assumption 3.4.5 (Sufficient nonlinearity of g)**.** Let $q := d_z + \sum_{B \in \mathcal{B}} \frac{|B|(|B|+1)}{2}$. For all $z \in \mathcal{Z}^{\text{train}}$, g is such that the following matrix has linearly independent columns (i.e. full column-rank):

$$W(z) := \left[\left[D_i g^{(B)}(z_B) \right]_{i \in B} \ \left[D^2_{i,i'} g^{(B)}(z_B) \right]_{(i,i') \in B^2_{\leq}} \right]_{B \in \mathcal{B}} \in \mathbb{R}^{d_x \times q} \,, \tag{3.6}$$

where $B^2_{\leq} := B^2 \cap \{(i, i') \mid i' \leq i\}$. Note this implies $d_x \geq q$.

This is reminiscent of the "sufficient variability" assumptions found in the nonlinear ICA litterature, which usually concerns the distribution of the latent variable $\boldsymbol{z}$ as opposed to the decoder $\boldsymbol{f}$ [Hyvärinen and Morioka, 2016, 2017, Hyvärinen et al., 2019, Khemakhem et al., 2020b,c, Lachapelle et al., 2022b, Zheng et al., 2022]. We clarify this link in Appendix D.1.3 and provide intuitions why sufficient nonlinearity can be satisfied when $d_x \gg d_z$.

From local to global disentanglement The following result provides additional assumptions to guarantee *global* disentanglement (Definition 3.4.2) as opposed to only local disentanglement (Definition 3.4.3). See Appendix D.1.4 for its proof.

Theorem 3.4.6 (Global disentanglement via additive decoders)**.** *Suppose that all the assumptions of Theorem 3.4.4 hold, along with the extra assumptions stated below:*

- *Support of the true latents $\mathcal{Z}^{\text{train}}$ is path-connected (Definition D.1.3)*
- *Block-specific decoders $g^{(B)}$ and $\hat{g}^{(B)}$ are injective for all blocks $B \in \mathcal{B}$*

Then for optimal solution $(\hat{f}, \hat{g})$ under the reconstruction loss we have $\hat{g}$ is (globally) $\mathcal{B}$-disentangled w.r.t. g (Definition 3.4.2). Further, for all $B \in \mathcal{B}$, we have the following:

$$\hat{g}^{(B)}(z_B) = g^{(\pi(B))}(\bar{v}_{\pi(B)}(z_B)) + c^{(B)}, \text{ for all } z_B \in \hat{\mathcal{Z}}_B^{\text{train}}, \tag{3.7}$$

where the functions $\bar{v}_{\pi(B)}$ are from Definition 3.4.2 and the vectors $c^{(B)} \in \mathbb{R}^{d_x}$ are constants such that $\sum_{B\in\mathcal{B}} c^{(B)} = 0$. We also have that the functions $\bar{v}_{\pi(B)} : \hat{\mathcal{Z}}_B^{\text{train}} \to \mathcal{Z}_{\pi(B)}^{\text{train}}$ are C^2-diffeomorphisms and have the following form:

$$\bar{v}_{\pi(B)}(z_B) = (g^{\pi(B)})^{-1}(\hat{g}^{(B)}(z_B) - c^{(B)}), \text{ for all } z_B \in \hat{\mathcal{Z}}_B^{\text{train}}. \tag{3.8}$$

Equation (3.7) in the above result shows that each block-specific learned decoder $\hat{g}^{(B)}$ is "imitating" a block-specific ground-truth decoder $g^{\pi(B)}$. Indeed, the "object-specific" image outputted by the decoder $\hat{g}^{(B)}$ evaluated at some $z_B \in \hat{\mathcal{Z}}_B^{\text{train}}$ is the same as the image outputted by $g^{(B)}$ evaluated at $v(z_B) \in \mathcal{Z}_B^{\text{train}}$, *up to an additive constant vector* $c^{(B)}$. These constants cancel each other out when taking the sum of the block-specific decoders. Equation (3.8) provides an explicit form for the function $\bar{v}_{\pi(B)}$, which is essentially the learned block-specific decoder composed with the inverse of the ground-truth block-specific decoder. Note that assuming that the support of $\mathbb{P}_z^{\text{train}}$, $\mathcal{Z}^{\text{train}}$, is **path-connected** (see Definition D.1.3 in appendix) is crucial to go from local to global disentanglement since it prevents the permutation π of Definition 3.4.3 from changing between two disconnected regions of $\hat{\mathcal{Z}}^{\text{train}}$. See Figure 3.2 for an illustration. In Appendix D.1.5, we discuss the additional assumption that each $g^{(B)}$ must be injective and show that, in general, it is not equivalent to the assumption that $\sum_{B\in\mathcal{B}} g^{(B)}$ is injective.

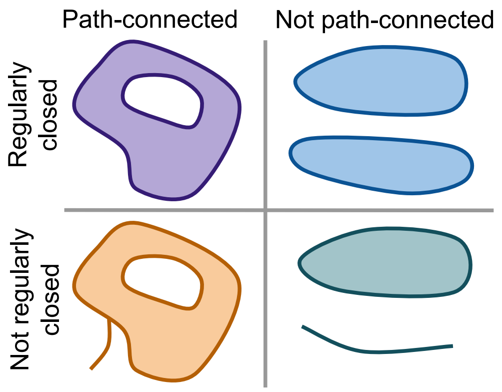


Figure 3.2: Illustrating regularly closed sets and path-connected sets. Theorem 3.4.6 requires $\mathcal{Z}^{\text{train}}$ to satisfy both properties.

3.4.2 Cartesian-product extrapolation

In this section, we show how a learned additive decoder can be used to generate images x that are "out of support" in the sense that $x \notin g(\mathcal{Z}^{\text{train}})$, but that are still on the manifold of "reasonable" images, i.e. $x \in g(\mathcal{Z}^{\text{test}})$. To characterize the set of images the learned decoder can generate, we will rely on the notion of "cartesian-product extension", which we define next.

Definition 3.4.7 (Cartesian-product extension)**.** Given a set $\mathcal{Z} \subseteq \mathbb{R}^{d_z}$ and partition $\mathcal{B}$ of $[d_z]$, we define the Cartesian-product extension of $\mathcal{Z}$ as

$$\text{CPE}_{\mathcal{B}}(\mathcal{Z}) := \prod_{B \in \mathcal{B}} \mathcal{Z}_B \text{ , where } \mathcal{Z}_B := \{z_B \mid z \in \mathcal{Z}\}.$$

Let us define $\bar{v} : \text{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\text{train}}) \to \text{CPE}_{\mathcal{B}}(\mathcal{Z}^{\text{train}})$ to be the natural extension of the function $v : \hat{\mathcal{Z}}^{\text{train}} \to \mathcal{Z}^{\text{train}}$. More explicitly, $\bar{v}$ is the "concatenation" of the functions $\bar{v}_B$ given in Definition 3.4.2:

$$\bar{v}(z)^\top := [\bar{v}_{B_1}(z_{\pi^{-1}(B_1)})^\top \cdots \bar{v}_{B_\ell}(z_{\pi^{-1}(B_\ell)})^\top] \,, \tag{3.9}$$

where ℓ is the number of blocks in $\mathcal{B}$. This map is a diffeomorphism because each $\bar{v}_{\pi(B)}$ is a diffeomorphism from $\hat{\mathcal{Z}}^{\text{train}}_B$ to $\mathcal{Z}^{\text{train}}_{\pi(B)}$ by Theorem 3.4.6. We already know that $\hat{g}(z) = g \circ \bar{v}(z)$ for all $z \in \hat{\mathcal{Z}}^{\text{train}}$. The following result shows that this equality holds in fact on the larger set $\text{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\text{train}})$, the Cartesian-product extension of $\hat{\mathcal{Z}}^{\text{train}}$. See right of Figure 3.1 for an illustration of the following corollary.

Corollary 3.4.8 (Cartesian-product extrapolation)**.** *Suppose all the assumptions of Theorem 3.4.6 hold. Then we have the following:*

$$\sum_{B \in \mathcal{B}} \hat{g}^{(B)}(z_B) = \sum_{B \in \mathcal{B}} g^{(\pi(B))}(\bar{v}_{\pi(B)}(z_B)) \; \forall z \in \text{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\text{train}}) \tag{3.10}$$

Furthermore, if $\text{CPE}_{\mathcal{B}}(\mathcal{Z}^{\text{train}}) \subseteq \mathcal{Z}^{\text{test}}$*, then* $\hat{g}(\text{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\text{train}})) \subseteq g(\mathcal{Z}^{\text{test}})$*.*

Equation (3.10) tells us that the learned decoder $\hat{g}$ "imitates" the ground-truth g not just over $\hat{\mathcal{Z}}^{\text{train}}$, but also over its Cartesian-product extension, hence we can generate observations never seen during training. Intuitively, cartesian-product extrapolation is akin to compositional generalization since the decoder is

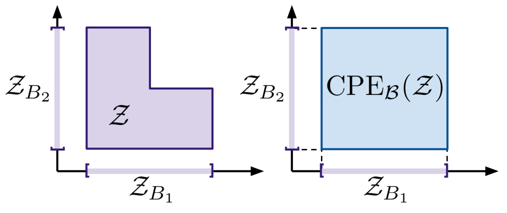


Figure 3.3: Illustration of Definition 3.4.7.

able to generate images corresponding to the novel
combinations of the latent factors by training on a set of limited combinations.

Disentanglement is not enough for extrapolation. To the best of our knowledge, Corollary 3.4.8 is the first result that formalizes how disentanglement can induce extrapolation. It illustrates that disentanglement alone is not sufficient to enable extrapolation and that one needs to restrict the hypothesis class of decoders in some way. Indeed, given a learned decoder $\hat{g}$ that is disentangled w.r.t. g on the training support $\mathcal{Z}^{\text{train}}$, one cannot guarantee both decoders will "agree" outside the training domain without further restricting $\hat{g}$ and g. This work has focused on "additivity", but we believe other restrictios could help as well.

3.5 Experiments

We now present empirical validations of the theoretical results presented earlier. To achieve this, we compare the ability of additive and non-additive decoders to both identify ground-truth latent factors (Theorems 3.4.4 & 3.4.6) and extrapolate (Corollary 3.4.8) when trained to solve the reconstruction task on simple images ($64 \times 64 \times 3$) consisting of two balls moving in space [Ahuja et al., 2022]. See Appendix D.2.1 for training details. We consider two datasets: one where the two ball positions can only vary along the y-axis (**ScalarLatents**) and one where the positions can vary along both the x and y axes (**BlockLatents**).

ScalarLatents: The ground-truth latent vector $\boldsymbol{z} \in \mathbb{R}^2$ is such that z_1 and z_2 corresponds to the height (y-coordinate) of the first and second ball, respectively. Thus the partition is simply $\mathcal{B} = \{\{1\}, \{2\}\}$. This simple setting is interesting since the low dimensionality of the latent space ($d_z = 2$) allows for exhaustive visualizations like Figure 3.4. To study Cartesian-product extrapolation, we sample $\boldsymbol{z}$ from a distribution with a L-shaped support given by $\mathcal{Z}^{\text{train}} := [0, 1] \times [0, 1] \setminus [0.5, 1] \times [0.5, 1]$, so that the training set does not contain images where both balls appear in the upper half of the image (see Appendix D.2.2).

BlockLatents: The ground-truth latent vector $\boldsymbol{z} \in \mathbb{R}^4$ is such that $z_{\{1,2\}}$ and $z_{\{3,4\}}$ correspond to the x, y position of the first and second ball, respectively (the partition is simply $\mathcal{B} = \{\{1, 2\}, \{3, 4\}\}$, i.e. each object has two latent factors). Thus, this more challenging setting illustrates "block-disentanglement". The latent z is sampled uniformly from the hypercube $[0, 1]^4$ but the images presenting occlusion (when a ball is behind another) are rejected from the dataset. We also present an additional version of this dataset where we sample from the hypercube $[0, 1]^4$ with dependencies. See Appendix D.2.2 for more details.

Evaluation metrics: To evaluate disentanglement, we compute a matrix of scores $(s_{B,B'}) \in \mathbb{R}^{\ell \times \ell}$ where ℓ is the number of blocks in $\mathcal{B}$ and $s_{B,B'}$ is a score measuring how well we

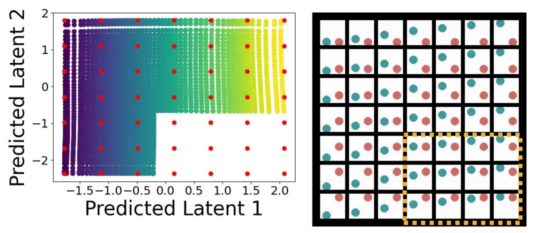
	ScalarLatents				**BlockLatents** (independent $\boldsymbol{z}$)		**BlockLatents** (dependent $\boldsymbol{z}$)	
Decoders	RMSE	$\text{LMS}_{\text{Spear}}$	RMSE^{OOS}	$\text{LMS}^{\text{OOS}}_{\text{Spear}}$	RMSE	LMS_{Tree}	RMSE	LMS_{Tree}
Non-add.	$.06_{\pm .002}$	$70.6_{\pm 5.21}$	$.18_{\pm .012}$	$73.7_{\pm 4.64}$	$.02_{\pm .001}$	$53.9_{\pm 7.58}$	$.02_{\pm .001}$	$78.1_{\pm 2.92}$
Additive	$.06_{\pm .002}$	$\mathbf{91.5}_{\pm \mathbf{3.57}}$	$.11_{\pm .018}$	$\mathbf{89.5}_{\pm \mathbf{5.02}}$	$.03_{\pm .012}$	$\mathbf{92.2}_{\pm \mathbf{4.91}}$	$.01_{\pm .002}$	$\mathbf{99.9}_{\pm \mathbf{0.02}}$

Table 3.1: Reporting reconstruction mean squared error (RMSE $\downarrow$) and the Latent Matching Score (LMS $\uparrow$) for the three datasets considered: **ScalarLatents** and **BlockLatents** with independent and dependent latents. Runs were repeated with 10 random initializations. RMSE^{OOS} and $\text{LMS}^{\text{OOS}}_{\text{Spear}}$ are the same metric but evaluated out of support (see Appendix D.2.3 for details). While the standard error is high, the differences are still clear as can be seen in their box plot version in Appendix D.2.4.

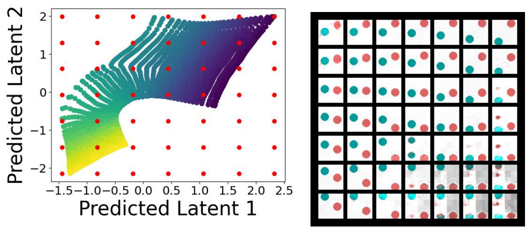
can predict the ground-truth block z_B from the learned latent block $\hat{z}_{B'} = \hat{f}_{B'}(x)$ outputted by the encoder. The final Latent Matching Score (LMS) is computed as LMS $= \arg\max_{\pi \in \mathfrak{S}_{\mathcal{B}}} \frac{1}{\ell} \sum_{B \in \mathcal{B}} s_{B,\pi(B)}$, where $\mathfrak{S}_{\mathcal{B}}$ is the set of permutations respecting $\mathcal{B}$ (Definition 3.4.1). When $\mathcal{B} := \{\{1\}, \dots, \{d_z\}\}$ and the score used is the absolute value of the correlation, LMS is simply the *mean correlation coefficient* (MCC) (1.13). Because our theory guarantees recovery of the latents only up to invertible and potentially nonlinear transformations, we use the Spearman correlation, which can capture nonlinear relationships unlike the Pearson correlation. We denote this score by $\text{LMS}_{\text{Spear}}$ and will use it in the dataset **ScalarLatents**. For the **BlockLatents** dataset, we cannot use Spearman correlation (because $\boldsymbol{z}_B$ are two dimensional). Instead, we take the score $s_{B,B'}$ to be the R^2 score of a regression tree. We denote this score by LMS_{tree}. There are subtleties to take care of when one wants to evaluate LMS_{tree} on a non-additive model due to the fact that the learned representation does not have a natural partition $\mathcal{B}$. We must thus search over partitions. We discuss this and provide further details on the metrics in Appendix D.2.3.

3.5.1 Results

Additivity is important for disentanglement. Table 3.1 shows that the additive decoder obtains a much higher $\text{LMS}_{\text{Spear}}$ & LMS_{Tree} than its non-additive counterpart on all three datasets considered, even if both decoders have very small reconstruction errors. This is corroborated by the visualizations of Figures 3.4 & 3.5. Appendix D.2.5 additionally shows object-specific reconstructions for the **BlockLatents** dataset. We emphasize that disentanglement is possible even when the latent factors are dependent (or causally related), as shown on the **ScalarLatents** dataset (L-shaped support implies dependencies) and on the **BlockLatents** dataset with dependencies (Table 3.1). Note that prior works have relied on interventions [Ahuja et al., 2023, 2022, Brehmer et al., 2022] or Cartesian-product

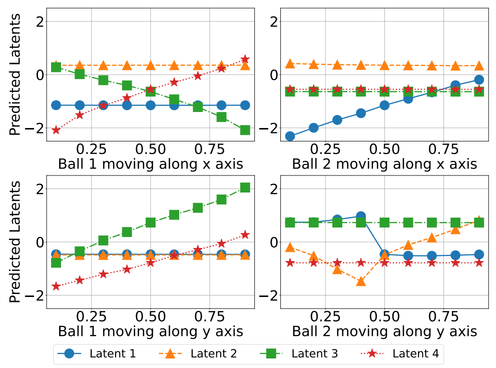


(a) Additive decoder

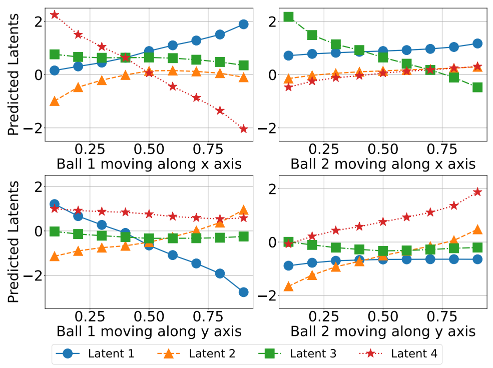


(b) Non-additive decoder

Figure 3.4: Figure (a) shows latent representation outputted by the encoder $\hat{f}(x)$ over the *training* dataset, and the corresponding reconstructed images of the additive decoder with median $\text{LMS}_{\text{Spear}}$ among runs performed on the **ScalarLatents** dataset. Figure (b) shows the same thing for the non-additive decoder. The color gradient corresponds to the value of one of the ground-truth factor, the red dots correspond to factors used to generate the images and the yellow dashed square highlights extrapolated images.



(a) Additive Decoder



(b) Non-Additive Decoder

Figure 3.5: Latent responses for the case of independent latents in the **BlockLatent** dataset. In each plot, we report the latent factors predicted from multiple images where one ball moves along only one axis at a time. For the additive case, at most two latents change, while more than two latents change for the non-additive case. See Appendix D.2.5 for details.

supports Wang and Jordan [2022], Roth et al. [2023] to deal with dependencies.

Additivity is important for Cartesian-product extrapolation. Figure 3.4 illustrates that the additive decoder can generate images that are outside the training domain (both balls in upper half of the image) while its non-additive counterpart cannot. Furthermore, Table 3.1 also corroborates this showing that the “out-of-support" (OOS) reconstruction MSE and $\text{LMS}_{\text{Spear}}$ (evaluated only on the samples never seen during training) are significantly better for the additive than for the non-additive decoder.

Importance of connected support. Theorem 3.4.6 required that the support of the latent factors, $\mathcal{Z}^{\text{train}}$, was path-connected. Appendix D.2.6 shows experiments where this assumption is violated, which yields lower $\text{LMS}_{\text{Spear}}$ for the additive decoder, thus highlighting the importance of this assumption.

3.6 Conclusion

We provided an in-depth identifiability analysis of *additive decoders*, which bears resemblance to decoders used in OCRL, and introduced a novel theoretical framework showing how this architecture can generate reasonable images never seen during training via "Cartesian-product extrapolation". We validated these results empirically and confirmed that additivity is indeed crucial. By studying rigorously how disentanglement can induce extrapolation, our work highlighted the necessity of restricting the decoder to extrapolate and set the stage for future works to explore disentanglement and extrapolation in other function classes. We postulate that the type of identifiability analysis introduced in this work has the potential of expanding our understanding of creativity in generative models, ultimately resulting in representations that generalize better.

A major limitation with additive decoders is that they cannot model occlusion. Occlusion occurs when an object is partially hidden behind another one. Consider the example of two images with each consisting of two objects, A and B. In both images, the position of object A is the same and in exactly one of the images, object B partially occludes object A. Since the position of object A did not change, its corresponding latent block z_A is also unchanged between both images. However, the pixels occupied by object A do change between both images because of occlusion. The issue is that, because of additivity, z_A and z_B cannot interact to make some pixels that belonged to object A "disappear" to be replaced by pixels of object B. In practice, object-centric representation learning methods rely on a masking mechanism which allows interactions between z_A and z_B (E.q. 3.1). Hence, future work involves extending our results for provable latent identification & extrapolation for the case of masked addition such that the case with object occlusion is handled. A concurrent work by Wiedemer et al. [2024] establishes provable extrapolation for a more general class of functions that can handle masked addition, however, they assume access to the true latent variables and their analysis does not show how to achieve provable disentanglement along with compositionality for the proposed class of mixing functions. Hence, it is still an important open problem to prove such results for a more general class of mixing functions than additive decoders.

Further, our work provides guarantees only for extrapolation of the decoder but not for the extrapolation of the encoder. The experiments with **ScalarLatents** Figure 3.4a show generation of novel images by traversing the learned latent space, and we never explored whether the latent variables generated by the encoder for novel images would be meaningful. An important future step would be to extend our analysis and method for guarantees on encoder extrapolation as well. Recent work by Wiedemer et al. [2023] proposes encoder consistency loss for the same, and it would be interesting to develop further on this problem.

Chapter 4

Future Work: Compositional Generalization with Additive Energy Models

This chapter is based on my ongoing project with Kartik Ahuja, Ioannis Mitliagkas, Mohammad Pezeshki, and Pascal Vincent. I am leading the project on both theoretical and empirical fronts. The project emerged from several brainstorming sessions between Pascal Vincent and Divyat Mahajan, where Pascal advocated for the use of additive energy models.

From the outset, Pascal Vincent and Divyat Mahajan worked on the proofs for the extrapolation of the generative case of additive energy models, with significant contributions from Kartik Ahuja and Ioannis Mitliagkas as the project progressed. Pascal Vincent worked on the exploration of the discriminative case of additive energy models, along with inputs from Divyat Mahajan. Kartik Ahuja introduced the idea of deriving probabilistic bounds for extrapolation with randomly observed training groups, which was further analyzed in detail by Ioannis Mitliagkas and Divyat Mahajan.

The empirical analysis of the proposed approach has been led by Divyat Mahajan, under guidance from the other collaborators. The writing has been primarily led by Divyat Mahajan, with contributions from Kartik Ahuja on the theoretical results.

4.1 Introduction

In the previous chapter, we looked at additive decoders and established guarantees for latent identification and cartesian-product extrapolation with them. An important aspect from the analysis of additive decoders is that we can guarantee the learned decoder would show compositional generalization, i.e, it can generate images corresponding to novel combinations of learned factors. To understand this result better, suppose that we observed the factors of variation or obtained them via some disentangled representation learning approach. Now if we want to build compositional models that can generalize to novel combinations of these factors, then one theoretically sound approach is to learn an additive model in terms of these factors. However, additive decoders have a limited expressivity as they do not allow for any interaction between factors, and it is important to focus on more expressive class of functions for which we can establish extrapolation guarantees.

In this work, we introduce additive energy models and study their compositional generalization abilities under the assumption that *we know the latent factors of variation* (z). This is done to simplify the setup and focus solely on compositional generalization first with observed factors. Additive energy models (4.3) formulate the conditional data distribution as boltzmann distribution, $p(x|z) = \frac{1}{B(z)} \exp(-E(x,z))$, where the energy function is additive in terms of the factors, $E(x,z) = \sum_i E_i(x, z_i)$. Hence, unlike additive decoders, we have a more flexible probabilistic mapping from factors (z) to observations (x), which supports interaction between factors via the partition function $B(z)$.

Another main objective of our analysis is to establish guarantees for compositional generalization with additive energy models for the case of discrete factors. Extrapolation with discrete factors was not studied in additive decoders since provable disentanglement was the primary aim for them, which makes it difficult to use discrete factors. Hence, we first start with Section 4.2 that discusses the challenges with discrete extrapolation of additive functions in detail, and provides novel results regarding the sufficient conditions (Corollary 4.2.4, Corollary 4.2.6) on the support of discrete factors for compositional generalization. This section can be interpreted as extending the cartesian-product extrapolation result with additive decoders to discrete factors.

Section 4.3 shifts the focus to additive energy models, and provides the theoretical results (Theorem 4.3.2) for provable extrapolation with them. Since estimation of partition function can be challenging in energy based models, we propose a novel approach inspired from energy based models that avoids estimating the partition function and yet can extrapolate to novel combinations (Theorem 4.3.5).

Finally, we evaluate the proposed method for the task of group-robust classification/subpopulation shifts [Yang et al., 2023] in Section 4.4. Our preliminary results are promising, and show that additive energy based methods achieve good performance for compositional generalization in high-dimensional image datasets.

4.2 Characterizing Extrapolation for Discrete Factors

Let $z \in \mathbb{R}^{d_z}$ denote the inputs (factors) and consider additive functions g over them (akin to additive decoders),

$$g(z) = \sum_{i=1}^{d_z} g_i(z_i)$$

Lets denote the training support of input factors by $\mathcal{Z}^{\text{train}}$, and define the cartesian-product extension of training support as $\text{CPE}(\mathcal{Z}^{\text{train}}) \coloneqq \mathcal{Z}_1^{\text{train}} \times \mathcal{Z}_2^{\text{train}} \times \cdots \mathcal{Z}_{d_z}^{\text{train}}$. Lets denote the model trained with the inductive bias of additivity as $\hat{g}(z) = \sum_{i=1}^{d_z} \hat{g}_i(z_i)$ Then our task of cartesian-product extrapolation is defined as follows.

Definition 4.2.1 (Cartesian-Product Extrapolation)**.** If the learned additive model ($\hat{g}$) and the true additive model (g) match on training distribution, $\hat{g}(z) = g(z)\ \forall z \in \mathcal{Z}^{\text{train}}$, then their predictions also match on the cartesian-product of the training support, $\hat{g}(z) = g(z)\ \forall z \in \text{CPE}(\mathcal{Z}^{\text{train}})$.

Further, we assume $\mathcal{Z}^{\text{test}} \subseteq \text{CPE}(\mathcal{Z}^{\text{train}})$, therefore $\hat{g}(z) = g(z)\ \forall z \in \mathcal{Z}^{\text{test}}$.

Note that as per the above setup, we have that the support of any component $\mathcal{Z}_i$ is the same across the training distribution and its cartesian-product extension, summarized below.

$$\mathcal{Z}_i^{\text{train}} = \text{CPE}(\mathcal{Z}^{\text{train}})_i \ \ \forall i \in [1, \cdots, d_z] \tag{4.1}$$

Lets first analyze the case of continuous factors, where we can show the following.

$$\begin{aligned}
&\hat{g}(z) = g(z) \quad \forall z \in \mathcal{Z}^{\text{train}} \\
\implies &\sum_i \hat{g}_i(z_i) = \sum_i g_i(z_i) \quad \forall z \in \mathcal{Z}^{\text{train}} \\
\implies &\frac{\partial \hat{g}_i(z_i)}{\partial z_i} = \frac{\partial g_i(z_i)}{\partial z_i} \quad \forall z_i \in \mathcal{Z}_i^{\text{train}} \\
\implies &\hat{g}_i(z_i) = g_i(z_i) + C_i \quad \forall z_i \in \mathcal{Z}_i^{\text{train}} \quad \text{where} \sum_i C_i = 0.
\end{aligned}$$

Its important to note that the last equality follows under the assumption that the support of z is connected, otherwise C_i could be function of z (check Appendix B.2 for more details). Using equation (4.1), we have the following:

$$\hat{g}_i(z_i) = g_i(z_i) + C_i \quad \forall z_i \in \mathcal{Z}_i^{\text{train}} \iff \hat{g}_i(z_i) = g_i(z_i) + C_i \quad \forall z_i \in \text{CPE}(\mathcal{Z}^{\text{train}})_i$$

With the above equality, we can easily show extrapolation to novel points from CPE($\mathcal{Z}^{\text{train}}$).

$$\begin{aligned}\hat{g}(z) &= \sum_i \hat{g}_i(z_i) \quad \forall z \in \text{CPE}(\mathcal{Z}^{\text{train}}) \\ &= \sum_i g_i(z_i) + C_i \quad \forall z \in \text{CPE}(\mathcal{Z}^{\text{train}}) \\ &= g(z) \quad \forall z \in \mathcal{Z}^{\text{train}} \quad (\text{because} \sum_i C_i = 0)\end{aligned}$$

The challenge with discrete factors. Note that if the support of z was not (path) connected then C_i would be a function of z as well and we cannot achieve extrapolation. This precisely highlights the challenge with discrete factors as their support is not connected! In general, for a discrete latent variable we only have the following.

$$\hat{g}_i(z_i) = g_i(z_i) + C_i(z_i) \quad \text{s.t.} \quad \sum_i C_i(z_i) = 0 \quad \forall z_i \in \mathcal{Z}^{\text{train}} \tag{4.2}$$

The implication of this for novel points $z \in \text{CPE}(\mathcal{Z}^{\text{train}})$ is as follows:

$$\begin{aligned}\hat{g}(z) &= \sum_i \hat{g}_i(z_i) \quad \forall z \in \text{CPE}(\mathcal{Z}^{\text{train}}) \\ &= \sum_i g_i(z_i) + C_i(z_i) \quad \forall z \in \text{CPE}(\mathcal{Z}^{\text{train}}) \\ &= g(z) + \sum_i C_i(z_i) \quad \forall z \in \text{CPE}(\mathcal{Z}^{\text{train}})\end{aligned}$$

Hence, the model is wrong in its predictions by the offset $\sum_i C_i(z_i)$ which is not necessarily zero for novel factors.

4.2.1 Path Connected Support for Discrete Factors

We now develop the analogue of path connected support for discrete factors that would enable cartesian-product extrapolation. Lets construct the following graph based on the training support of discrete factors $\mathcal{Z}^{\text{train}}$.

Definition 4.2.2. Given $\mathcal{Z}^{\text{train}}$, construct an undirected graph $G = (V, E)$ where each factor is a node in the graph, $z \in V \;\; \forall \;\; z \in \mathcal{Z}^{\text{train}}$. Further, $(z^i, z^j) \in E$ if $H(z^i, z^j) = 1$, where H denotes the hamming distance. Hence, there exist an edge between $z^i, z^j \in V$ if z^i and z^j only differ in exactly one component.

To understand the importance of above graph, consider two factors (z^1, z^2) that have an edge between them, $(z^1, z^2) \in E$. Without loss of generality assume they only differ at the k^{th} component, hence, $C_i(z_i^1) = C_i(z_i^2)\ \ \forall i \neq k$. Following (4.2), we have $\sum_i C_i(z_i^1) = \sum_i C_i(z_i^2) = 0$, which we simplify as follows.

$$\begin{aligned}
&\sum_i C_i(z_i^1) = \sum_i C_i(z_i^2) \\
\implies &\sum_{i \neq k} C_i(z_i^1) + C_k(z_k^1) = \sum_{i \neq k} C_i(z_i^2) + C_k(z_k^2) \\
\implies &C_k(z_k^1) = C_k(z_k^2)
\end{aligned}$$

Therefore, if two factors z^1, z^2 are connected by an edge then all the offsets C_i have the same value for them. We can extend this argument via induction to have the following claim.

Proposition 4.2.3. *If two factors $z^i, z^j \in \mathcal{Z}^{\text{train}}$ are connected via a path in the graph G, then we have $C_k(z_k^i) = C_k(z_k^j)\ \ \forall k \in d_z$.*

A corollary from this is we assume that all factors $z \in \mathcal{Z}^{\text{train}}$ are connected via path in G, then offsets C_k are constant over the entire support, akin to the case of continuous factors with path-connected support! Since offsets don't vary across the factors and we know $\sum_i C_i = 0$, therefore we generalize to novel points as well.

Corollary 4.2.4. *Assume $\mathcal{Z}^{\text{train}}$ is "path-connected", i.e, all factors are connected to each other via a path in the graph G. Then we achieve cartesian-product extrapolation (Definition 4.2.1) with additive functions.*

4.2.2 Affine Hull Extrapolation

We now discuss an alternate characterization of extrapolation with discrete factors that relates novel factors as an affine combination of observed factors in the training data. The motivation for this alternative characterization is that it would help us avoid computing the partition function with energy based models (which is a challenging task) while allowing extrapolation to the novel points. We understand this might not be very clear as of now, so the reader may interpret this as another way to analyze discrete extrapolation.

Consider an example of 2 dimensional binary factors, $z = (z_1, z_2)$ where $z_i \in \{0, 1\}$, such that $\{(0,0), (0,1), (1,0)\} \in \mathcal{Z}^{\text{train}}$. Then for the novel factor $z = (1,1)$, we have $\hat{g}(1,1) = \hat{g}(1,0) - \hat{g}(0,0) + \hat{g}(0,1)$. Therefore, if we estimate the function perfectly for factors in training dataset, then we can extrapolate as prediction for the novel factor is an affine combination of the prediction for training factors.

In other words, perfect estimation on the training support should imply extrapolation to the affine hull extension of the training support ($\mathsf{Aff}(\mathcal{Z}^{\text{train}})$). To formally define $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$, lets consider discrete factors where each component z_i can take m different possible values. Let $\tau(z_i) \in \mathbb{R}^m$ denote the 1-hot transformation of z_i, and denote their concatenation by $\tau(z) = [\tau(z_1), \cdots, \tau(z_{d_z})] \in \mathbb{R}^{d_z \times m}$. Now we can define the affine hull extension as follows:

Definition 4.2.5 (Affine Hull Extension of Training Support)**.** All factors $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ must satisfy the following constraint: $\tau(z') = \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z \tau(z)$ where $\sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z = 1$.

Hence, the one-hot concatenation vector for $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ can be expressed as an affine combination of one-hot concatenation vectors of training factors. Note that we can evaluate additive function g for point $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ as follows (proof in Appendix E.1):

$$\hat{g}(z') = \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z \hat{g}(z) \text{ where } \tau(z') = \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z \tau(z)$$

Hence, function evaluation for factors in $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ is an affine combination of the function evaluation for factors in training data. Therefore, we can state the following corollary, that additive functions can perform affine hull extrapolation.

Corollary 4.2.6. *If the learned additive model ($\hat{g}$) and the true additive model (g) match on training distribution, $\hat{g}(z) = g(z)\ \forall z \in \mathcal{Z}^{\text{train}}$, then their predictions also match on the affine hull extension of the training support, $\hat{g}(z) = g(z)$ for $z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.*

Relationship between $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$ and $\text{CPE}(\mathcal{Z}^{\text{train}})$. While these two extensions of the training support might look very different, we believe that $\mathsf{Aff}(\mathcal{Z}^{\text{train}}) = \text{CPE}(\mathcal{Z}^{\text{train}})$ though we don't have a formal proof yet. For example, consider the 2-d binary factor case (z_1, z_2) and assume $\mathcal{Z}^{\text{train}} = \{(0, 0, (0, 1), (1, 0))\}$. Then the novel factor $(1, 1) \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ as $\tau(1, 1) = \tau(1, 0) - \tau(0, 0) + \tau(0, 1)$, hence $\mathsf{Aff}(\mathcal{Z}^{\text{train}}) = \text{CPE}(\mathcal{Z}^{\text{train}})$.

Another way to analyze the relationship between them is to answer the following question; how many factors do we need to observe during training so that $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$ captures the full cartesian-product space, i.e., $\bigtimes_{i=1}^{d_z}[m]$ (assuming each component has m possible values)? For the case of 2-dimensional factors ($d_z = 2$), we do a probabilistic analysis where we start sampling factors uniformly at random (this can be relaxed to more general distributions) and show that we need $O(m \log m)$ factors for $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$ to contain the entire cartesian-product space with a high probability (proof in Appendix E.2).

Theorem 4.2.7. *Assume 2-d factors, i.e., $d_z = 2$. If the number of sampled factors is more than $8c * m \log(m)$, then $\mathsf{Aff}(\mathcal{Z}^{\text{train}}) = [m] \times [m]$ with probability $\geq 1 - \frac{1}{c}$.*

Hence, the above theorem states that we need $O(m \log m)$ factors during training to extrapolate to all the m^2 factors. The proof technique doesn't work for higher dimensional factors ($d_z > 2$), however, we have performed simulations that hint the general bound will be of the form $O(d_z * m \log m)$, which is remarkably small given the complete space of factors grows exponentially in d_z. We hope to show this bound theoretically as well in future work.

4.3 Extrapolation via Additive Energy Models

We now study the question of extrapolation with discrete factors for additive energy models. Additive energy models makes a mild yet realistic assumption on $p(x|z)$; they assume that the associated energy function $E(x, z)$ is additive in terms of factors z, i.e., $E(x, z) = \sum_{i=1}^{d_z} E_i(x, z_i)$. We use the following equivalent notation for additive energy $E(x, z)$ as $\langle 1, \boldsymbol{E}(x, z)\rangle$ where $\boldsymbol{E}(x, z) = [E_1(x, z_1), \cdots, E_{d_z}(x, z_{d_z})]$ as shown below.

$$p(x|z) = \frac{1}{B(z)}\Big[\exp\Big(- \langle 1, \boldsymbol{E}(x, z)\rangle \Big)\Big] \tag{4.3}$$

where $B(z)$ is the partition function $B(z) = \int \exp(-\langle 1, \boldsymbol{E}(x, z)\rangle)dx$.

Unlike additive functions in the previous section, additive energy models are more flexible as the observations are modeled in a probabilistic manner, along with interaction terms between latent factors modeled by $B(z)$. The simpler structure of additive functions allowed us to show extrapolation guarantees with them (Corollary 4.2.4, Corollary 4.2.6), and our goal now is to establish similar guarantees with additive energy models.

We first state an assumption that will be useful for all our proofs.

Assumption 4.3.1 (Invariant Support of Data)**.** The support of the distribution $p(x|z)$ is independent of z, i.e, the range of values allowed for x are the same given any factor z.

This is a strong but necessary assumption because we are seeking extrapolation to novel factors $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$, and if there is a shift in the support of $p(x|z')$, i.e., $\exists\ x'$ such that $p(x'|z) = 0\ \forall z \in \mathcal{Z}^{\text{train}}$, then intuitively we cannot constrain $p(x'|z')$ as a function of the training data. We now state our result for affine hull extrapolation with learned additive energy models $\hat{p}(x|z) = \frac{1}{\hat{B}(z)}[\exp(-\langle 1, \hat{\boldsymbol{E}}(x, z)\rangle)]$, with the proof in Appendix E.3.

Theorem 4.3.2. *[Affine Hull Extrapolation with Additive Energy Models] Under Assumption 4.3.1, if the learned additive energy model $\hat{p}(x|z)$ matches the true additive energy model $p(x|z)$ on training data, i.e., $\hat{p}(x|z) = p(x|z)\ \forall z \in \mathcal{Z}^{\text{train}}$, then we would have affine hull extrapolation, $\hat{p}(x|z) = p(x|z)\ \forall z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.*

4.3.1 Extrapolation for Classification Problems

Note that we did not explicitly discuss how to perform density estimation with energy based models $\hat{p}(x|z)$, but in general it is a hard task that involves computing the partition function $B(z) = \int \exp(-\langle 1, \boldsymbol{E}(x,z)\rangle)dx$. We have not worked further on methods and experiments to justify Theorem 4.3.2 for density estimation with additive energy models, but we do believe its an important future work. The current focus of the project is on extrapolation in classification (discriminative) tasks ($p(z|x)$) with additive energy models, where we provide novel techniques to bypass computation of the partition function.

Definition 4.3.3 (Compositional Classification Task)**.** We samples factors z from the prior distribution $p(z)$, and sample data from the conditional distribution $p(x|z)$ given by the additive energy model (4.3) that satisfies the invariant support assumption (4.3.1). The learner has access to samples (x, z) where $z \in \mathcal{Z}^{\text{train}}$, and the objective is to correctly model $p(z|x)$ for all $z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$. Further, we assume that the learner knows the true prior distribution $p(z)$.

Note that the above classification task can be solved using the route of generative (bayesian) classification, i.e, the learner models $\hat{p}(x|z)$ as an additive energy model and extrapolates it using Theorem 4.3.2. Then we can accurately estimate $p(z|x)$ over the affine-hull extension $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$ as $\hat{p}(z|x) = \text{Softmax}(\log \hat{p}(x|z) + \log p(z))$, which is dervied using bayes rule as follows:

$$\hat{p}(z|x) = \frac{\hat{p}(x|z)p(z)}{\hat{p}(x)} = \frac{\hat{p}(x|z)p(z)}{\int \hat{p}(x|z)p(z)dx} = \text{Softmax}(\log \hat{p}(x|z) + \log p(z))$$

We write this formally for convenience of the reader.

Corollary 4.3.4. *Given the data generation process in Definition 4.3.3, and the estimator* $\hat{p}(z|x) = \text{Softmax}(\log \hat{p}(x|z) + \log p(z))$ *where* $\hat{p}(x|z)$ *corresponds to learned additive energy model, i.e,* $\hat{p}(x|z) = \frac{1}{\hat{B}(z)}[\exp(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle)]$*. Then the learner* $\hat{p}(z|x)$ *can solve the compositional classification task, i.e,* $p(z|x) = \hat{p}(z|x)\ \forall z \in \mathcal{Z}^{\text{train}}$ *implies* $p(z|x) = \hat{p}(z|x)\ \forall z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$*.*

But as we mentioned before, estimating $\hat{p}(x|z)$ with additive energy models would require us to solve the difficult task of estimating partition function $B(z)$. Instead, we propose the following estimator for $\hat{p}(z|x)$, which is inspired from additive energy model but does not restrict the interaction term between factors to be a partition function.

$$\hat{p}(z|x) = \text{Softmax}\Big(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle - \log \hat{M}(z) + \log p(z)\Big) \tag{4.4}$$

where $\hat{M}(z)$ is a free parameter which is not related to energy $E(x, z)$ unlike partition functions. Note that even if $\hat{p}(z|x) = p(z|x)\ \forall z \in \mathcal{Z}^{\text{train}}$, we still won't extrapolate to $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$ as $\hat{M}(z)$ is free to choose any values for $z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})/\mathcal{Z}^{\text{train}}$, which wouldn't happen if $\hat{M}(z)$ was constrained to be the partition function (Corollary 4.3.4).

However, we propose the following procedure to extrapolate $\hat{M}(z)$ for novel factors $z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$. Define $\hat{Q}(z)$ as a function of $\hat{\boldsymbol{E}}(x, z)$ and $\hat{M}(z)$ as follows,

$$\hat{Q}(z) = \mathbb{E}_{x \sim p^{\text{train}}(x)} \left[\frac{\exp\Big(- \langle 1, \hat{\boldsymbol{E}}(x, z) \rangle \Big)}{\sum_{\tilde{z} \in \mathcal{Z}^{\text{train}}} \exp\Big(- \langle 1, \hat{\boldsymbol{E}}(x, \tilde{z}) \rangle - \log \hat{M}(\tilde{z}) + \log p(\tilde{z}) \Big)} \right]$$

The idea is to replace $\hat{M}(z)$ by $\hat{Q}(z)$ in the estimator (4.4) for extrapolating to novel points $z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$. More formally, define the estimator $\tilde{p}(z|x)$, which is a function of $\hat{\boldsymbol{E}}(x, z)$ and $\hat{Q}(z)$ as follows:

$$\tilde{p}(z|x) = \text{Softmax}\Big(- \langle 1, \hat{\boldsymbol{E}}(x, z) \rangle - \log \hat{Q}(z) + \log p(z) \Big) \tag{4.5}$$

Then we can solve compositional classification using the estimator $\tilde{p}(z|x)$.

Theorem 4.3.5. *Consider the data generation process in Definition 4.3.3 and the proposed estimator $\hat{p}(z|x)$ (4.4). If $\hat{p}(z|x) = p(z|x)\ \forall z \in \mathcal{Z}^{\text{train}}$, then we can solve the compositional classification task with $\tilde{p}(z|x)$ (4.5), i.e., $\tilde{p}(z|x) = p(z|x)\ \forall z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.*

The proof for the above theorem can be found in Appendix E.4. Hence, even though the estimator $\hat{p}(z|x)$ cannot solve the compositional classification task, its extension $\tilde{p}(z|x)$ can achieve the desired objective!

4.4 Experiments

We test our additive energy framework for group robust classification/subpopulation shifts [Yang et al., 2023], where the the factors (groups) z consist of a class label y and a spurious attribute a, i.e, $z = (y, a)$ where y and a are correlated. The goal is to predict the class labels y from observations x, under the assumption that the distribution of factors changes from training to test, $p^{\text{train}}(z) \neq p^{\text{test}}(z)$. For example, consider the Waterbirds dataset [Wah et al., 2011], where the class label y is binary (land bird vs water bird) and the attribute a is binary (land background vs water background) as well. In the training dataset, the land birds mostly occur on land background and the waterbirds mostly occur on water background. While this distribution changes at test time, hence the classifier using spurious correlation between y

and a would fail to generalize.

In our setup, we create an extreme version of sub-population shift problem where the learner does not have access to any samples from one of the four groups during training in the Waterbirds dataset, while all the groups are present at test time. This setup reflects the compositional classification task (Definition 4.3.3) with $\mathcal{Z}^{\text{test}} = \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.

4.4.1 Implementation of Proposed Approach

We describe the implementation of our approach for the problem of compositional population shifts with $z = (y, a)$. Our estimator $\hat{p}(z|x)$ (4.4) models each energy term $E_i(x, y)$ and $E(x, a)$ as a linear layer composed with representations of a pre-trained network $\phi(x)$, i.e, $E_i(x, y) =< W_y, \phi(x) >$ and $E_i(x, a) =< W_a, \phi(x) >$ where W_y and W_a are parameters of the model. The interaction terms $\hat{M}(y, a)$ are parameters initialized to one, and $p(z)$ assigns prior probability based on counts of each factor in the training dataset. We write this as equation below for the convenience of the reader.

$$\hat{p}(z|x) = \text{Softmax}\Big(- < W_y, \phi(x) > - < W_a, \phi(x) > - \log \hat{M}(z) + \log p(z)\Big)$$

To make prediction for class labels y, we marginalize the probabilities over the attributes a, $\hat{p}(y|x) = \sum_a \hat{p}(y, a|x)$. We train this model with the cross-entropy objective where the parameters of representation network $\phi(x)$ are frozen.

$$\min_{W_y, W_a, \hat{M}} \mathbb{E}_{x,(y,a)} - [y \log \hat{p}(y|x)]$$

After training the parameters, we switch to the estimator $\tilde{p}(z|x)$ (4.5) for inference at test distribution. Since we have no knowledge about the $p^{\text{test}}(z)$ we assign equal probabilties to all the factors, which essentially has not effect on the Softmax, hence we drop that term. Therefore, the final predictor for test time is $\hat{y} = \arg\min_y \sum_a \tilde{p}(y, a|x)$, where

$$\tilde{p}(y, a|x) = \text{Softmax}\Big(- < W_y, \phi(x) > - < W_a, \phi(x) > - \log \hat{Q}(z)\Big)$$

4.4.2 Setup

We experiment on the widely used benchnmarks for subpopulation shifts; Waterbirds [Wah et al., 2011], CelebA [Liu et al., 2015], and MetaShift [Liang and Zou, 2022]. All these datasets have binary class label y and binary attribute a, therefore we have a total of 4 factors (groups). To adapt these benchmarks for compositional shifts, we drop samples one of

Removed (y, a)	Method	Average Acc	Worst Group Acc
(0, 0)	ERM	0.76 (0.0)	0.69 (0.0)
(0, 0)	GroupDRO	0.86 (0.0)	0.78 (0.0)
(0, 0)	AddEnergy	0.88 (0.0)	0.86 (0.0)
(0, 1)	ERM	0.71 (0.0)	0.38 (0.0)
(0, 1)	GroupDRO	0.79 (0.01)	0.41 (0.05)
(0, 1)	AddEnergy	0.87 (0.0)	0.79 (0.01)
(1, 0)	ERM	0.81 (0.01)	0.1 (0.02)
(1, 0)	GroupDRO	0.92 (0.0)	0.74 (0.04)
(1, 0)	AddEnergy	0.88 (0.0)	0.85 (0.0)
(1, 1)	ERM	0.89 (0.0)	0.53 (0.0)
(1, 1)	GroupDRO	0.91 (0.0)	0.77 (0.04)
(1, 1)	AddEnergy	0.89 (0.0)	0.86 (0.0)

Table 4.1: Results for compositional generalization on the Waterbirds benchmark. The first column describes the factors that were dropped during training. The performance for both the metrics is denoted as mean ± standard error over 3 random seeds on the test dataset.

these groups entirely at training, while all the groups are present at test time. More details about each datasets are provided in Appendix E.5.1.

For baselines, we use the naive ERM estimator, and GroupDRO [Sagawa et al., 2019] which is a popular approach for tackling subpopulation shifts. For the baselines as well as our approach we use pretrained ResNet-50 architecture as the repersentaiton network $\phi(x)$ and do not finetune it further. The learnable parameters are linear layers on top of these representations.

For metrics, we report the average accuracy and the worst-group accuracy on the test dataset. Due to imbalances in group distribution, a method can obtain good average accuracy despite having bad worst-group accuracy. Hence, the worst-group accuracy is more indicative of robustness to spurious correlations.

4.4.3 Preliminary Results

Table 4.1 presents the results for the Waterbirds benchmark, with results for CelebA and MetaShit in Appendix E.5.2. We have 4 different settings created by dropping one of the four factors during training. The proposed approach (AddEnergy) obtains significantly better peformance than ERM and GroupDRO in terms of worst group accuracy in all the scenarios. Further, the performance gains with AddEnergy over baselines is much higher for the scenario where we dropped the groups $(0, 1)$ and $(1, 0)$, which were the minority groups in the original training distribution (Table E.1), as in these cases the strength of spurious correlations increases, thus making it difficult for the baselines to generalize. Hence, our empirical results provide evidence that additive energy assumption is realistic for modeling high-dimensional images, and the proposed estimator (4.4) is able to extrapolate to novel combinations from $\mathsf{Aff}(\mathcal{Z}^{\text{train}})$ without estimating the partition function.

Planned Experiments. Our plan is to use more complex subpopulation shift benchmarks like CivilComments [Borkan et al., 2019], MultiNLI [Williams et al., 2017], and NICO++ [Zhang et al., 2023] where we both the class label y and spurious attribute a are non-binary and the total number of factors are large, around 360 for NICO++.

Further, we plan to carry experiments to verify Theorem 4.2.7 on with synthetic datasets like Colored-MNIST with factors as (class label, color); where we can systematically test the influence of increasing the number of factors observed during training, and whether we are able to empirically verify the $O(m \log m)$ bound. Once this is verified on ColoredMNIST, we plan to test this for more complex datasets like ImageNet-Background [Xiao et al., 2020].

4.5 Future Directions

Compositional generalization with unknown factors. The main assumption throughout our analysis with additive energy based model is that we observe the factors z responsible for generating the data x. The next step is to solve the compositional generalization task (Definition 4.3.3) without having access to the labeled factors z. However, before tackling this challenging case directly, we aim to simplify the problem with the assumption that we know the relevant class variables y but the other attributes are unknown, where $z = (y, a_1, \cdots, a_{d_z-1})$. This bears similarity to disentanglement with weak supervision as observing y should help us identify the latent z from observations x. This setup has also been the focus of recent works by Pezeshki et al. [2023] and Tsirigotis et al. [2024], where they aim to solve subpopulation shifts with access to class labels y but no access to the spurious attributes. While these works have shown good performance for inferring factors, they have not been designed for the compositional shift task that is the focus of our work. These approaches lack specific inductive biases that would favor them to discover composable latent attributes from observations. Also, they do not provide any identification guarantees, which can be the focus of our work. Hence, creating an approach to discover composable latent factors given observations x and class labels y in an identifiable manner is an open problem, and it serves as the ideal next step to extend our framework for compositional generalization with additive energy models.

Compositional Generative Modeling. While we proposed theoretical results (Theorem 4.3.2) for extrapolation with additive energy models for generative tasks, we have not empirically verified it yet. The current focus of the project was limited to compositional classification task, hence we did not make progress in methodology for learning additive energy models for generative tasks. This is another important problem for us to work on, where the focus is on compositional generation and developing efficient strategies to train additive energy models.

As a first step towards this goal, we would experiment on datasets like moving balls, etc. used in prior works [Wiedemer et al., 2024] and show that additive energy models can be learned from data with occlusions, and generate novel composition of these objects. For this task we can assume that we know the latent factors (e.g. position coordinates) of these objects and focus on solving compositional generation, akin to the experiments done by Wiedemer et al. [2024]. Once we are able to make good progress on this task, we can move to the next step of compositional generation without observing the latent factors. Currently, we do not have theoretical evidence for identification of latent factors using additive energy based decoders, but perhaps we can establish results similar to block identifibility with additive decoders. The reason for this is additive energy model bears similarity to additive decoders if we consider the score function $\nabla_x \log p(x|z) = - < 1, \nabla_x \boldsymbol{E}(x, z) >$, and assuming perfect estimation of score function would imply the following:

$$< 1, \nabla_x \hat{\boldsymbol{E}}(x, \hat{z}) >=< 1, \nabla_x \boldsymbol{E}(x, z) >$$

This is similar to the identity we obtained as a result of perfect optimization of reconstruction objective with additive decoders, so the same proof techniques could be helpful.

Further, there are a series of interesting works [Du et al., 2021, Liu et al., 2023, Su et al., 2024] that provide empirical evidence for disentanglement with architectures that resemble additive energy model. Hence, we believe that under the right assumptions it must be possible to identify latent factors with additive energy models.

List of Contributions

- Evaluating Interventional Reasoning Capabilities of Large Language Models. [**Preprint**]
  Tejas Kasetty, Divyat Mahajan, Gintare Karolina Dziugaite, Alexandre Drouin, Dhanya Sridhar

- Empirical Analysis of Model Selection for Heterogeneous Causal Effect Estimation.
  Divyat Mahajan, Ioannis Mitliagkas, Brady Neal, Vasilis Syrgkanis
  International Conference on Learning Representations [**ICLR 2024 (Spotlight)**]

- Additive decoders for latent variables identification and cartesian-product extrapolation.
  Sébastien Lachapelle, Divyat Mahajan*, Ioannis Mitliagkas, Simon Lacoste-Julien*
  Advances in Neural Information Processing Systems [**NeurIPS 2023 (Oral)**]

- Interventional causal representation learning.
  Kartik Ahuja, Divyat Mahajan, Yixin Wang, Yoshua Bengio
  International Conference on Machine Learning [**ICML 2023 (Oral)**]

- Synergies between disentanglement and sparsity: Generalization and identifiability in multi-task learning.
  Sébastien Lachapelle, Tristan Deleu, Divyat Mahajan, Ioannis Mitliagkas, Yoshua Bengio, Simon Lacoste-Julien, Quentin Bertrand
  International Conference on Machine Learning [**ICML 2023**]

- Towards efficient representation identification in supervised learning.
  Kartik Ahuja, Divyat Mahajan*, Vasilis Syrgkanis, Ioannis Mitliagkas.*
  Conference on Causal Learning and Reasoning [**CLeaR 2022**]

Timeline

Fall 2024 Trimester. I plan to finish the ongoing project on compositional generalization with additive energy models Chapter 4 and submit this ICLR 2025. There is another project that I am leading on amortized learning of structural causal models that I didn't describe in this report. This project started in my internship with the causal machine learning group at MSR Cambridge, and I am planning to submit that to ICLR 2025 as well.

Winter 2025, Summer 2025, Fall 2025 Trimesters. I plan to work on the future problems described in Section 4.5; the project on compositional generalization with only class labels known, and compositional generative models. There is a collaboration that I will being soon with researchers at Service Now on causality with large language models; I couldn't go in details about that in the report. I plan to make progress on this project during 2025 as well.

Winter 2026, Summer 2026. I plan to wrap any existing projects and submit my thesis by the end of Summer 2026.

References

K. Ahuja, J. Hartford, and Y. Bengio. Weakly supervised representation learning with sparse perturbations, 2022.

K. Ahuja, D. Mahajan, Y. Wang, and Y. Bengio. Interventional causal representation learning. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 2013.

Y. Bengio, T. Deleu, N. Rahaman, R. Ke, S. Lachapelle, O. Bilaniuk, A. Goyal, and C. Pal. A meta-transfer objective for learning to disentangle causal mechanisms. *arXiv preprint arXiv:1901.10912*, 2019.

M. Besserve, R. Sun, D. Janzing, and B. Schölkopf. A theory of independent mechanisms for extrapolation in generative models. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

D. Borkan, L. Dixon, J. Sorensen, N. Thain, and L. Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pages 491–500, 2019.

J. Brehmer, P. De Haan, P. Lippe, and T. Cohen. Weakly supervised causal representation learning. In *Advances in Neural Information Processing Systems*, 2022.

P. Brouillard, S. Lachapelle, A. Lacoste, S. Lacoste-Julien, and A. Drouin. Differentiable causal discovery from interventional data. *arXiv preprint arXiv:2007.01754*, 2020.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

S. Buchholz, M. Besserve, and B. Schölkopf. Function classes for identifiable nonlinear independent component analysis. In *Advances in Neural Information Processing Systems*, 2022.

S. Buchholz, G. Rajendran, E. Rosenfeld, B. Aragam, B. Schölkopf, and P. Ravikumar. Learning linear causal representations from interventions under general nonlinear mixing. *Advances in Neural Information Processing Systems*, 36, 2024.

C. P. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner. Monet: Unsupervised scene decomposition and representation, 2019.

R. Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

M. Chickering, D. Heckerman, and C. Meek. Large-sample learning of bayesian networks is np-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004.

P. Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

E. Crawford and J. Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

D. F. Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016.

G. Darmois. General analysis of stochastic bonds: special study of linear factorial analysis. *Journal of the International Statistical Institute*, pages 2–8, 1953.

T. Deleu, A. Góis, C. Emezue, M. Rankawat, S. Lacoste-Julien, S. Bauer, and Y. Bengio. Bayesian structure learning with generative flow networks. *arXiv preprint arXiv:2202.13903*, 2022.

T. Deleu, M. Nishikawa-Toomey, J. Subramanian, N. Malkin, L. Charlin, and Y. Bengio. Joint bayesian inference of graphical structure and parameters with a single generative flow network. *Advances in Neural Information Processing Systems*, 36, 2024.

A. Dittadi, S. Papa, M. De Vita, B. Schölkopf, O. Winther, and F. Locatello. Generalization and robustness implications in object-centric learning. *arXiv preprint arXiv:2107.00637*, 2021.

A. Dixit, O. Parnas, B. Li, J. Chen, C. P. Fulco, L. Jerby-Arnon, N. D. Marjanovic, D. Dionne, T. Burks, R. Raychowdhury, et al. Perturb-seq: dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *cell*, 167(7):1853–1866, 2016.

Y. Du and I. Mordatch. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems*, 2019.

Y. Du, S. Li, Y. Sharma, J. Tenenbaum, and I. Mordatch. Unsupervised learning of compositional energy concepts. *Advances in Neural Information Processing Systems*, 34: 15608–15620, 2021.

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

M. Engelcke, A. R. Kosiorek, O. P. Jones, and I. Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *International Conference on Learning Representations*, 2020.

S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, K. Kavukcuoglu, and G. E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems*, 2016.

W. Feller. *An introduction to probability theory and its applications, vol 2.* John Wiley & Sons, 2008.

J. C. Foster, J. M. Taylor, and S. J. Ruberg. Subgroup identification from randomized clinical trial data. *Statistics in medicine*, 30(24):2867–2880, 2011.

M. Fumero, F. Wenzel, L. Zancato, A. Achille, E. Rodolà, S. Soatto, B. Schölkopf, and F. Locatello. Leveraging sparse and shared feature activations for disentangled representation learning. *Advances in Neural Information Processing Systems*, 36, 2024.

R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11): 665–673, 2020.

K. Greff, A. Rasmus, M. Berglund, T. Hao, H. Valpola, and J. Schmidhuber. Tagger: Deep unsupervised perceptual grouping. In *Advances in Neural Information Processing Systems*, 2016.

K. Greff, S. van Steenkiste, and J. Schmidhuber. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, 2017.

K. Greff, R. L. Kaufman, R. Kabra, N. Watters, C. Burgess, D. Zoran, L. Matthey, M. Botvinick, and A. Lerchner. Multi-object representation learning with iterative variational inference. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

L. Gresele, J. von Kügelgen, V. Stimper, B. Schölkopf, and M. Besserve. Independent mechanism analysis, a new concept? *arXiv preprint arXiv:2106.05200*, 2021.

H. Hälvä and A. Hyvarinen. Hidden markov nonlinear ica: Unsupervised learning from nonstationary time series. In *Conference on Uncertainty in Artificial Intelligence*, pages 939–948. PMLR, 2020.

A. Hauser and P. Bühlmann. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *The Journal of Machine Learning Research*, 13(1):2409–2464, 2012.

P. O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, B. Schölkopf, et al. Nonlinear causal discovery with additive noise models. In *NIPS*, volume 21, pages 689–696. Citeseer, 2008.

C.-Y. Hsieh, J. Zhang, Z. Ma, A. Kembhavi, and R. Krishna. Sugarcrepe: Fixing hackable benchmarks for vision-language compositionality. *Advances in neural information processing systems*, 36, 2024.

B. Huang, K. Zhang, Y. Lin, B. Schölkopf, and C. Glymour. Generalized score functions for causal discovery. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1551–1560, 2018.

A. Hyvarinen and H. Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in Neural Information Processing Systems*, 29:3765–3773, 2016.

A. Hyvarinen and H. Morioka. Nonlinear ica of temporally dependent stationary sources. In *Artificial Intelligence and Statistics*, pages 460–469. PMLR, 2017.

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

A. Hyvärinen and P. Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural networks*, 12(3):429–439, 1999.

A. Hyvarinen, H. Sasaki, and R. Turner. Nonlinear ica using auxiliary variables and generalized contrastive learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 859–868. PMLR, 2019.

A. Hyvärinen and H. Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. In *Advances in Neural Information Processing Systems*, 2016.

A. Hyvärinen and H. Morioka. Nonlinear ICA of Temporally Dependent Stationary Sources. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

A. Hyvärinen, H. Sasaki, and R. E. Turner. Nonlinear ica using auxiliary variables and generalized contrastive learning. In *AISTATS*. PMLR, 2019.

M. Ibrahim, Q. Garrido, A. Morcos, and D. Bouchacourt. The robustness limits of sota vision models to natural variation. *arXiv preprint arXiv:2210.13604*, 2022.

Y. Jiang and B. Aragam. Learning nonparametric latent causal graphs with unknown interventions, 2023.

N. R. Ke, S.-J. Dunn, J. Bornschein, S. Chiappa, M. Rey, J.-B. Lespiau, A. Cassirer, J. Wang, T. Weber, D. Barrett, M. Botvinick, A. Goyal, M. Mozer, and D. Rezende. Discogen: Learning to discover gene regulatory networks, 2023.

I. Khemakhem, D. Kingma, R. Monti, and A. Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR, 2020a.

I. Khemakhem, D. Kingma, R. Monti, and A. Hyvärinen. Variational autoencoders and nonlinear ica: A unifying framework. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 2020b.

I. Khemakhem, R. Monti, D. Kingma, and A. Hyvärinen. Ice-beem: Identifiable conditional energy-based deep models based on nonlinear ica. In *Advances in Neural Information Processing Systems*, 2020c.

I. Khemakhem, R. P. Monti, D. P. Kingma, and A. Hyvärinen. Ice-beem: Identifiable conditional energy-based deep models based on nonlinear ica. *arXiv preprint arXiv:2002.11537*, 2020d.

M. Kocaoglu, C. Snyder, A. G. Dimakis, and S. Vishwanath. CausalGAN: Learning causal implicit generative models with adversarial training. In *International Conference on Learning Representations*, 2018.

D. Krueger, E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, D. Zhang, R. Le Priol, and A. Courville. Out-of-distribution generalization via risk extrapolation (rex). In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

S. Lachapelle and S. Lacoste-Julien. Partial disentanglement via mechanism sparsity. In *UAI 2022 Workshop on Causal Representation Learning*, 2022.

S. Lachapelle, P. Brouillard, T. Deleu, and S. Lacoste-Julien. Gradient-based neural dag learning. *arXiv preprint arXiv:1906.02226*, 2019.

S. Lachapelle, T. Deleu, D. Mahajan, I. Mitliagkas, Y. Bengio, S. Lacoste-Julien, and Q. Bertrand. Synergies between disentanglement and sparsity: a multi-task learning perspective, 2022a.

S. Lachapelle, P. Rodriguez Lopez, Y. Sharma, K. E. Everett, R. Le Priol, A. Lacoste, and S. Lacoste-Julien. Disentanglement via mechanism sparsity regularization: A new principle for nonlinear ICA. In *First Conference on Causal Learning and Reasoning*, 2022b.

F. Leeb, G. Lanzillotta, Y. Annadani, M. Besserve, S. Bauer, and B. Schölkopf. Structure by architecture: Disentangled representations without regularization, 2021.

W. Liang and J. Zou. Metashift: A dataset of datasets for evaluating contextual distribution shifts and training conflicts. *arXiv preprint arXiv:2202.06523*, 2022.

Z. Lin, Y. Wu, S. V. Peri, W. Sun, G. Singh, F. Deng, J. Jiang, and S. Ahn. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. In *International Conference on Learning Representations*, 2020.

P. Lippe, T. Cohen, and E. Gavves. Efficient neural causal discovery without acyclicity constraints. *arXiv preprint arXiv:2107.10483*, 2021.

P. Lippe, S. Magliacane, S. Löwe, Y. M. Asano, T. Cohen, and E. Gavves. iCITRIS: Causal representation learning for instantaneous temporal effects. In *UAI 2022 Workshop on Causal Representation Learning*, 2022a.

P. Lippe, S. Magliacane, S. Löwe, Y. M. Asano, T. Cohen, and E. Gavves. CITRIS: Causal identifiability from temporal intervened sequences, 2022b.

N. Liu, Y. Du, S. Li, J. B. Tenenbaum, and A. Torralba. Unsupervised compositional concepts discovery with text-to-image generative models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2095, 2023.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

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 *international conference on machine learning*, pages 4114–4124. PMLR, 2019.

F. Locatello, B. Poole, G. Rätsch, B. Schölkopf, O. Bachem, and M. Tschannen. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pages 6348–6359. PMLR, 2020a.

F. Locatello, M. Tschannen, S. Bauer, G. Rätsch, B. Schölkopf, and O. Bachem. Disentangling factors of variations using few labels. In *International Conference on Learning Representations*, 2020b.

F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020c.

A. Mansouri, J. Hartford, K. Ahuja, and Y. Bengio. Object-centric causal representation learning. In *NeurIPS 2022 Workshop on Symmetry and Geometry in Neural Representations*, 2022.

B. Mityagin. The zero set of a real analytic function. *arXiv preprint arXiv:1512.07276*, 2015.

G. E. Moran, D. Sridhar, Y. Wang, and D. Blei. Identifiable deep generative models via sparse decoding. *Transactions on Machine Learning Research*, 2022.

J. Munkres. *Analysis On Manifolds*. Basic Books, 1991.

J. R. Munkres. *Topology*. Prentice Hall, Inc., 2 edition, 2000.

N. Pawlowski, D. Coelho de Castro, and B. Glocker. Deep structural causal models for tractable counterfactual inference. *Advances in neural information processing systems*, 33: 857–869, 2020.

J. Pearl. *Causality*. Cambridge university press, 2009.

J. Peters and P. Bühlmann. Identifiability of gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, 2014.

J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 2014.

J. Peters, D. Janzing, and B. Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

M. Pezeshki, D. Bouchacourt, M. Ibrahim, N. Ballas, P. Vincent, and D. Lopez-Paz. Discovering environments with xrm. *arXiv preprint arXiv:2309.16748*, 2023.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

C. Rodriguez. Introduction to functional analysis. Cambridge MA, 2021. URL `https://ocw.mit.edu/courses/18-102-introduction-to-functional-analysis-spring-2021/`. MIT OpenCourseWare.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

K. Roth, M. Ibrahim, Z. Akata, P. Vincent, and D. Bouchacourt. Disentanglement of correlated factors via hausdorff factorized support. In *The Eleventh International Conference on Learning Representations*, 2023.

D. B. Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100(469):322–331, 2005.

K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

B. Schölkopf. Causality for machine learning. *arXiv preprint arXiv:1911.10500*, 2019.

B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij. On causal and anticausal learning. *arXiv preprint arXiv:1206.6471*, 2012.

B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, and Y. Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

P. Spirtes, C. N. Glymour, R. Scheines, and D. Heckerman. *Causation, prediction, and search.* MIT press, 2000.

C. Squires, A. Seigal, S. Bhate, and C. Uhler. Linear causal disentanglement via interventions. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

J. Su, N. Liu, Y. Wang, J. B. Tenenbaum, and Y. Du. Compositional image decomposition with diffusion models. *arXiv preprint arXiv:2406.19298*, 2024.

X. Sun, D. Janzing, B. Schölkopf, and K. Fukumizu. A kernel-based causal learning algorithm. In *Proceedings of the 24th international conference on Machine learning*, pages 855–862, 2007.

A. Taleb and C. Jutten. Source separation in post-nonlinear mixtures. *IEEE Transactions on Signal Processing*, 1999.

F. J. Theis. Uniqueness of complex and multidimensional independent component analysis. *Signal Processing*, 84(5):951–956, 2004.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65:31–78, 2006.

C. Tsirigotis, J. Monteiro, P. Rodriguez, D. Vazquez, and A. C. Courville. Group robust classification without any group information. *Advances in Neural Information Processing Systems*, 36, 2024.

V. Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.

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. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

J. von Kügelgen, M. Besserve, L. Wendong, L. Gresele, A. Kekić, E. Bareinboim, D. Blei, and B. Schölkopf. Nonparametric identifiability of causal representations from unknown interventions. *Advances in Neural Information Processing Systems*, 36, 2024.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Y. Wang and M. I. Jordan. Desiderata for representation learning: A causal perspective. *arXiv preprint arXiv:2109.03795*, 2021.

Y. Wang and M. I. Jordan. Desiderata for representation learning: A causal perspective, 2022.

Z. Wang, L. Gui, J. Negrea, and V. Veitch. Concept algebra for text-controlled vision models, 2023.

T. W. Webb, Z. Dulberg, S. M. Frankland, A. A. Petrov, R. C. O'Reilly, and J. D. Cohen. Learning representations that support extrapolation. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

T. Wiedemer, J. Brady, A. Panfilov, A. Juhos, M. Bethge, and W. Brendel. Provable compositional generalization for object-centric learning. *arXiv preprint arXiv:2310.05327*, 2023.

T. Wiedemer, P. Mayilvahanan, M. Bethge, and W. Brendel. Compositional generalization from first principles. *Advances in Neural Information Processing Systems*, 36, 2024.

A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

K. Xiao, L. Engstrom, A. Ilyas, and A. Madry. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994*, 2020.

Y. Xie, J. E. Brand, and B. Jann. Estimating heterogeneous treatment effects with observational data. *Sociological methodology*, 42(1):314–347, 2012.

Y. Yang, H. Zhang, D. Katabi, and M. Ghassemi. Change is hard: A closer look at subpopulation shift. *arXiv preprint arXiv:2302.12254*, 2023.

W. Yao, G. Chen, and K. Zhang. Learning latent causal dynamics. *arXiv preprint arXiv:2202.04828*, 2022a.

W. Yao, Y. Sun, A. Ho, C. Sun, and K. Zhang. Learning temporally causal latent processes from general temporal data. In *International Conference on Learning Representations*, 2022b.

J. Zhang, K. Greenewald, C. Squires, A. Srivastava, K. Shanmugam, and C. Uhler. Identifiability guarantees for causal disentanglement from soft interventions. *Advances in Neural Information Processing Systems*, 36, 2024.

K. Zhang, J. Peters, D. Janzing, and B. Schölkopf. Kernel-based conditional independence test and application in causal discovery. *arXiv preprint arXiv:1202.3775*, 2012.

X. Zhang, Y. He, R. Xu, H. Yu, Z. Shen, and P. Cui. Nico++: Towards better benchmarking for domain generalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16036–16047, 2023.

Y. Zhang and Q. Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.

X. Zheng, B. Aragam, P. K. Ravikumar, and E. P. Xing. Dags with no tears: Continuous optimization for structure learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/e347c51419ffb23ca3fd5050202f9c3d-Paper.pdf.

Y. Zheng, I. Ng, and K. Zhang. On the identifiability of nonlinear ICA: Sparsity and beyond. In *Advances in Neural Information Processing Systems*, 2022.

R. S. Zimmermann, Y. Sharma, S. Schneider, M. Bethge, and W. Brendel. Contrastive learning inverts the data generating process. *arXiv preprint arXiv:2102.08850*, 2021.

Appendix A

Supplementary Material: Mathematical Preliminaries

A.1 Measure Theory

This section summarizes main concepts on measure theory from the lecture notes of the MIT OCW course on functional analysis [Rodriguez, 2021], specifically the lectures 6-9 that present measure theory.

The motivation behind measure theory is to understand how we can assign a notion of length to arbitrary subsets of real numbers. Let's start the discussion by considering intervals, for which we can easily define length. An interval $I \subset \mathbb{R}$ is defined by two endpoints (l, r) such that $l \leq r$ and it contains all the real numbers x such that $l < x < r$. If $l, r \in I$ then it a **closed interval**, else we refer it as an **open interval**. Given any interval $I \subset R$, we can define its length as $l(I) = r - l$. But how we assign notion of length to any arbitrary subset of $\mathbb{R}$? We define outer measure as a first step towards this problem.

Definition A.1.1 (Outer Measure)**.** For any subset $A \subseteq \mathbb{R}$, we define the outer measure $m^*(A) : P(\mathbb{R}) \rightarrow [0, \infty]$ as follows:

$$m^*(A) = inf \left\{ \sum_n l(I_n) \mid \{I_n\} \text{ is a countable collection of open intervals s.t. } A \subseteq \cup_n I_n \right\}$$

where $l(I)$ denotes the length of an interval.

The definitions states that if we measure the length of all possible coverings of a set A using union of open intervals $\{I_n\}$, then outer measure corresponds to the greatest lower bound (infimum) on the length of these coverings. Hence, outer measure provide us with a notion to "measure" the set A. In fact, if A is an interval, then we have the outer measure is the same as the length, $m^*(A) = l(A)$.

Sets with zero outer measure. For an empty set $A = \phi$, the outer measure will be zero, $m^*(A) = 0$. To prove this, we consider an interval $I = (-\epsilon, \epsilon)$ that contains A. Therefore, by the definition of outer measure we have $m^*(A) \leq l(I) = 2\epsilon$. As $\epsilon \to 0$, we obtain $0 \leq m^*(A) \leq l(I) = 0$, which implies $m^*(A) = 0$.

Further, we can show that the outer measure of a set A containing a single point ($A = \{a\}$) will be zero as well. To prove it, consider the open interval $I = (a - \epsilon, a + \epsilon)$ and follow the same proof technique as above. Also, this can be extended to show that all countable sets A have zero outer measure.

Monotonicity of outer measure. Consider any two sets A, B such that $A \subseteq B$, then we have $m^*(A) \leq m^*(B)$. Intuitively, for any interval that I that contains B, we must have that I contains A as well; this can be used to prove $m^*(A) \leq m^*(B)$.

Countable Sub-Additivity of outer measure. Consider a countable collection of pairwise disjoint sets $\{\cup_n A_n\}$, then for outer measure m^* we have the following: $m^*(\cup_n A_n) \leq \sum_n m^*(A_n)$. The proof for this claim is provided in the lecture notes 6 (Theorem 62) of the functional analysis course [Rodriguez, 2021].

Towards Lebesgue measure. Given outer measure satisfies countable sub-additivity ($m^*(\cup_n A_n) \leq \sum_n m^*(A_n)$), the next question to tackle is when does the outer measure achieve countable additivity ($m^*(\cup_n A_n) = \sum_n m^*(A_n)$)? Note that countable additivity is a desirable property for measuring sets, as it implies that the measure of a countable union of pairwise disjoint sets is equal to the sum of the measure of each set; therefore we can measure each individual set and use them to measure the set constructued by their union.

Infact, if we restrict the outer measure to a particular class of sets, termed as lebesque measurable sets, then outer measure would satisfy countable addivity on them!

Definition A.1.2 (Lebesgue Measurable Sets)**.** A set $E \subseteq \mathbb{R}$ is lebesgue measurable if $\forall A \subseteq \mathbb{R}$ we have

$$m^*(A) = m^*(A \cap E) + m^*(A \cap E^c)$$

Definition A.1.3 (Lebesgue Measure)**.** The restriction of outer measure (m^*) to lebesgue measurable sets is called the lebesgue measure (m).

Note that lebesgue measure is not defined for all subsets of $\mathbb{R}$, while the outer measure is defined on all subsets of $\mathbb{R}$. This restriction to lebesgue measurable sets allow the important property of countable additivity. Therefore, with lebesgue measure we retain all the properties we proved earlier for outer measure, along with the countable additivity. The key properties of the lebesgue measure can summarized below.

1. By definition, lebesgue measure is non-negative, $m(A) \geq 0$.
2. Lebesgue measure of an empty set is zero, $m(\{\phi\}) = 0$.
3. Countable Additivity: For countable collection of disjoint sets $\{A_n\}$, $m(\cup_n A_n) = \sum_n m(A_n)$.
4. Lebesgue measure of an interval (I) is equal to its length, $m(I) = l(I)$.

Lets understand the structure of lebesgue measurable sets (Def. A.1.2) in more detail. Note that set empty $E = \{\phi\}$ is lebesgue measurable; since we have $A \cap E = \phi$ and $A \cap E^c = A$, and we know that $m^*(\{\phi\}) = 0$. Further, by definition if E is lebesgue measurable then E^c is also lebesgue measurable. It can also be proved that if a countable collection of sets $\{E_n\}$ is lebesgue measurable, then their union $(\cup_n E_n)$ is also lebesgue measurable.

To summarize, the set of lebesgue measurable sets are closed under complement and countable unions. Alternatively, the set of lebesgue measurable sets form a σ-algebra, defined as follows.

Definition A.1.4 (σ-algebra)**.** Given a set X, σ-algebra (Σ) is a collection of subsets of X ($\Sigma \subset \mathcal{P}(X)$) if we have the following:

1. $\phi \in \Sigma$
2. Closed under complement: If $E \in \Sigma$, then $E^c \in \Sigma$
3. Closed under countable union: If $E_n \in \Sigma$ for a countable collection $\{E_n\}$, then $\cup_n E_n \in \Sigma$.

Generalizing the discussion to arbitrary measure spaces. Lebesgue measure is an example of a function that maps a collection of subsets of $\mathbb{R}$ to non-negative real numbers, allowing us to measure these sets. We now present the concept of a measure, which is inspired from the properties of lebesgue measure and allows to define more general ways of "measuring" sets.

Definition A.1.5 (Measure)**.** Given a set X and σ-algebra Σ of X, a function $\mu : \Sigma \to [0, \infty]$ is called as measure if we have the following:

1. $\mu(\{\phi\}) = 0$
2. For countable collection of disjoint sets $\{E_n\}$ such that $E_n \in \Sigma$, we have $\mu(\cup_n E_n) = \sum_n \mu(E_n)$.

Further, (X, Σ) is called measurable space and the tuple (X, Σ, μ) is called measure space.

It is easy to see that lebesgue measure satisfies all the conditions in the above definition with $X = \mathbb{R}$, hence making it a valid measure. Another popular example of a measure is the **counting measure**, defined as follows: for all finite sets $E \in \Sigma$ for a measurable space (X, Σ) the counting measure (μ_c) is equal to the cardinality of the set, $\mu_c(E) = |E|$. For infinite sets, we define the counting measure to be infinite as well. Hence, counting measure provides with a very intuitive way of "measuring" sets, by simply counting the total numbers of elements in that set.

Appendix B

Supplementary Material: Background

B.1 Backdoor estimator for ATE

To recap the setup, we intervene on the binary treatment variable W and want to estimate the average treatment effect on outcome variable Y. Hence, we obtain the following as per the objective of ATE:

$$\begin{aligned}\text{ATE} :=\ & \mathbb{E}[Y|do(W=1)] - \mathbb{E}[Y|do(W=0)] \\ \implies & \int Y p(Y|do(W=1))dY - \int Y p(Y|do(W=0))dy\end{aligned}$$

Now using the backdoor estimator, we can express the interventional distribution as a function of the observed distribution, $p(Y|do(W)) = \int p(Y|X,W)p(X)dx$. Substituting this in the expression above for ATE we obtain the following:

$$\begin{aligned}\text{ATE} :=\ & \int Y p(Y|do(W=1))dY - \int Y p(Y|do(W=0))dy \\ \implies & \int Y \int p(Y|X,W=1)p(X)dx\ dy - \int Y \int p(Y|X,W=0)p(X)dx\ dy \\ \implies & \int \int Y p(Y|X,W=1)dy\ dx - \int \int Y p(Y|X,W=0)dy\ dx \\ \implies & \mathbb{E}_X[\,\mathbb{E}_Y[Y|X,W=1] - \mathbb{E}_X[\,\mathbb{E}_Y[Y|X,W=0]]\end{aligned}$$

B.2 Local vs Global Disentanglement

As per the Definition 1.2.3, we have that jacobian of $v : \mathcal{Z} \rightarrow \hat{\mathcal{Z}}$ is a permuted diagonal matrix. Let consider an example with 2-dimensional latent factors ($d_z = 2$) and say the jacobian Dv is given by the matrix $[[0,1],[1,0]]$, which implies $\dfrac{\partial v_1(z)}{\partial z_2} = 0$ and $\dfrac{\partial v_2(z)}{\partial z_1} = 0$. Lets focus on only the first expression, integrating that would imply:

$$v_1(z) = z_2 + C_1 \;\; \forall z \in \mathcal{Z}$$

However, C_1 is constant if the support $\mathcal{Z}$ is path-connected. Otherwise we could divide $\mathcal{Z} = \mathcal{Z}^1 \cup \mathcal{Z}^2$ such that $C_1(z) \neq C_1(z')$ where $z \in \mathcal{Z}^1$ and $z' \in \mathcal{Z}^2$. The implication of this is that $v_1(z)$ changes based on whether $z \in \mathcal{Z}^1$ or $z \in \mathcal{Z}^2$, and if the these supports were correlated with z_1, then $v_1(z)$ is a function of both z_1 and z_2! Which implies the disentanglement holds locally in regions of connected support but it does not hold globally across the entire support.

To understand this with an example, construct $\mathcal{Z}^1 = \{z \mid z_1 < -1\}$ and $\mathcal{Z}^2 = \{z \mid; \; z_1 > 1\}$. Also, $C_1 = -1 \; \forall z \in \mathcal{Z}^1$ and $C_1 = 1 \; \forall z \in \mathcal{Z}^2$. Then we have $v_1(z) = z_2 - 1 \; \forall z \in \mathcal{Z}^1$ and $v_1(z) = z_2 + 1 \; \forall z \in \mathcal{Z}^2$. Hence, for each connected block $\mathcal{Z}^1$ and $\mathcal{Z}^2$, the function $v_1(z)$ only depends on z_2 but that dependence changes based on z_1! Hence, globally $v_1(z)$ depends on both z_1 and z_2.

Hence, Definition 1.2.3 implies that a one-to-one correspondence between z and $\hat{z}$ locally, which may not hold across the entire support. If instead the function $v : \mathcal{Z} \rightarrow \hat{\mathcal{Z}}$ maintains the same local disentanglement structure across its entire support, then we term it as global disentanglement.

B.3 Indeterminacy in latent identification with reconstruction objective

We consider the optimal solution $\hat{f}, \hat{g}$ to the reconstruction objective, i.e., $\mathbb{E}_{x \sim X} ||x - \hat{g}(\hat{f}(x))||^2 = 0$, which implies $x = \hat{g}(\hat{f}(x)) \; \forall x \in \mathcal{X}$, *except over a set of measure zero.*

Now denoting $\hat{z} = \hat{f}(x)$ and substituting $x = g(z)$ per the data generation process, we establish the indeterminacy in latent recovery as follows:

$$\begin{aligned} &\hat{g}(\hat{f}(x)) = x \ \ \forall x \in \mathcal{X} \\ \implies &\hat{g}(\hat{z}) = g(z) \ \ \forall z \in \mathcal{Z} \\ \implies &\hat{z} = \hat{g}^{-1} \circ g(z) \ \ \forall z \in \mathcal{Z} \end{aligned}$$

Hence, we have $\hat{z} = v(z) = \hat{g}^{-1} \circ g(z) \ \ \forall z \in \mathcal{Z}$.

Note: The second step the derivation above relies on the assumption that g is an injective function. Otherwise we could have different $z, z' \in \mathbb{Z}$ such that $x = g(z) = g(z')$. Injectivity of the mixing function g is a crucial assumption in several works on identifiable representation learning.

Remark on v being diffeomorphism. If we assume that the true decoder g and the learned decoder $\hat{g}$ are diffeomorphisms, then v is also a diffeomorphism as diffeomorphisms are closed under composition. Hence, for some proof techniques we can even operate with the inverse mapping, i.e, we define $v : \hat{\mathcal{Z}} \to \mathcal{Z}$ as $z = v(\hat{z}) = g \circ \hat{g}(z) \ \ \forall \hat{z} \in \hat{\mathcal{Z}}$. For example, proofs in Chapter 3, we consider the v as a map from the learned latent space to the true latent space.

B.4 Proof of Proposition 1.2.4 (Linear ICA)

Proposition 1.2.4. *Let $Z, \hat{Z} \in \mathbb{R}^{d_z}$ be random variables s.t. $\hat{z} = Az \ \ \forall z \in \mathcal{Z}$. If Z and $\hat{Z}$ have mutually independent components and no component of Z is gaussian, then A is permutation & scaling matrix.*

Proof. We first state the Darmois-Skitovitch Theorem as it will be useful for proving this proposition.

Darmois-Skitovitch Theorem [Theis, 2004]: Let $Y_1 = \sum_{i=1}^{d} \alpha_i Z_i, \quad Y_2 = \sum_{i=1}^{d} \beta_i Z_i$ be linear combination of mutually independent random variables $\{Z_1, \cdots, Z_n\}$ where $\alpha_i, \beta_i \in \mathbb{R}$. If Y_1 and Y_2 are mutually independent, then all Z_i with $\alpha_i \beta_i \neq 0$ are gaussian.

Now lets consider two components of the learned latents, $\hat{z}_i, \hat{z}_j$, which can be written as follows using the relationship $\hat{z} = Az \ \forall z \in \mathbb{Z}$.

$$\hat{Z}_i = \sum_{k=1}^{d_z} A_{i,k} z_k \ , \quad \hat{Z}_j = \sum_{k=1}^{d_z} A_{j,k} z_k$$

Since both $Z, \hat{Z}$ has mutually independent components, therefore, by Darmois-Skitovitch

Theorem we have any latent component z_k with $A_{i,k} * A_{j,k} \neq 0$ must be gaussian. Since we have assumed that no latent component Z_k is gaussian, we must have $A_{i,k} * A_{j,k} = 0$ $\forall i, j, k \in [d_z]$ $, i \neq j$, which implies either $A_{i,k} = 0$ or $A_{j,k} = 0$ $\forall i, j, k \in [d_z]$ $, i \neq j$.

Now for a fixed k we cannot have $A_{i,k} = 0$ $\forall i \in [d_z]$ as the matrix A is invertible. Therefore, for each $k \in [d_z]$, we have a unique $i^* \in [d_z]$ such that $A_{i^*,k} \neq 0$ and $A_{i,k} = 0$ $\forall i \neq i^*$. Hence, each component of the true latent z_k can contribute to only one component of the learned latent $\hat{z}_{i^*}$. Alternatively, each column of the matrix A contains only a single non-zero entry, i.e., each column is 1-sparse. Since the matrix A is invertible, we must have the columns are linearly independent and because the columns are 1-sparse, each column $A_{:,k}$ is a scaling of the basis vector $e_{\pi(k)}$ where π is permutation over the set $[d_z]$. This concludes the proof that the matrix A is a permutation & scaling matrix. □

B.5 Non-Linear ICA Using Auxiliary Variables and Contrastive Learning

Theorem 1.2.6. *Given the data generation process (E.q. 1.9) and the optimal solution $(\hat{f}, \hat{h})$ under the learning objective (E.q. 1.10), along with the extra assumptions stated below:*

1. *The learned encoder $\hat{f} = (\hat{f}_1, \cdots, \hat{f}_{d_z})$ and the mixing function g are C^2-diffemorphisms.*

2. *The functions $q_i(z_i, u)$ in the log density function $p(Z|U)$ are C^2-functions.*

3. ***Assumption of Sufficient Variability.*** *Denote the diagonal terms of the jacobian and hessian of $q(z, u)$ w.r.t z as $w(z, u) = \big(\frac{\partial q_1(z_1,u)}{\partial z_1}, \cdots, \frac{\partial q_{d_z}(z_{d_z},u)}{\partial z_{d_z}}, \;\; \frac{\partial^2 q_1(z_1,u)}{\partial^2 z_1}, \cdots, \frac{\partial^2 q_{d_z}(z_{d_z},u)}{\partial^2 z_{d_z}}\big)$. Then $\forall z \in \mathcal{Z}$ $\exists$ $2 * d_z + 1$ values for u such that the following set of vectors are independent: $\Big\{ w(z, u_i) - w(z, u_0) \;\mid\; i \in [1, 2 * d_z] \Big\}$.*

Then we achieve disentanglement (Def. 1.2.3) with the learned latent variables $\hat{z}$ where $\hat{z} = \hat{f}(x)$.

To get some intuition behind the theorem, lets first understand how optimizing the learning objective constrains the indeterminacy in latent recovery, $\hat{z} = v(z) = \hat{f} \circ g(z)$ (E.q. 1.8). Note that for the optimal solution of binary logistic regression we have that the learned logit ($\hat{\psi}(x, u)$) should equal the difference in log density function of the two classes, i.e., $\log p(x, u) - \log p(x, u^*)$.

$$\hat{\psi}(x, u) = \log p(x, u) - \log p(x, u^*)$$

Note that x and u^* are independent as u^* is randomly sampled, hence $\log p(x, u^*) = \log p(x) + \log(u)$. As per the usual factorization of $\log p(x, u) = \log p(x|u) + log(u)$, we see that the factor of $\log p(x|u)$ cancel out and we have the following.

$$\hat{\psi}(x, u) = \log p(x|u) - \log p(x)$$

Now we perform change of variable $x = g(z)$ and using the change of probability density formulae $p(x) = p(g^{-1}(x)) \, |\det J(g^{-1}(x))|$, we have the following.

$$\begin{aligned}
&\hat{\psi}(x, u) = \log p(x|u) - \log p(x) \\
\implies &\hat{\psi}(x, u) = \log p(g^{-1}(x)|u) + \log|\det J(g^{-1}(x))| - \log p(g^{-1}(x)) - \log|\det J(g^{-1}(x))| \\
\implies &\hat{\psi}(x, u) = \log p(g^{-1}(x)|u) - \log p(g^{-1}(x)) \\
\implies &\sum_{i=1}^{d_z} \hat{h}_i(\hat{f}_i(x)), u) = \log p(g^{-1}(x)|u) - \log p(g^{-1}(x))
\end{aligned}$$

Now using $z = g^{-1}(x)$ and $\hat{z} = \hat{f}(x)$ s.t. $\hat{z} = v(z) = \hat{f} \circ g(x)$, we have the final expression.

$$\sum_{i=1}^{d_z} \hat{h}_i(v_i(z)), u) = \sum_{i=1}^{d_z} q_i(z_i, u) - logp(z)$$

Following the proof by the authors, we would differentiate the above equality twice w.r.t z, hence the assumption of C^2 for the functions $(g, \hat{f}, q_i)$. Also, the proof would require us to work with the inverse of $v(z)$, i.e., $v^{-1}(z) = g^{-1} \circ \hat{f}^{-1}(z)$, hence we need $(g, \hat{f})$ to be C^2-diffeomorphisms. The assumption of sufficient variability intuitively suggests that the effect of auxiliary information (U) on the latent variables (Z) should be diverse. Since this assumption is hard to justify, the authors also prove that exponential family distributions under some assumptions on their sufficient statistics would satisfy this assumption.

B.6 Weakly-supervised disentanglement without compromises

Theorem 1.2.7. *Given the data generation process (E.q. 1.11) and the optimal solution $(\hat{f}, \hat{g})$ under the learning objective (E.q. 1.12), along with the following extra assumptions:*

1. *The mixing function/true decoder (g) and the learned decoder $(\hat{g})$ are diffeomorphisms*

2. *The learner knows the coordinate set S which is fixed across all the data pairs.*

Then we achieve block disentanglement, i.e., $\hat{z} = v(z)$ such that $\hat{z}_S$ depends only on the latent component in S (z_S) and $\hat{z}_{\bar{S}}$ depends only on the latent components in $\bar{S}$ ($z_{\bar{S}}$).

To get some intuition behind the theorem, we first characterize the indeterminacy in latent recovery $\hat{z} = v(z) = \hat{g}^{-1} \circ g(z)$, which is derived in a similar way as with Auto-Encoders (E.q. 1.7). Then we utilize the constraints on the learned $(\hat{z}, \hat{\tilde{z}})$ and true latent pairs $(z, \tilde{z})$ that their components are mutually independent and they need to match on the set S, which further restricts the function v and allows us to achieve block disentanglement. I found flaws in the proof for this claim by the authors and have discussed that in detail ahead, where I also provide an alternative proof for this claim.

Before discussing the proof, there are some important extension of the block identifiability results proposed by the authors. The identification result above can be extended to the case when the learner does not know the set S but knows its cardinality, hence it knows the total number of shared components across the pairs $(X, \tilde{X})$ The learner enforces the constraint that components of $(\hat{Z}, \hat{\tilde{Z}})$ need to match on a coordinate set T where $|T| = |S|$. Therefore, we would still obtain block disentanglement, $\hat{z} = v(z)$ such that $\hat{z}_T$ depends only on the latent component in S (z_S) and $\hat{z}_{\bar{T}}$ depends only on the latent components in $\bar{S}$ ($z_{\bar{S}}$). For the case of varying S across data pairs $(X, \tilde{X})$, we can get stronger identification guarantees and the authors show that we can achieve disentanglement under assumptions on the variation of S.

Proof. We now discuss the proof for Theorem 1.2.7 done by the authors, and highlight the issues with their proof along with potential fix. We first restate the constraint that the learned latents variables $(\hat{z}, \hat{\tilde{z}})$ have shared components on the coordinate set S (E.q. 1.12).

$$\hat{z}_i = \hat{\tilde{z}}_i \quad \forall i \in S$$
$$\hat{z}_j \neq \hat{\tilde{z}}_j \quad \forall j \in \bar{S}$$

Note that for Auto-Encoders the learned latents $(\hat{z}, \hat{\tilde{z}})$ are related to the true latents $(z, \tilde{z})$ via the function $v(z) = \hat{g}^{-1} \circ g(z)\ \forall z \in \mathbb{Z}$; as proved in Appendix B.3. Hence, we substitute $\hat{z} = v(z)$ & $\hat{\tilde{z}} = v(\tilde{z})$ in the equation above.

$$v_i(z) = v_i(\tilde{z}) \quad \forall i \in S$$
$$v_j(z) \neq v_j(\tilde{z}) \quad \forall j \in \bar{S}$$

Following the data generation process(E.q. 1.11), we know the true latents also share components on the coordinate set S, i.e, $\tilde{z} = h(z, y, S)$ where $\tilde{z}_S = z_S$ and $\tilde{z}_{\bar{S}} = y$ such that $\tilde{z}_{\bar{S}} \neq z_{\bar{S}}$.

Lets assume the set S represents the first $d_z - k$ coordinates of the latents, as it does not affect our analysis and we would only incur a permutation indeterminacy. Hence, we can write $z = (z_S, z_{\bar{S}})$ and $\tilde{z} = (z_S, y)$ where $y \neq z_{\bar{S}}$. Substituting this in the equation above, we obtain the following.

$$v_i(z_S, z_{\bar{S}}) = v_i(z_S, y) \quad \forall i \in S \tag{a}$$

$$v_j(z_S, z_{\bar{S}}) \neq v_j(z_S, y) \quad \forall j \in \bar{S} \tag{b}$$

Comment on the proof by authors. The authors in their proof conclude from the above equality that the jacobian of v w.r.t z must be block-diagonal matrix with the first block of dimension $|S| \times |S|$ which implies $v_S(z)$ is a function of only z_S, and the second block of dimension $|\bar{S}| \times |\bar{S}|$ which implies $v_{\bar{S}}(z)$ is a function of only $z_{\bar{S}}$.

However, we can construct a counterexample with $z \in R^2$, $|S| = |\bar{S}| = 1$ and function v as a linear operator defined by the matrix $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. We have $v_1(z_1, z_2) = v_1(z_1, y)$ and $v_2(z_1, z_2) \neq v_2(z_1, y)$ $\forall z_2 \neq y$. Since v is a linear operator, its jacobain is the same v which is not block diagonal. Hence, the system of equations a and b can be satisfied by a function v such that the jacobian of v is not block diagonal.

Note that if we differentiate equation a w.r.t $z_j \in \bar{S}$, then we have the following.

$$\begin{aligned} \frac{\partial v_i(z_S, z_{\bar{S}})}{\partial z_j} &= \frac{\partial v_i(z_S, y)}{\partial z_j} \quad \forall i \in S, \ \forall j \in \bar{S} \\ \implies \frac{\partial v_i(z_S, z_{\bar{S}})}{\partial z_j} &= 0 \ \ \forall i \in S, \ \forall j \in \bar{S} \end{aligned} \tag{B.1}$$

Hence, we have $Jac_z(v)_{S,\bar{S}} = 0$, which implies $v_S(z)$ if a function of only z_S. Therefore, in general we can say the jacobain of v is $\begin{pmatrix} A_{|S|\times|S|} & 0_{|S|\times|\bar{S}|} \\ B_{|\bar{S}|\times|S|} & C_{|\bar{S}|\times|\bar{S}|} \end{pmatrix}$ where the block $B_{|\bar{S}|\times|S|}$ is not necessarily zero as shown by the counterexample above.

My attempt at the proof. So far we have established that following the system of equations a and b we can conclude that the jacobain of v is $\begin{pmatrix} A_{|S|\times|S|} & 0_{|S|\times|\bar{S}|} \\ B_{|\bar{S}|\times|S|} & C_{|\bar{S}|\times|\bar{S}|} \end{pmatrix}$. Now I will make use of the constraint of mutual independence between the components of the true latents z (E.q. 1.11) and the learned latents $\hat{z}$ (E.q. 1.12) and show that for linear functions v we will achieve block disentanglement (jacobian of v is block diagonal). Note that I could not prove this result for a general non-linear invertible function v.

Claim: If the function $\hat{z} = v(z)$ is an invertible linear transformation, then under the constraints a, b, and $B.1$, along with mutual independence in components of Z and $\hat{Z}$, we have that jacobian of v w.r.t z is block diagonal, $Jac_z(v) = \begin{pmatrix} A_{|S|\times|S|} & 0_{|S|\times|\bar{S}|} \\ 0_{|\bar{S}|\times|S|} & C_{|\bar{S}|\times|\bar{S}|} \end{pmatrix}$.

Proof for the claim. Since we have assumed that v is linear transformation, lets represent it via the matrix V, which implies $v_i(z) = \sum_{k\in S} V_{i,k}z_k + \sum_{k\in\bar{S}} V_{i,k}z_k$. Further, following E.q. B.1, we have $v_i(z) = \sum_{k\in S} V_{i,k}z_k \ \ \forall i \in S$ as there is no dependence on the components in $z_{\bar{S}}$.

Now pick any $i \in S$ and any $j \in \bar{S}$, then because $\hat{z}_i = v_i(z)$ and $\hat{z}_j = v_j(z)$ are independent, we must have the covariance between them must be zero, $Cov(\hat{z}_i, \hat{z}_j) = 0$, which can be further simplified as follows.

$$\begin{aligned} &Cov(v_i(z), v_j(z)) = 0 \\ \implies &Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in S} V_{j,k}z_k + \sum_{k\in\bar{S}} V_{j,k}z_k\Big) = 0 \\ \implies &Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in S} V_{j,k}z_k\Big) + Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in\bar{S}} V_{j,k}z_k\Big) = 0 \end{aligned} \tag{B.2}$$

Lets analyze the second term in the equation above, $Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in\bar{S}} V_{j,k}z_k\Big)$. This can be further simplified as follows:

$$Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in\bar{S}} V_{j,k}z_k\Big) = \sum_{k'\in S}\sum_{k\in\bar{S}} V_{i,k'}V_{j,k}Cov\Big(z_{k'}, z_k\Big) = 0$$

where the last equality follows from the assumption that the components of latent variable Z are mutually independent, hence $Cov\Big(z_{k'}, z_k\Big) = 0 \ \forall k' \in S, k \in \bar{S}$.

Therefore susbtituing this in E.q. B.2, we obtain $Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in S} V_{j,k}z_k\Big) = 0$, which is simplified further below.

$$\begin{aligned} &Cov\Big(\sum_{k'\in S} V_{i,k'}z_{k'},\ \sum_{k\in S} V_{j,k}z_k\Big) = 0 \\ \implies &\sum_{k'\in S}\sum_{k\in S} V_{i,k'}V_{j,k}Cov\Big(z_{k'}, z_k\Big) = 0 \\ \implies &\sum_{k\in S} V_{i,k}V_{j,k}Cov\Big(z_k, z_k\Big) \quad + \sum_{k,k'\in S\ ,\ k\neq k'} V_{i,k'}V_{j,k}Cov\Big(z_{k'}, z_k\Big) = 0 \\ \implies &\sum_{k\in S} V_{i,k}V_{j,k}Var\Big(z_k\Big) = 0 \end{aligned}$$

where the last equality follows from the assumption that the components of latent variable Z are mutually independent.

Lets denote $V_{j,k}Var\Big(z_k\Big)$ as $V_{j',k}$, then we have $\sum_{k\in S} V_{i,k}V_{j',k} = 0 \ \forall i \in S \ \& \ j \in \bar{S}$

This implies $\sum_{k\in S} V_{j',k}A^k = 0$ where $A^k = (V_{1,k}, \cdots, V_{|S|,k})$ denotes the restriction of the k-th column of matrix V to the first $|S|$ elements. Since the matrix V is invertible and has the form $\begin{pmatrix} A_{|S|\times|S|} & 0_{|S|\times|\bar{S}|} \\ B_{|\bar{S}|\times|S|} & C_{|\bar{S}|\times|\bar{S}|} \end{pmatrix}$, it implies that matrix $A_{|S|\times|S|}$ is invertible as well. Therefore, the set of vectors $\{A^1, \cdots, A^{|S|}\}$ are linearly independent.

Hence, $\sum_{k\in S} V_{j',k}A^k = 0$ implies $V_{j',k} = 0 \ \forall k \in S \ \& \ j \in \bar{S}$ due to the linear independence of columns of A, which can be further simplified as follow under the assumption that $Var(z_k) > 0 \ \forall k \in S$.

$$\begin{aligned} & V_{j',k} = 0 \ \forall k \in S \ \& \ j \in \bar{S} \\ \implies & V_{j,k}Var\Big(z_k\Big) = 0 \ \forall k \in S \ \& \ j \in \bar{S} \\ \implies & V_{j,k} = 0 \ \forall k \in S \ \& \ j \in \bar{S} \end{aligned}$$

This proves the claim that all elements the matrix V in the off diagonal blocks are zero. □

Appendix C

Supplementary Material: Towards efficient representation identification in supervised learning

C.1 Proof for Linear Identification with ERM.

Corollary C.1.1. *Given the data generation process (Assumption 2.2.1) except the assumption of mutual independence and non-gaussianity on Z, and the optimal solution $\Theta^\dagger \circ \Phi^\dagger$ to the ERM objective (2.4) with ℓ as square loss for regression and cross-entropy loss for classification, along with the exta assumptions stated below:*

- *Assumption 2.3.2 for the true solution g^{-1} and Γ.*
- *The number of tasks k is equal to the dimension of the latent d,*

Then we achieve linear identification (Definition 1.2.1) with the learned latent variables $\hat{z} = \Phi^\dagger(x)$.

Proof. Consider we are in the regression setting, then using $X \leftarrow g(Z)$ and $Y \leftarrow \Gamma Z + N$ as per the data generation process, the risk of a predictor f can be written as follows:

$$\begin{aligned} R(f) &= \mathbb{E}\big[\|Y - f(X)\|^2\big] \\ &= \mathbb{E}\big[\|\Gamma Z + N - f\circ g(Z)\|^2\big] \\ &= \mathbb{E}\Big[\|\Gamma Z - f\circ g(Z)\|^2\Big] + \mathbb{E}\big[\|N\|^2\big] - 2 * \mathbb{E}[(\Gamma Z - f\circ g(Z))^\intercal N] \\ &= \mathbb{E}\Big[\|\Gamma Z - f\circ g(Z)\|^2\Big] + \mathbb{E}\big[\|N\|^2\big] \qquad \text{(since } Z \perp N \text{ and } \mathbb{E}[N]=0) \end{aligned} \tag{C.1}$$

Hence, we have $R(f) \geq \mathbb{E}\big[\|N\|^2\big]$ for all functions $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$. Since $g^{-1} \in \mathcal{H}_\Phi$ and $\Gamma \in \mathcal{H}_\Theta$, $\Gamma \circ g^{-1}$ is a valid solution as per the ERM (2.3) objective and also achieves the lowest error possible, i.e., $R(\Gamma \circ g^{-1}) = \mathbb{E}[\|N\|^2]$. Hence, if we consider the optimal solution $(\Theta^\dagger \circ \Phi^\dagger\ )$ to ERM, then we must have the following equality except over a set of measure zero.

$$\begin{aligned} &\Theta^\dagger \circ \Phi^\dagger(X) = \Gamma Z \\ \implies &\Phi^\dagger(X) = (\Theta^\dagger)^{-1}\Gamma Z \\ \implies &\hat{Z} = AZ \end{aligned} \tag{C.2}$$

Note that the matrix $A = (\Theta^\dagger)^{-1}\Gamma$ is invertible as both $\Theta^\dagger$ and Γ are invertible due to Assumption 2.3.2. Therefore, we have $\hat{Z} = AZ$ with an invertible matrix A, therefore we achieve linear identification.

Classification Case. We can also carry out the same proof for the multi-task classification case. In multi-task classification we can write the condition for optimality as

$$\sigma\Big(\Omega^\dagger\Phi^\dagger(X)\Big) = \sigma\Big(\Gamma Z\Big) \tag{C.3}$$

where sigmoid is applied separately to each element, and since the sigmoids are equal, this implies the individual elements are also equal. Therefore, $\Omega^\dagger\Phi^\dagger(X) = \Gamma Z$ and we can use the same analysis as the regression case from this point on. □

C.2 Proof of Theorem 2.3.3: IC-ERM (case k=d)

Theorem 2.3.3. *Given the data generation process (Assumption 2.2.1) and the optimal solution $\Theta^\dagger \circ \Phi^\dagger$ to IC-ERM (2.3) with ℓ as square loss for regression and cross-entropy loss for classification, along with the exta assumptions stated below:*

- *Assumption 2.3.2 for the true solution g^{-1} and Γ.*
- *The number of tasks k is equal to the dimension of the latent d,*

Then we achieve permutation & scaling identifiability (Definition 1.2.2) with the learned latent variables $\hat{z} = \Phi^\dagger(x)$.

Proof. Consider we are in the regression setting for the data generation process in Assumption 2.2.1. Therefore, using $X \leftarrow g(Z)$ and $Y \leftarrow \Gamma Z + N$, the risk of a predictor f can be written as follows:

$$\begin{aligned}R(f) &= \mathbb{E}\big[\|Y - f(X)\|^2\big]\\ &= \mathbb{E}\big[\|\Gamma Z + N - f \circ g(Z)\|^2\big]\\ &= \mathbb{E}\Big[\|\Gamma Z - f \circ g(Z)\|^2\Big] + \mathbb{E}\big[\|N\|^2\big] - 2 * \mathbb{E}[(\Gamma Z - f \circ g(Z))^\mathsf{T} N]\\ &= \mathbb{E}\Big[\|\Gamma Z - f \circ g(Z)\|^2\Big] + \mathbb{E}\big[\|N\|^2\big] \qquad (\text{since } Z \perp N \text{ and } \mathbb{E}[N] = 0)\end{aligned} \tag{C.4}$$

From the above it is clear that $R(f) \geq \mathbb{E}\big[\|N\|^2\big]$ for all functions $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$. Since $g^{-1} \in \mathcal{H}_\Phi$, $\Gamma \in \mathcal{H}_\Theta$ and $g^{-1}(X)$ has all mutually independent components, $\Gamma \circ g^{-1}$ satisfies the constraints in IC-ERM (2.3) and also achieves the lowest error possible, i.e., $R(\Gamma \circ g^{-1}) = \mathbb{E}[\|N\|^2]$. Consider any solution to constrained ERM in (2.3). The solution must satisfy the following equality except over a set of measure zero.

$$\begin{aligned}\Theta^\dagger \circ \Phi^\dagger(X) &= \Gamma Z\\ \implies \Phi^\dagger(X) &= (\Theta^\dagger)^{-1}\Gamma Z\end{aligned} \tag{C.5}$$

Let us call $\Phi^\dagger(X) = Z^\dagger$ and $A = (\Theta^\dagger)^{-1}\Gamma$. Hence, the above equality becomes $Z^\dagger = AZ$, where all the components of $Z^\dagger$ are independent (Eq: (2.3)) and all the components of Z are independent (Assumption 2.2.1). We will now argue that the matrix A can be written as a permutation matrix times a scaling matrix. We first show that in each column of A there is exactly one non-zero element. Consider column k of A denoted as $[A]_k$. Since A is invertible all elements of the column cannot be zero. Now suppose at least two elements i and j of $[A]_k$ are non-zero. Consider the corresponding components of $Z^\dagger$. Since $Z_i^\dagger$ and $Z_j^\dagger$ are both independent and since $[A]_{ik}$ and $[A]_{jk}$ are both non-zero, from Darmois' theorem [Darmois, 1953] it follows that Z_k is a Gaussian random variable. However, this leads to a contradiction as we assumed none of the random variables in Z follow a Gaussian distribution. Therefore, exactly one element in $[A]_k$ is non-zero. We can say this about all the columns of A. No two columns will have the same row with a non-zero entry or otherwise A would not be invertible. Therefore, A can be expressed as a matrix permutation times a scaling matrix, where the scaling takes care of the exact non-zero value in the row and the permutation matrix takes care of the address of the element which is non-zero. This completes the proof. □

C.3 Proof of Theorem 2.4.1: ERM-ICA (Case k=d)

Theorem 2.4.1. *If Assumptions 2.2.1, 2.3.2 hold and the number of tasks k is equal to the dimension of the latent d, then the solution $\Omega^\dagger \circ \Phi^\dagger$ to ERM-ICA ((2.4), (2.6)) with ℓ*

as square loss for regression and cross-entropy loss for classification identifies true Z up to permutation and scaling.

Proof. Although the initial half of the proof is identical to the proof of Theorem 2.3.3 we repeat it for clarity. Consider we are in the regression setting for the data generation process in Assumption 2.2.1. Therefore, using $X \leftarrow g(Z)$ and $Y \leftarrow \Gamma Z + N$, the risk of a predictor f can be written as follows:

$$\begin{aligned}
R(f) &= \mathbb{E}\big[\|Y - f(X)\|^2\big] \\
&= \mathbb{E}\big[\|\Gamma Z + N - f \circ g(Z)\|^2\big] \\
&= \mathbb{E}\Big[\|\Gamma Z - f \circ g(Z)\|^2\Big] + \mathbb{E}\big[\|N\|^2\big] - 2 * \mathbb{E}[(\Gamma Z - f \circ g(Z))^{\mathsf{T}} N] \\
&= \mathbb{E}\Big[\|\Gamma Z - f \circ g(Z)\|^2\Big] + \mathbb{E}\big[\|N\|^2\big] \qquad (\text{since } Z \perp N \text{ and } \mathbb{E}[N] = 0)
\end{aligned} \tag{C.6}$$

From the above it is clear that $R(f) \geq \mathbb{E}\big[\|N\|^2\big]$ for all functions $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$. Since $g^{-1} \in \mathcal{H}_\Phi$, $\Gamma \in \mathcal{H}_\Theta$ and $g^{-1}(X)$ has all mutually independent components, $\Gamma \circ g^{-1}$ satisfies the constraints in IC-ERM (2.3) and also achieves the lowest error possible, i.e., $R(\Gamma \circ g^{-1}) = \mathbb{E}[\|N\|^2]$.

Consider any solution to ERM in (2.4). The solution must satisfy the following equality except over a set of measure zero.

$$\begin{aligned}
\Theta^\dagger \circ \Phi^\dagger(X) &= \Gamma Z \\
\implies \Phi^\dagger(X) &= (\Theta^\dagger)^{-1}\Gamma Z
\end{aligned} \tag{C.7}$$

Since $\Phi^\dagger(X)$ is a linear combination of independent latents with at least one latent non-Gaussian (and also the latents have a finite second moment). We can use the result from [Comon, 1994] that states $\Omega^\dagger$ that solves equation (2.6) relates to $(\Theta^\dagger)^{-1}$ as follows

$$(\Omega^\dagger)^{-1} = (\Gamma P \Lambda)^{-1} = \Lambda^{-1} P^{-1} \Gamma^{-1} \tag{C.8}$$

Substituting the above into (C.7) we get $\Phi^\dagger(X) = \Lambda^{-1}P^{-1}\Gamma^{-1}\Gamma Z = \Lambda^{-1}P^{-1}Z$. This completes the proof.

We can also carry out the same proof for the multi-task classification case. In multi-task classification we can write the condition for optimality as

$$\sigma\Big(\Omega^\dagger \Phi^\dagger(X)\Big) = \sigma\Big(\Gamma Z\Big) \tag{C.9}$$

where sigmoid is applied separately to each element, and since the sigmoids are equal, this implies the individual elements are also equal. Therefore, $\Omega^{\dagger}\Phi^{\dagger}(X) = \Gamma Z$ and we can use the same analysis as the regression case from this point on. Also, note that we equated the sigmoids in first place, because the LHS corresponds to $\hat{p}(Y|X)$ and RHS corresponds to true $p(Y|X)$. $\square$

C.4 Identification with fewer tasks than the latent dimension

In this section, we study the setting when the number of tasks k is equal to one. Since this setting is extreme, we need to make stronger assumptions to show latent identification guarantees. Before we lay down the assumptions, we provide some notation. Since we only have a single task, instead of using the matrix $\Gamma \in \mathbb{R}^{k\times d}$, we use $\gamma \in \mathbb{R}^{d}$ to signify the coefficients that generate the label in the single task setting. We assume each component of γ is non-zero. In the single task setting for regression problems, the label generation is written as $Y \leftarrow \gamma^{\mathsf{T}} Z + N_Y$, and the rest of the notation is the same as the data generation process in Assumption 2.2.1. We rewrite the data generation process in Assumption 2.2.1 for the single task case in terms of normalized variables $U = Z \odot \gamma$.

Assumption C.4.1. The data generation process for regressions is described as

$$\begin{aligned} Z &\leftarrow h(N_Z) \\ Y &\leftarrow \mathbf{1}^{\mathsf{T}} U + N_Y, \\ X &\leftarrow g^{'}(U), \end{aligned} \tag{C.10}$$

where $g^{'}(U) = g(U \odot \frac{1}{\gamma})$, where $U \odot \frac{1}{\gamma} = [\frac{U_1}{\gamma_1}, \cdots, \frac{U_d}{\gamma_d}]$. We assume that all the components of U are mutually independent and identically distributed (i.i.d.).

Note that $g^{'}$ is invertible since g is invertible and each element of γ is also non-zero. Hence, for simplicity, we can deal with the identification of U. If we identify U up to permutation and scaling, then Z is automatically identified up to permutation and scaling. The predictor we learn is a composition of linear predictor θ and a representation Φ, which is written as $\theta \circ \Phi(X) = \theta^{\mathsf{T}}\Phi(X)$. The learner searches for θ in the set $\mathcal{H}_\theta$, where $\mathcal{H}_\theta$ consists of linear predictors with all non-zero components, and Φ in the set $\mathcal{H}_\Phi$.

We can further simplify the predictor as follows: $\theta^{\mathsf{T}}\Phi(X) = \mathbf{1}^{\mathsf{T}}(\Phi(X)\odot\theta)$, where $\Phi(X)\odot\theta$ is component-wise multiplication expressed as $\Phi(X) \odot \theta = [\Phi_1(X) * \theta_1, \cdots, \Phi_d(X) * \theta_d]$. Therefore, instead of searching over $\mathcal{H}_\theta$ such that all components of θ are non-zero, we can fix $\mathcal{H}_\theta = \{\mathbf{1}\}$ and carry out the search over representations $\mathcal{H}_\Phi$ only. For the rest of the section, without loss of generality, we assume the predictor is of the form $\mathbf{1} \circ \Phi(X) = \mathbf{1}^{\mathsf{T}}\Phi(X)$. We restate the IC-ERM (2.3) with this parametrization and an additional constraint that all components are now required to be independent and identically distributed (i.i.d.). We provide a formal definition for the convenience of the reader below, where $\overset{d}{=}$ denotes identical in distribution.

Definition C.4.2. Independent & Identically Distributed (i.i.d.). A random vector $V = [V_1, \cdots, V_d]$ is said to be i.i.d. if 1) $V_i(X) \overset{d}{=} V_j(X)\ \forall i, j \in \{1, \cdots, d\}$ 2) $p(\{V_i\}_{i\in\mathcal{M}}) = \prod_{i\in\mathcal{M}} p(V_i)\ \forall \mathcal{M} \subseteq \{1, \cdots, d\}$.

The reparametrized IC-ERM (2.3) constraint is stated as follows.

$$\min_{\Phi\in\mathcal{H}_\Phi} R(\mathbf{1}\circ\Phi) \ \text{ s.t. } \Phi(X) \text{ is i.i.d. (Definition C.4.2)} \tag{C.11}$$

Next, we state the assumptions on each component of U (recall each component of U is i.i.d. from Assumption C.4.1) and $\mathcal{H}_\Phi$ under which we show that the reparametrized IC-ERM objective (equation (C.11)) recovers the true latent variables U up to permutations. We assume each component of U is a continuous random variable with probability density function (PDF) r. Define the support of each component of U as $\mathcal{S} = \{u \mid r(u) > 0, u \in \mathbb{R}\}$. Define a ball of radius $\sqrt{2}p$ as $\mathcal{B}_p = \{u \mid |u|^2 \leq 2p^2, u \in \mathbb{R}\}$.

Assumption C.4.3. Each component of U is a continuous random variable from the exponential family with probability density r. $\log(r)$ is a polynomial with degree p (where p is odd) written as

$$\log\big(r(u)\big) = \sum_{k=0}^{p} a_k u^k$$

where the absolute value of the coefficients of the polynomial are bounded by a_{max}, i.e., $|a_k| \leq a_{\mathsf{max}}$ for all $k \in \{1, \cdots, p\}$, and the absolute value of the coefficient of the highest degree term is at least a_{min}, i.e., $|a_p| \geq a_{\mathsf{min}} > 0$. The support of r is sufficiently large that it contains $\mathcal{B}_p$, i.e., $\mathcal{B}_p \subseteq \mathcal{S}$. Also, the moment generation function of each component i of U, $M_{U_i}(t)$, exists for all t.

Remark on the PDFs under the above assumption. The above assumption considers distributions in the exponential family, where the log-PDF can be expressed as a polynomial. Note that as long as the support of the distribution is bounded, every polynomial with bounded coefficients leads to a valid PDF (i.e., it integrates to one) and we only need to set the value of a_0 appropriately.

We now state our assumptions on the hypothesis class $\mathcal{H}_\Phi$ that the learner searches over. Observe that $\Phi(X)$ can be written as $h(U) = \Phi\Big(g^{'}(U)\Big)$ (since $X = g^{'}(U)$). We write the set of all the maps h constructed from composition of $\Phi \in \mathcal{H}_\Phi$ and $g^{'}$ as $\mathcal{H}_\Phi \circ g^{'}$. Define $w(u_1, \cdots, u_d) = \log\Big(\big|\mathsf{det}\big[J(h(u_1,\cdots,u_d))\big]\big|\Big)$, where det is the determinant, $J(h(u_1,\cdots,u_d))$ is the Jacobian of h computed at $(u_1,\cdots,u_d)$. The set of all the w's obtained from all $h \in \mathcal{H}_\Phi \circ g^{'}$ is denoted as $\mathcal{W}$

Assumption C.4.4. $\mathcal{H}_\Phi$ consists of analytic bijections. For each $\Phi \in \mathcal{H}_\Phi$, the moment generating function of each component $i \in \{1,\cdots,d\}$, $\Phi_i(X)$, denoted as $M_{\Phi_i(X)}(t)$ exists for all t. Each $w \in \mathcal{W}$ is a finite degree polynomial with degree at most q, where the absolute values of the coefficients in the polynomial are bounded by b_{max}.

Define $p_{\mathsf{min}} = \max\Big\{\kappa \log(8(d-1)), \frac{4a_{\mathsf{max}}(d-1)}{a_{\mathsf{min}}}, \frac{\log\Big(\frac{4b_{\mathsf{max}}*n_{\mathsf{poly}}}{a_{\mathsf{min}}}\Big)}{2} + q\Big\}$, where n_{poly} is the maximum number of non-zero coefficients in any polynomial $w \in \mathcal{W}$, κ is small constant (see Appendix C.5).

Theorem C.4.5 (Single Task Case)**.** *If the Assumptions C.4.1, C.4.3, C.4.4 hold,* $(g')^{-1} \in \mathcal{H}_\Phi$*, and* p *is sufficiently large* ($p \geq p_{\mathsf{min}}$)*, then the solution* $\Phi^\dagger(X)$ *of reparametrized IC-ERM objective* (C.11) *recovers the true latent* U *in the data generation process in Asssumption C.4.1 up to permutations.*

Proof sketch. The complete proof is available in Appendix Section C.5 and we provide an overview here. We use the optimality condition that the prediction made by the learned model, $\mathbf{1}^\mathsf{T} h(U)$, exactly matches the true mean, $\mathbf{1}^\mathsf{T} U$, along with the constraints that each component of U are i.i.d. and each component of $h(U)$ are i.i.d., to derive that the distributions of U and $h(U)$ are the same. We substitute this condition in the change of variables formula that relates the densities of U and $h(U)$. Using the Assumption C.4.4, we can show that if the highest absolute value of U and the highest absolute value of $h(U)$ are not equal, then the term with the highest absolute value among U and $h(U)$ will dominate, leading to contradiction in the identity obtained by change of variables formula. Based on this, we can conclude that the highest absolute value of U and the highest absolute value of $h(U)$ must be equal. We iteratively apply this argument to show that all the absolute values of U and $h(U)$ are related by permutation. We can extend this argument to the actual values instead of absolute values. Since h is analytic we argue that the relationship of permutation holds in a neighborhood. Then we use properties of analytic functions [Mityagin, 2015] to conclude that the relationship holds on the entire space.

Why does the bound on p grow linearly in d? We provide some geometric intuition into why the degree of the polynomial (p) of the log-PDF ($\log(r)$) needs to be large. If the dimension of the latent space d is large, then the second term in p_{min} dominates, i.e., p has to grow linearly in d. The simplification in the proof yields that the mapping h must satisfy the following condition – $\|u\|_k = \|h(u)\|_k$ for all $k \in \{1, \cdots, p\}$, where $\|\cdot\|_k$ is the k^{th} norm. Hence, h is a bijection that preserves all norms up to the p^{th} norm. If U is 2 dimensional, then the a bijection h that preserves the ℓ_1 norm and the ℓ_2 norm is composed of permutations and sign-flips. In general, since U is d dimensional, we need at least d constraints on h in the form $\|u\|_k = \|h(u)\|_k$ and thus $p \geq d$, which ensure that the only map that satisfies these constraints is composed of permutations and sign-flips.

Significance and remarks on Theorem C.4.5. Theorem C.4.5 shows that if we use IC-ERM principle, i.e., constraint the representations to be independent, then we continue to recover the latents even if the number of tasks is small. We can show that the above theorem also extends to binary classification. We admit that strong assumptions were made to arrive at the above result, while other assumptions such as bound on p growing linearly in d seem necessary, but we would like to remind the reader that we are operating in the extreme single task regime. In the previous section, when the number of tasks was equal to the dimension of the latent (when we have sufficiently many tasks), we had shown the success of the IC-ERM (2.3) objective (Theorem 2.3.3) for identification of latent variables with much fewer assumptions. In contrast, in Theorem C.4.5, we saw that with more assumptions on the distribution we can guarantee latent recovery with even one task. If we are in the middle, i.e., when the number of tasks is between one and the dimension of the latent, then the above Theorem C.4.5 says we only require the assumptions (Assumptions C.4.1, C.4.3, C.4.4) to hold for at least one task.

Note on the case k>d: We did not discuss the case when the number of tasks is greater than the dimension of the latent variables. This is because we can select a subset $S^{'}$ of tasks, such that $|S^{'}| = d$ and then proceed in a similar fashion as Theorem 2.3.3. This question arises commonly in linear ICA literature, and selecting a subset of tasks is the standard practice.

C.5 Proof of Theorem C.4.5: IC-ERM for the case k=1

Theorem C.4.5 (Single Task Case). *If the Assumptions C.4.1, C.4.3, C.4.4 hold, $(g')^{-1} \in \mathcal{H}_{\Phi}$, and p is sufficiently large ($p \geq p_{\mathsf{min}}$), then the solution $\Phi^{\dagger}(X)$ of reparametrized IC-ERM objective (C.11) recovers the true latent U in the data generation process in Asssumption C.4.1 up to permutations.*

Proof. We write $\Phi^{\dagger}(X) = [\Phi^{\dagger}_1(X), \cdots, \Phi^{\dagger}_d(X)]$. We call $\Phi^{\dagger}_i(X) = V_i$ and the vector $V = [V_1, \cdots, V_d] = [\Phi^{\dagger}_1(X), \cdots, \Phi^{\dagger}_d(X)]$.

Observe that since $(g^{'})^{-1} \in \mathcal{H}_{\Phi}$, we can use the same argument used in the proof of Theorem 2.3.3 to conclude that $(g^{'})^{-1}$ is a valid solution for the reparametrized IC-ERM objective (Eq: (C.11)). Notice that the terms $\Theta^{\dagger}$ and Γ that appear in the proof of Theorem 2.3.3, they are equal to identity here due to the Assumption C.4.1 and IC-ERM (C.11). Therefore, following the standard argument in Theorem 2.3.3, we can conclude that any solution $\Phi^{\dagger}(X)$ of reparametrized IC-ERM (C.11) satisfies the following:

$$\begin{aligned}\sum_i \Phi^{\dagger}_i(X) &= \sum_i (g^{'})^{-1}_i(X) \\ \sum_i V_i &= \sum_i U_i\end{aligned} \tag{C.12}$$

We write the moment generation function of a random variable U_i as $M_{U_i}(t) = \mathbb{E}[e^{tU_i}]$. We substitute the moment generating functions to get the following identity.

$$\sum_i V_i = \sum_i U_i \implies \Pi_i M_{V_i}(t) = \Pi_i M_{U_i}(t) \tag{C.13}$$

Since $U_i \overset{d}{=} U_j$ and $V_i \overset{d}{=} V_j$, we can use $M_{U_i}(t) = M_{U_j}(t)$ and $M_{V_i}(t) = M_{V_j}(t)$ for all $t \in \mathbb{R}$ and simplify as follows:

$$\begin{aligned}\Big(M_{V_i}(t)\Big)^d = \Big(M_{U_i}(t)\Big)^d &\implies M_{V_i}(t) = M_{U_i}(t) \\ &\implies V_i \overset{d}{=} U_i, \forall i \in \{1, \cdots, d\}\end{aligned} \tag{C.14}$$

In the second step of the above simplification, we use the fact that the moment generating function is positive. In the third step, we use the fact that if moment generating functions exist and are equal, then the random variables are equal in distribution [Feller, 2008]. Having established that the distributions are equal, we now show that the random variables are equal up to permutations.

Since the vector V is an invertible transform h of U, where $V = h(U)$. We can write the pdf of U in terms of V as follows.

$$\prod_i p(u_i) = \prod_i p(v_i)\big|\mathsf{det}(\mathsf{J}(h(u)))\big|, \tag{C.15}$$

where p corresponds to the pdf of U_i (recall that pdfs of all components is the same). We take log on both sides of the above equation to get the following:

$$\sum_i \log(p(u_i)) = \sum_i \log(p(v_i)) + \log\Big(\big|\mathsf{det}(\mathsf{J}(h(u)))\big|\Big) \tag{C.16}$$

From Assumption C.4.3, we substitute a polynomial for $\log(p(u)) = \sum_{k=0}^{p} a_k u^k$. From Assumption C.4.4, we express this $\log\Big(\big|\mathsf{det}(\mathsf{J}(h(u)))\big|\Big) = \sum b_k \prod_i u_i^{\theta_k(i)}$. We substitute these polynomials into the above equation to get the following:

$$\begin{aligned} &\sum_i \log(p(u_i)) - \sum_i \log(p(v_i)) - \log\Big(\big|\mathsf{det}(\mathsf{J}(h))\big|\Big) = 0 \\ \implies &\sum_{k=1}^{p} a_k \sum_{i=1}^{d} (u_i^k - v_i^k) - \sum_m b_m \prod_i u_i^{\theta_m(i)} = 0 \end{aligned} \tag{C.17}$$

In the proof, we first focus on comparing the largest absolute value among u's and largest absolute value among v's. Without loss of generality, we assume that $|u_j| > |u_i|$ for all $i \neq j$ (u_j is the largest absolute value among u's) and $|v_r| > |v_i|$ for all $i \neq r$ (v_r is the largest absolute value among v's). Consider the setting when $|u_j| \geq p^2 > 1$ (these points exist in the support because of the Assumption C.4.3). We can write $|u_j| = \alpha p^2$, where $\alpha \geq 1$.

There are three cases to further consider:

a) $|v_r| > |u_j|$

b) $|v_r| < |u_j|$

c) $|v_r| = |u_j|$

We first start by analyzing the case b). Since all values of $|v_i|$ and $|u_i|$ are also strictly less than $|u_j|$, we have the following:

- $\exists\, c < 1$ such that $|u_i| < c\alpha p^2 \;\; \forall\, i \neq j$
- $|v_i| < c\alpha p^2 \;\; \forall\, i \in \{1, \cdots, d\}$, where $c < 1$.

We divide the identity in equation (C.17) by u_j^p and separate the terms in such a way that only the term a_p is in the LHS and the rest of the terms are pushed to the RHS. We show further simplification of the identity below.

$$
\begin{aligned}
&|a_p| = \Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{u_j^p} - \frac{v_i^p}{u_j^p}\Big) + \sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{u_j^p} - \frac{v_i^k}{u_j^p}\Big) - \sum_m b_m \frac{1}{u_j^p}\prod_i u_i^{\theta_m(i)}\Big| \\
&\implies |a_p| \le \Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{u_j^p} - \frac{v_i^p}{u_j^p}\Big)\Big| + \Big|\sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{u_j^p} - \frac{v_i^k}{u_j^p}\Big)\Big| + \Big|\sum_m b_m \frac{1}{u_j^p}\prod_i u_i^{\theta_m(i)}\Big| \\
&\implies 1 \le \frac{1}{|a_p|}\Bigg(\Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{u_j^p} - \frac{v_i^p}{u_j^p}\Big)\Big| + \Big|\sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{u_j^p} - \frac{v_i^k}{u_j^p}\Big)\Big| + \Big|\sum_m b_m \frac{1}{u_j^p}\prod_i u_i^{\theta_m(i)}\Big|\Bigg)
\end{aligned}
\tag{C.18}
$$

We analyze each of the terms in the RHS separately. The simplification of the first term yields the following expression.

$$
\begin{aligned}
\frac{1}{|a_p|}\Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{u_j^p} - \frac{v_i^p}{u_j^p}\Big)\Big| &\le \frac{1}{|a_p|}\Big|a_p\Big| \sum_{i=1}^{d-1}\Big(\Big|\frac{u_i^p}{u_j^p}\Big| + \Big|\frac{v_i^p}{u_j^p}\Big|\Big) \\
&\le 2c^p(d-1)
\end{aligned}
\tag{C.19}
$$

The simplification of the second term in the RHS of the last equation in (C.18) yields the following expression.

$$
\begin{aligned}
\frac{1}{|a_p|}\Big|\sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{u_j^p} - \frac{v_i^k}{u_j^p}\Big)\Big| &\le \frac{1}{|a_p|}\sum_{k=1}^{p-1}|a_k| \sum_{i=1}^{d}\Big(\Big|\frac{u_i^k}{u_j^p}\Big| + \Big|\frac{v_i^k}{u_j^p}\Big)\Big| \\
&\le \sum_{k=1}^{p-1} 2|a_k| \sum_{i=1}^{d}\Big(\Big|\frac{u_j^k}{u_j^p}\Big|\Big)\Big| \\
&\le \frac{1}{|a_p|}\Big(\sum_{k=1}^{p-1} 2|a_k|(d-1)\Big)\frac{1}{\alpha p^2} \\
&\le \frac{2a_{\mathsf{max}}(d-1)}{a_{\mathsf{min}}p}
\end{aligned}
\tag{C.20}
$$

The simplification of the third term in the RHS of the last equation in (C.18) yields the following expression.

$$
\begin{aligned}
\frac{1}{|a_p|}\Big|\sum_m b_m \frac{1}{u_j^p}\prod_i u_i^{\theta_m(i)}\Big| &\le \sum_m |b_m|\frac{1}{|u_j|^{(p-q)}} \\
&\le \frac{b_{\mathsf{max}}}{a_{\mathsf{min}}}\frac{n_{\mathsf{poly}}}{(\alpha p^2)^{(p-q)}}
\end{aligned}
\tag{C.21}
$$

where n_{poly} corresponds to the number of non-zero terms in the polynomial expansion of the log-determinant. In the above simplification, we used the fact that $\sum_i \theta_m(i) \leq q$ and $|u_i| < |u_j|$. Analyzing the RHS in equations (C.19)-(C.21), we see that if p becomes sufficiently large, the RHS becomes less than 1. This contradicts the relationship in equation (C.18). Therefore, all $|v_i|$ cannot be strictly less than $|u_i|$. This rules out case b).

We now derive the bounds on the value of p as follows. Assume $p \geq 2q$, i.e., the degree of the log-pdf of each component of U is at least twice the degree of the log-determinant of the Jacobian of h.

From the equation (C.19) we get the following bound on p

$$\begin{aligned} & 2c^p(d-1) \leq \frac{1}{4} \\ \implies & 8(d-1) \leq \frac{1}{c^p} \\ \implies & \log(8(d-1)) \leq p\log(\frac{1}{c}) \\ \implies & p \geq \frac{\log(8(d-1))}{\log(\frac{1}{c})} \end{aligned} \tag{C.22}$$

From the equation (C.20) we get the following bound on p

$$\frac{a_{\mathsf{max}}(d-1)}{a_{\mathsf{min}}p} \leq \frac{1}{4} \implies p \geq \frac{4a_{\mathsf{max}}(d-1)}{a_{\mathsf{min}}} \tag{C.23}$$

From the equation (C.21) we get the following bound on p

$$\begin{aligned} & \frac{b_{\mathsf{max}}}{|a_p|}\frac{n_{\mathsf{poly}}}{(\alpha p^2)^{(p-q)}} \leq \frac{1}{4} \\ \implies & \log\left(\frac{4b_{\mathsf{max}}n_{\mathsf{poly}}}{|a_p|}\right) \leq 2(p-q)\log p \\ \implies & p \geq \frac{\log\left(\frac{4b_{\mathsf{max}}n_{\mathsf{poly}}}{a_{\mathsf{min}}}\right)}{2} + q \end{aligned} \tag{C.24}$$

From the above equations (C.22), (C.23), and (C.24), we get that if

$$p \geq \max\left\{\frac{\log(8(d-1))}{\log(\frac{1}{c})}, \frac{4a_{\mathsf{max}}(d-1)}{a_{\mathsf{min}}}, \frac{\log\left(\frac{4b_{\mathsf{max}}n_{\mathsf{poly}}}{a_{\mathsf{min}}}\right)}{2} + q\right\} \tag{C.25}$$

then the sum of the terms in the RHS in equation (C.18) is at most $\frac{3}{4}$ and the term in the LHS in equation (C.18) is 1, which leads to a contradiction.

From the above expression, we gather that the second and third term should dominate in determining the lower bound for p. From the second term, we gather that the lower bound increases linearly in the dimension of the latent, and from the third term we gather that p must be greater than q by a factor that grows logarithmically in the number of terms in the polynomial of the log determinant.

Let us now consider the case a) ($|v_r| > |u_j|$) which is similar to the case b) analyzed above. Since values of $|u_i|$ are strictly less than $|u_j|$ and $|v_r|$, and since $|v_r| > |u_j|$, there exist a $c < 1$ such that $|u_i| \leq c\alpha p^2$, $|v_r| \geq \frac{1}{c}\alpha p^2$, $|v_i| \leq c|v_r|$, where $c < 1$. We follow the same steps as done in the analysis for case b). We separate the equation so that only the term a_p (obtained by dividing v_r^p with v_r^p) is in the LHS and the rest on the RHS.

$$\begin{aligned}
|a_p| &= \Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{v_r^p} - \frac{v_i^p}{v_r^p}\Big) + \sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{v_r^p} - \frac{v_i^k}{v_r^p}\Big) - \sum_m b_m \frac{1}{v_r^p}\prod_i u_i^{\theta_m(i)}\Big| \\
\implies |a_p| &\leq \Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{v_r^p} - \frac{v_i^p}{v_r^p}\Big)\Big| + \Big|\sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{v_r^p} - \frac{v_i^k}{v_r^p}\Big)\Big| + \Big|\sum_m b_m \frac{1}{u_j^p}\prod_i u_i^{\theta_m(i)}\Big| \\
\implies 1 &\leq \frac{1}{|a_p|}\Bigg(\Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{v_r^p} - \frac{v_i^p}{v_r^p}\Big)\Big| + \Big|\sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{v_r^p} - \frac{v_i^k}{v_r^p}\Big)\Big| + \Big|\sum_m b_m \frac{1}{v_r^p}\prod_i u_i^{\theta_m(i)}\Big|\Bigg)
\end{aligned} \tag{C.26}$$

We analyze each of the terms in the RHS in equation (C.26) separately. The simplification of the first term in the RHS of the above yields the following upper bound.

$$\begin{aligned}
\frac{1}{|a_p|}\Big|a_p \sum_{i=1}^{d-1}\Big(\frac{u_i^p}{v_r^p} - \frac{v_i^p}{v_r^p}\Big)\Big| &\leq \frac{1}{|a_p|}\Big|a_p\Big|\sum_{i=1}^{d-1}\Big(\Big|\frac{u_i^p}{v_r^p}\Big| + \Big|\frac{v_i^p}{v_r^p}\Big|\Big) \\
&\leq 2c^p(d-1)
\end{aligned} \tag{C.27}$$

The simplification of the second term in the RHS of equation (C.26) yields the following upper bound.

$$
\begin{aligned}
\frac{1}{|a_p|}\Big|\sum_{k=1}^{p-1} a_k \sum_{i=1}^{d}\Big(\frac{u_i^k}{v_r^p}-\frac{v_i^k}{v_r^p}\Big)\Big| &\leq \frac{1}{|a_p|}\sum_{k=1}^{p-1}|a_k|\sum_{i=1}^{d}\Big(\Big|\frac{u_i^k}{v_r^p}\Big|+\Big|\frac{v_i^k}{v_r^p}\Big)\Big| \\
&\leq \sum_{k=1}^{p-1}2|a_k|\sum_{i=1}^{d}\Big(\Big|\frac{u_j^k}{v_r^p}\Big|\Big)\Big| \\
&\leq \frac{1}{|a_p|}\Big(\sum_{k=1}^{p-1}2|a_k|(d-1)\Big)\frac{1}{\alpha p^2} \\
&\leq \frac{a_{\mathsf{max}}(d-1)c}{|a_p|\alpha p} \\
&\leq \frac{a_{\mathsf{max}}(d-1)}{a_{\mathsf{min}}p}
\end{aligned}
\tag{C.28}
$$

The simplification of the third term in the RHS of equation (C.26) yields the following upper bound.

$$
\begin{aligned}
\frac{1}{|a_p|}\Big|\sum_{m} b_m \frac{1}{v_r^p}\prod_i u_i^{\theta_m(i)}\Big| &\leq \sum_m |b_m|\frac{1}{|v_r|^{(p-q)}} \\
&\leq \frac{b_{\mathsf{max}}}{|a_p|}\frac{n_{\mathsf{poly}}\Big(c^{p-q}\Big)}{(\alpha p^2)^{(p-q)}} \\
&\leq \frac{b_{\mathsf{max}}}{|a_p|}\frac{n_{\mathsf{poly}}}{(\alpha p^2)^{(p-q)}}
\end{aligned}
\tag{C.29}
$$

Analyzing the RHS in equation (C.27)-(C.29), we see that if p becomes sufficiently large then the RHS becomes less than 1. This contradicts the relationship in equation (C.26). Therefore, there is no $|v_i|$ that is strictly larger than $|u_i|$. This rules out case a). In fact from equations (C.27)-(C.29) we can get the same bound on p as in case b).

Thus, the only possibility is case c), i.e., $|v_r| = |u_j| \implies v_r = u_j$ or $v_r = -u_j$. Consider the case when p is odd. In that case, $v_r = u_j$ is the only option that works. We substitute $v_r = u_j$ in the equation (C.18) and repeat the same argument for the second highest absolute value, and so on. This leads to the conclusion that for each component u there is a component of v such that both of them are equal. Hence, we have established the relationship $v_r = u_j$. For another sample where index j corresponds to the highest absolute value and is in the neighbourhood of u_j, the relationship $v_r = u_j$ must continue to hold. If this does not happen, then there would be another component $v_q = u_j$ where $q \neq r$. However, if that were the case, then it would contradict the continuity of h.

Therefore, the relationship $v_r = u_j$ (the match between index j for u and index r for v) holds for a neighbourhood of values of vector u. Since each component of h is analytic, we can use the fact that the neighbourhood of vector of values of u for which the relationship $v_r = u_j$ holds has a positive measure, and then from [Mityagin, 2015] it follows that this relationship would hold on the entire space. We can draw the same conclusion for all the components of h and conclude that h is a permutation map.

□

C.6 Experiments: Implementation Details

Model Architecture

- Fully Connected Layer: (Data Dim, 100)
- BatchNormalization(100)
- LeakyReLU(0.5)
- Full Connected Layer: (100, Data Dim)
- BatchNormalization(Data Dim)
- LeakyReLU(0.5)
- Fully Connected Layer: (Data Dim, Total Tasks)

We consider the part of the network before the final fully connected layer as the representation network, and use the output from the representation network for training ICA/PCA after the ERM step.

Hyperparameters We use SGD to train all the methods with the different hyperparameters across each task mentioned below. In every case, we select the best model during the course of training based on the validation loss, and also use a learning rate scheduler that reduces the existing learning rate by half after every 50 epochs. Also, regarding ICA, we use the FastICA solver in sklearn with $30,000$ maximum iterations and data whitening.

- **Regression:** Learning Rate: 0.01, Batch Size: 512, Total Epochs: 1000, Weight Decay: $5e-4$, Momentum: 0.9
- **Classification:** Learning Rate: 0.05, Batch Size: 512, Total Epochs: 200, Weight Decay: $5e-4$, Momentum: 0.9

Appendix D

Supplementary Material: Additive Decoders for Latent Variables Identification and Cartesian-Product Extrapolation

$g\big\|_A$		Restriction of g to the set A
Dg		Jacobian of g
D^2g		Hessian of g
$B \subseteq [n]$		Subset of indices
$\|B\|$		Cardinality of the set B
x_B		Vector formed with the ith coordinates of x, for all $i \in B$
$X_{B,B'}$		Matrix formed with the entries $(i,j) \in B \times B'$ of X.
Given $\mathcal{X} \subseteq \mathbb{R}^n$, $\mathcal{X}_B$	$:=$	$\{x_B \mid x \in \mathcal{X}\}$ (projection of $\mathcal{X}$)
$\mathcal{B}$		A partition of $[d_z]$ (assumed contiguous w.l.o.g.)
$B \in \mathcal{B}$		A block of the partition $\mathcal{B}$
$B(i) \in \mathcal{B}$		The unique block of $\mathcal{B}$ that contains i
$\pi : [d_z] \to [d_z]$		A permutation
$S_{\mathcal{B}}$	$:=$	$\bigcup_{B\in\mathcal{B}} B^2$
$S_{\mathcal{B}}^c$	$:=$	$[d_z]^2 \setminus S_{\mathcal{B}}$
$\mathbb{R}^{d_z\times d_z}_{S_{\mathcal{B}}}$	$:=$	$\{M \in \mathbb{R}^{d_z\times d_z} \mid (i,j) \not\in S_{\mathcal{B}} \implies M_{i,j} = 0\}$
$\overline{\mathcal{X}}$		Closure of the subset $\mathcal{X} \subseteq \mathbb{R}^n$ in the standard topology of $\mathbb{R}^n$
$\mathcal{X}^\circ$		Interior of the subset $\mathcal{X} \subseteq \mathbb{R}^n$ in the standard topology of $\mathbb{R}^n$

Table D.1: Recurrent notations specific to this chapter.

D.1 Identifiability and Extrapolation Analysis

D.1.1 Useful definitions and lemmas

We start by recalling some notions of general topology that are going to be used later on. For a proper introduction to these concepts, see for example Munkres [2000].

Definition D.1.1 (Regularly closed sets)**.** A set $\mathcal{Z} \subseteq \mathbb{R}^{d_z}$ is regularly closed if $\mathcal{Z} = \overline{\mathcal{Z}^\circ}$, i.e. if it is equal to the closure of its interior (in the standard topology of $\mathbb{R}^n$).

Definition D.1.2 (Connected sets)**.** A set $\mathcal{Z} \subseteq \mathbb{R}^{d_z}$ is connected if it cannot be written as a union of non-empty and disjoint open sets (in the subspace topology).

Definition D.1.3 (Path-connected sets)**.** A set $\mathcal{Z} \subseteq \mathbb{R}^{d_z}$ is path-connected if for all pair of points $z^0, z^1 \in \mathcal{Z}$, there exists a continuous map $\boldsymbol{\phi} : [0, 1] \to \mathcal{Z}$ such that $\boldsymbol{\phi}(0) = z^0$ and $\boldsymbol{\phi}(1) = z^1$. Such a map is called a path between z^0 and z^1.

Definition D.1.4 (Homeomorphism)**.** Let A and B be subsets of $\mathbb{R}^n$ equipped with the subspace topology. A function $f : A \to B$ is an homeomorphism if it is bijective, continuous and its inverse is continuous.

The following technical lemma will be useful in the proof of Theorem 3.4.4. For it, we will need additional notation: Let $S \subset A \subset \mathbb{R}^n$. We already saw that $\overline{S}$ refers to the closure S in the $\mathbb{R}^n$ topology. We will denote by $\text{cl}_A(S)$ the closure of S in the subspace topology of A induced by $\mathbb{R}^n$, which is not necessarily the same as $\overline{S}$. In fact, both can be related via $\text{cl}_A = \overline{S} \cap A$ (see Munkres [2000, Theorem 17.4, p.95]).

Lemma D.1.5. *Let $A, B \subset \mathbb{R}^n$ and suppose there exists an homeomorphism $f : A \to B$. If A is regularly closed in $\mathbb{R}^n$, we have that $B \subset \overline{B^\circ}$.*

Proof. Note that $f\big|_{A^\circ}$ is a continuous injective function from the open set A° to $f(A^\circ)$. By the "invariance of domain" theorem [Munkres, 2000, p.381], we have that $f(A^\circ)$ must be open in $\mathbb{R}^n$. Of course, we have that $f(A^\circ) \subset B$, and thus $f(A^\circ) \subset B^\circ$ (the interior of B is the largest open set contained in B). Analogously, $f^{-1}\big|_{B^\circ}$ is a continuous injective function from the open set B° to $f^{-1}(B^\circ)$. Again, by "invariance of domain", $f^{-1}(B^\circ)$ must be open in $\mathbb{R}^n$ and thus $f^{-1}(B^\circ) \subset A^\circ$. We can conclude that $f(A^\circ) = B^\circ$, as shown below:

$$B = f(A) = f(\overline{A^\circ}) = f(\overline{A^\circ} \cap A) = f(\text{cl}_A(A^\circ)) \subseteq \text{cl}_B(f(A^\circ)) = \text{cl}_B(B^\circ) = \overline{B^\circ} \cap B \subset \overline{B^\circ}\,,$$

where the first inclusion holds by continuity of f [Munkres, 2000, Thm.18.1 p.104]. □

Lemma D.1.6 (Sparsity pattern of an invertible matrix contains a permutation)**.** *Let $L \in \mathbb{R}^{m\times m}$ be an invertible matrix. Then, there exists a permutation σ such that $L_{i,\sigma(i)} \neq 0$ for all i.*

Proof. This lemma is taken from Lachapelle et al. [2022b]. Since the matrix $\boldsymbol{L}$ is invertible, its determinant is non-zero, i.e.

$$\det(L) := \sum_{\pi \in \mathfrak{S}_m} \text{sign}(\pi) \prod_{i=1}^{m} L_{i,\pi(i)} \neq 0\,, \tag{D.1}$$

where $\mathfrak{S}_m$ is the set of m-permutations. This equation implies that at least one term of the sum is non-zero, meaning there exists $\pi \in \mathfrak{S}_m$ such that for all $i \in [m]$, $L_{i,\pi(i)} \neq 0$. □

Definition D.1.7 (Aligned subspaces of $\mathbb{R}^{m\times n}$)**.** Given a subset $S \subset \{1, ..., m\} \times \{1, ..., n\}$, we define

$$\mathbb{R}^{m\times n}_S := \{M \in \mathbb{R}^{m\times n} \mid (i,j) \notin S \implies M_{i,j} = 0\}\,. \tag{D.2}$$

Definition D.1.8 (Useful sets)**.** Given a partition $\mathcal{B}$ of $[d]$, we define

$$S_{\mathcal{B}} := \bigcup_{B\in\mathcal{B}} B^2 \qquad S^c_{\mathcal{B}} := \{1, \dots, d_z\}^2 \setminus S_{\mathcal{B}} \tag{D.3}$$

Definition D.1.9 (C^k-diffeomorphism)**.** Let $A \subseteq \mathbb{R}^n$ and $B \subseteq \mathbb{R}^m$. A map $f : A \to B$ is said to be a C^k-diffeomorphism if it is bijective, C^2 and has a C^2 inverse.

Remark D.1.10. Differentiability is typically defined for functions that have an open domain in $\mathbb{R}^n$. However, in the definition above, the set A might not be open in $\mathbb{R}^n$ and B might not be open in $\mathbb{R}^m$. In the case of an arbitrary domain A, it is customary to say that a function $f : A \subseteq \mathbb{R}^n \to \mathbb{R}^m$ is C^k if there exists a C^k function g defined on an open set $U \subseteq \mathbb{R}^n$ that contains A such that $g\big|_A = f$ (i.e. g extends f). With this definition, we have that a composition of C^k functions is C^k, as usual. See for example p.199 of Munkres [1991].

The following lemma allows us to unambiguously define the k first derivatives of a C^k function $f : A \to \mathbb{R}^m$ on the set $\overline{A^\circ}$.

Lemma D.1.11. *Let $A \subset \mathbb{R}^n$ and $f : A \to \mathbb{R}^m$ be a C^k function. Then, its k first derivatives is uniquely defined on $\overline{A^\circ}$ in the sense that they do not depend on the specific choice of C^k extension.*

Proof. Let $g : U \to \mathbb{R}^n$ and $h : V \to \mathbb{R}^n$ be two C^k extensions of f to $U \subset \mathbb{R}^n$ and $V \subset \mathbb{R}^n$ both open in $\mathbb{R}^n$. By definition,

$$g(x) = f(x) = h(x),\ \forall x \in A\,. \tag{D.4}$$

The usual derivative is uniquely defined on the interior of the domain, so that

$$Dg(x) = Df(x) = Dh(x),\ \forall x \in A^\circ\,. \tag{D.5}$$

Consider a point $x_0 \in \overline{A^\circ}$. By definition of closure, there exists a sequence $\{x_k\}_{k=1}^\infty \subset A^\circ$ s.t. $\lim_{k\to\infty} x_k = x_0$. We thus have that

$$\lim_{k\to\infty} Dg(x_k) = \lim_{k\to\infty} Dh(x_k) \tag{D.6}$$

$$Dg(x_0) = Dh(x_0)\,, \tag{D.7}$$

where we used the fact that the derivatives of g and h are continuous to go to the second line. Thus, all the C^k extensions of f must have equal derivatives on $\overline{A^\circ}$. This means we can unambiguously define the derivative of f everywhere on $\overline{A^\circ}$ to be equal to the derivative of one of its C^k extensions.

Since f is C^k, its derivative Df is C^{k-1}, we can thus apply the same argument to get that the second derivative of f is uniquely defined on $\overline{\overline{A^\circ}^\circ}$. It can be shown that $\overline{\overline{A^\circ}^\circ} = \overline{A^\circ}$. One can thus apply the same argument recursively to show that the first k derivatives of f are uniquely defined on $\overline{A^\circ}$. $\square$

Definition D.1.12 (C^k-diffeomorphism onto its image)**.** Let $A \subseteq \mathbb{R}^n$. A map $f : A \to \mathbb{R}^m$ is said to be a C^k-diffeomorphism onto its image if the restriction f to its image $\tilde{f} : A \to f(A)$ is a C^k-diffeomorphism.

Remark D.1.13. If $S \subseteq A \subseteq \mathbb{R}^n$ and $f : A \to \mathbb{R}^m$ is a C^k-diffeomorphism on its image, then the restriction of f to S, i.e. $f\big|_S$, is also a C^k diffeomorphism on its image. That is because $f\big|_S$ is clearly bijective, is C^k (simply take the C^k extension of f) and so is its inverse (simply take the C^k extension of f^{-1}).

D.1.2 Proof of Theorem 3.4.4: Local Disentanglement

Theorem 3.4.4 (Local disentanglement via additive decoders)**.** *Given the data generation process (Assumption 3.3.1) and the optimal solution $(\hat{f}, \hat{g})$ under the reconstruction loss, along with the extra assumptions stated below:*

- *Both true decoder g and learned $\hat{g}$ are additive functions (Definition 3.3.2)*
- *Learned decoder $\hat{g}$ is a C^2-diffeomorphism, the learned encoder $\hat{f}$ is continuous*
- *True deoder g is sufficiently nonlinear as formalized by Assumption 3.4.5*

Then we have that $\hat{g}$ is locally $\mathcal{B}$-disentangled w.r.t. g (Definition 3.4.3) .

Proof. Note that under the optimal solution to reconstruction objective (Appendix B.3), we can define the map $v := g^{-1} \circ \hat{g}$, which is a C^2-diffeomorphism from $\hat{\mathcal{Z}}^{\text{train}}$ to $\mathcal{Z}^{\text{train}}$. This allows one to write

$$g \circ v(z) = \hat{g}(z) \; \forall z \in \hat{\mathcal{Z}}^{\text{train}} \tag{D.8}$$

$$\sum_{B \in \mathcal{B}} g^{(B)}(v_B(z)) = \sum_{B \in \mathcal{B}} \hat{g}^{(B)}(z_B) \; \forall z \in \hat{\mathcal{Z}}^{\text{train}} . \tag{D.9}$$

Since $\mathcal{Z}^{\text{train}}$ is regularly closed and is diffeomorphic to $\hat{\mathcal{Z}}^{\text{train}}$, by Lemma D.1.5, we must have that $\hat{\mathcal{Z}}^{\text{train}} \subset \overline{(\hat{\mathcal{Z}}^{\text{train}})^\circ}$. Moreover, the left and right hand side of (D.9) are C^2, which means they have uniquely defined first and second derivatives on $\overline{(\hat{\mathcal{Z}}^{\text{train}})^\circ}$ by Lemma D.1.11. This means the derivatives are uniquely defined on $\hat{\mathcal{Z}}^{\text{train}}$.

Let $z \in \hat{\mathcal{Z}}^{\text{train}}$. Choose some $J \in \mathcal{B}$ and some $j \in J$. Differentiate both sides of the above equation with respect to z_j, which yields:

$$\sum_{B \in \mathcal{B}} \sum_{i \in B} D_i g^{(B)}(v_B(z)) D_j v_i(z) = D_j \hat{g}^{(J)}(z_J) . \tag{D.10}$$

Choose $J' \in \mathcal{B} \setminus \{J\}$ and $j' \in J'$. Differentiating the above w.r.t. $z_{j'}$ yields

$$\begin{aligned}
&\sum_{B \in \mathcal{B}} \sum_{i \in B} \left[D_i g^{(B)}(v_B(z)) D^2_{j,j'} v_i(z) + \sum_{i' \in B} D^2_{i,i'} g^{(B)}(v_B(z)) D_{j'} v_{i'}(z) D_j v_i(z) \right] = 0 \\
&\sum_{B \in \mathcal{B}} \left[\sum_{i \in B} \left[D_i g^{(B)}(v_B(z)) D^2_{j,j'} v_i(z) + D^2_{i,i} g^{(B)}(v_B(z)) D_{j'} v_i(z) D_j v_i(z) \right] + \right. \\
&\qquad \left. \sum_{(i,i') \in B^2_<} D^2_{i,i'} g^{(B)}(v_B(z)) (D_{j'} v_{i'}(z) D_j v_i(z) + D_{j'} v_i(z) D_j v_{i'}(z)) \right] = 0 \, ,
\end{aligned} \tag{D.11}$$

where $B^2_< := B^2 \cap \{(i, i') \mid i' < i\}$. For the sake of notational conciseness, we are going to refer to $S_\mathcal{B}$ and $S^c_\mathcal{B}$ as S and S^c (Definition D.1.8). Also, define

$$S_< := \bigcup_{B \in \mathcal{B}} B^2_< . \tag{D.12}$$

Let us define the vectors

$$\forall i \in \{1, ...d_z\},\ \vec{a}_i(z) := (D^2_{j,j'} v_i(z))_{(j,j') \in S^c} \tag{D.13}$$

$$\forall i \in \{1, ...d_z\},\ \vec{b}_i(z) := (D_{j'} v_i(z) D_j v_i(z))_{(j,j') \in S^c} \tag{D.14}$$

$$\forall B \in \mathcal{B},\ \forall (i, i') \in B^2_<,\ \vec{c}_{i,i'}(z) := (D_{j'} v_{i'}(z) D_j v_i(z) + D_{j'} v_i(z) D_j v_{i'}(z))_{(j,j') \in S^c} \tag{D.15}$$

This allows us to rewrite, for all $k \in \{1, ..., d_x\}$

$$\sum_{B \in \mathcal{B}} \left[\sum_{i \in B} \left[D_i g^{(B)}_k (v_B(z)) \vec{a}_i(z) + D^2_{i,i} g^{(B)}_k (v_B(z)) \vec{b}_i(z) \right] + \sum_{(i,i') \in B^2_<} D^2_{i,i'} g^{(B)}_k (v_B(z)) \vec{c}_{i,i'}(z) \right] = 0\,. \tag{D.16}$$

We define

$$w(z, k) := ((D_i g^{(B)}_k (z_B))_{i \in B}, (D^2_{i,i} g^{(B)}_k (z_B))_{i \in B}, (D^2_{i,i'} g^{(B)}_k (z_B))_{(i,i') \in B^2_<})_{B \in \mathcal{B}} \tag{D.17}$$

$$M(z) := [[\vec{a}_i(z)]_{i \in B}, [\vec{b}_i(z)]_{i \in B}, [\vec{c}_{i,i'}(z)]_{(i,i') \in B^2_<}]_{B \in \mathcal{B}}\,, \tag{D.18}$$

which allows us to write, for all $k \in \{1, ..., d_z\}$

$$M(z) w(v(z), k) = 0\,. \tag{D.19}$$

We can now recognize that the matrix $W(v(z))$ of Assumption 3.4.5 is given by

$$W(v(z))^\top = [w(v(z), 1)\ \dots\ w(v(z), d_x)] \tag{D.20}$$

which allows us to write

$$M(z) W(v(z))^\top = 0 \tag{D.21}$$

$$W(v(z)) M(z)^\top = 0 \tag{D.22}$$

Since $W(v(z))$ has full column-rank (by Assumption 3.4.5 and the fact that $v(z) \in \mathcal{Z}^{\text{train}}$), there exists q rows that are linearly independent. Let K be the index set of these rows. This means $W(v(z))_{K,\cdot}$ is an invertible matrix. We can thus write

$$W(v(z))_{K,\cdot} M(z)^\top = 0 \tag{D.23}$$

$$(W(v(z))_{K,\cdot})^{-1} W(v(z))_{K,\cdot} M(z)^\top = (W(v(z))_{K,\cdot})^{-1} 0 \tag{D.24}$$

$$M(z)^\top = 0\,, \tag{D.25}$$

which means, in particular, that, $\forall i \in \{1, \dots, d_z\}$, $\vec{b}_i(z) = 0$, i.e.,

$$\forall i \in \{1, \dots, d_z\}, \forall (j, j') \in S^c, D_j v_i(z) D_{j'} v_i(z) = 0 \tag{D.26}$$

Since the v is a diffeomorphism, its Jacobian matrix $Dv(z)$ is invertible everywhere. By Lemma D.1.6, this means there exists a permutation π such that, for all j, $D_j v_{\pi(j)}(z) \neq 0$. This and (D.26) imply that

$$\forall (j, j') \in S^c, \quad D_j v_{\pi(j')}(z) \underbrace{D_{j'} v_{\pi(j')}(z)}_{\neq 0} = 0, \tag{D.27}$$

$$\implies \forall (j, j') \in S^c, \quad D_j v_{\pi(j')}(z) = 0\,. \tag{D.28}$$

To show that $Dv(z)$ is a $\mathcal{B}$-block permutation matrix, the only thing left to show is that π respects $\mathcal{B}$. For this, we use the fact that, $\forall B \in \mathcal{B}, \forall (i, i') \in B^2_<$, $\vec{c}_{i,i'}(z) = 0$ (recall $M(z) = 0$). Because $\vec{c}_{i,i'}(z) = \vec{c}_{i',i}(z)$, we can write

$$\forall (i, i') \in S \text{ s.t. } i \neq i', \forall (j, j') \in S^c, D_{j'} v_{i'}(z) D_j v_i(z) + D_{j'} v_i(z) D_j v_{i'}(z) = 0\,. \tag{D.29}$$

We now show that if $(j, j') \in S^c$ (indices belong to different blocks), then $(\pi(j), \pi(j')) \in S^c$ (they also belong to different blocks). Assume this is false, i.e. there exists $(j_0, j'_0) \in S^c$ such that $(\pi(j_0), \pi(j'_0)) \in S$. Then we can apply (D.29) (with $i := \pi(j_0)$ and $i' := \pi(j'_0)$) and get

$$\underbrace{D_{j'_0} v_{\pi(j'_0)}(z) D_{j_0} v_{\pi(j_0)}(z)}_{\neq 0} + D_{j'_0} v_{\pi(j_0)}(z) D_{j_0} v_{\pi(j'_0)}(z) = 0\,, \tag{D.30}$$

where the left term in the sum is different of 0 because of the definition of π. This implies that

$$D_{j'_0} v_{\pi(j_0)}(z) D_{j_0} v_{\pi(j'_0)}(z) \neq 0\,, \tag{D.31}$$

otherwise (D.30) cannot hold. But (D.31) contradicts (D.28). Thus, we have that,

$$(j, j') \in S^c \implies (\pi(j), \pi(j')) \in S^c\,. \tag{D.32}$$

The contraposed is

$$(\pi(j), \pi(j')) \in S \implies (j, j') \in S \tag{D.33}$$

$$(j, j') \in S \implies (\pi^{-1}(j), \pi^{-1}(j')) \in S\,. \tag{D.34}$$

From the above, it is clear that π^{-1} respects $\mathcal{B}$ which implies that π respects $\mathcal{B}$ (Lemma D.1.14). Thus $Dv(z)$ is a $\mathcal{B}$-block permutation matrix. □

Lemma D.1.14 ($\mathcal{B}$-respecting permutations form a group)**.** *Let $\mathcal{B}$ be a partition of $\{1, \dots, d_z\}$ and let π and $\bar{\pi}$ be a permutation of $\{1, \dots, d_z\}$ that respect $\mathcal{B}$. The following holds:*

1. *The identity permutation e respects $\mathcal{B}$.*

2. *The composition $\pi \circ \bar{\pi}$ respects $\mathcal{B}$.*

3. *The inverse permutation π^{-1} respects $\mathcal{B}$.*

Proof. The first statement is trivial, since for all $B \in \mathcal{B}$, $e(B) = B \in \mathcal{B}$.

The second statement follows since for all $B \in \mathcal{B}$, $\bar{\pi}(B) \in \mathcal{B}$ and thus $\pi(\bar{\pi}(B)) \in \mathcal{B}$.

We now prove the third statement. Let $B \in \mathcal{B}$. Since π is surjective and respects $\mathcal{B}$, there exists a $B' \in \mathcal{B}$ such that $\pi(B') = B$. Thus, $\pi^{-1}(B) = \pi^{-1}(\pi(B')) = B' \in \mathcal{B}$. □

D.1.3 Sufficient nonlinearity v.s. sufficient variability in nonlinear ICA with auxiliary variables

In Section 3.4.1, we introduced the “sufficient nonlinearity” condition (Assumption 3.4.5) and highlighted its resemblance to the “sufficient variability” assumptions often found in the nonlinear ICA literature [Hyvärinen and Morioka, 2016, 2017, Hyvärinen et al., 2019, Khemakhem et al., 2020b,c, Lachapelle et al., 2022b, Zheng et al., 2022]. We now clarify this connection. To make the discussion more concrete, we consider the sufficient variability assumption found in Hyvärinen et al. [2019]. In this work, the latent variable $\boldsymbol{z}$ is assumed to be distributed according to

$$p(z \mid u) := \prod_{i=1}^{d_z} p_i(z_i \mid u) \tag{D.35}$$

In other words, the latent factors z_i are mutually conditionally independent given an observed auxiliary variable u. Define

$$w(z, u) := \left(\left(\frac{\partial}{\partial z_i} \log p_i(z_i \mid u) \right)_{i \in [d_z]} \left(\frac{\partial^2}{\partial z_i^2} \log p_i(z_i \mid u) \right)_{i \in [d_z]} \right) \in \mathbb{R}^{2d_z} \tag{D.36}$$

We now recall the assumption of sufficient variability of Hyvärinen et al. [2019]:

Assumption D.1.15 (Assumption of variability from Hyvärinen et al. [2019, Theorem 1])**.** For any $z \in \mathbb{R}^{d_z}$, there exists $2d_z + 1$ values of u, denoted by $u^{(0)}, u^{(1)}, \dots, u^{(2d_z)}$ such that the $2d_z$ vectors

$$w(z, u^{(1)}) - w(z, u^{(0)}), \dots, w(z, u^{(2d_z)}) - w(z, u^{(0)}) \tag{D.37}$$

are linearly independent.

To emphasize the resemblance with our assumption of sufficient nonlinearity, we rewrite it in the special case where the partition $\mathcal{B} := \{\{1\}, \dots, \{d_z\}\}$. Note that, in that case, $q := d_z + \sum_{B \in \mathcal{B}} \frac{|B|(|B|+1)}{2} = 2d_z$.

Assumption D.1.16 (Sufficient nonlinearity (trivial partition))**.** For all $z \in \mathcal{Z}^{\text{train}}$, g is such that the following matrix has independent columns (i.e. full column-rank):

$$W(z) := \left[\left[D_i g^{(i)}(z_i) \right]_{i \in [d_z]} \ \left[D^2_{i,i} g^{(i)}(z_i) \right]_{i \in [d_z]} \right] \in \mathbb{R}^{d_x \times 2d_z} \tag{D.38}$$

One can already see the resemblance between Assumptions D.1.15 & D.1.16, e.g. both have something to do with first and second derivatives. To make the connection even more explicit, define $w(z, k)$ to be the kth row of $W(z)$ (do not conflate with $w(z, u)$). Also, recall the basic fact from linear algebra that the column-rank is always equal to the row-rank. This means that $W(z)$ is full column-rank if and only if there exists k_1, ..., $k_{2d_z} \in [d_x]$ such that the vectors $w(z, k_1), \dots, w(z, k_{2d_z})$ are linearly independent. It is then easy to see the correspondance between $w(z, k)$ and $w(z, u) - w(z, u^{(0)})$ (from Assumption D.1.15) and between the pixel index $k \in [d_x]$ and the auxiliary variable u.

We now look at why Assumption 3.4.5 is likely to be satisfied when $d_x >> d_z$. Informally, one can see that when d_x is much larger than $2d_z$, the matrix $W(z)$ has much more rows than columns and thus it becomes more likely that we will find $2d_z$ rows that are linearly independent, thus satisfying Assumption 3.4.5.

To further highlight the importance of this assumption, we study the following example that shows why Theorem 3.4.4 does not apply if the ground-truth decoder g is linear. If that was the case, it would contradict the well known fact that linear ICA with independent Gaussian factors is unidentifiable.

Example D.1.17 (Importance of Assumption 3.4.5)**.** Suppose $x = g(z) = Az$ where $A \in \mathbb{R}^{d_x \times d_z}$ is full rank. Take $\hat{g}(z) := AVz$ and $\hat{f}(x) := V^{-1}A^{\dagger}x$ where $V \in \mathbb{R}^{d_z \times d_z}$ is invertible and $A^{\dagger}$ is the left pseudo inverse of A. By construction, we have that $\mathbb{E}[x - \hat{g}(\hat{f}(x))] = 0$ and g and $\hat{g}$ are $\mathcal{B}$-additive because $g(z) = \sum_{B \in \mathcal{B}} A_{\cdot,B} z_B$ and $\hat{g}(z) = \sum_{B \in \mathcal{B}} (AV)_{\cdot,B} z_B$. However, we still have that $v(z) := g^{-1} \circ \hat{g}(z) = Vz$ where V does not necessarily have a block-permutation structure, i.e. no disentanglement. The reason we cannot apply Theorem 3.4.4 here is because Assumption 3.4.5 is not satisfied. Indeed, the second derivatives of $g^{(B)}(z_B) := A_{\cdot,B} z_B$ are all zero and hence $W(z)$ cannot have full column-rank.

D.1.4 Proof of Theorem 3.4.6: Global Disentanglement

We start with a simple definition:

Definition D.1.18 ($\mathcal{B}$-block permutation matrices)**.** A matrix $\mathcal{A} \in \mathbb{R}^{d \times d}$ is a $\mathcal{B}$*-block permutation matrix* if it is invertible and can be written as $A = CP_{\pi}$ where P_{π} is the matrix representing the $\mathcal{B}$-respecting permutation π ($P_{\pi} e_i = e_{\pi(i)}$) and $C \in \mathbb{R}^{d \times d}_{S_{\mathcal{B}}}$ (See Definitions D.1.7 & D.1.8).

The following technical lemma leverages continuity and path-connectedness to show that the block-permutation structure must remain the same across the whole domain. It can be skipped at first read.

Lemma D.1.19. *Let $\mathcal{C}$ be a connected topological space and let $M : \mathcal{C} \to \mathbb{R}^{d \times d}$ be a continuous function. Suppose that, for all $c \in \mathcal{C}$, $M(c)$ is an invertible $\mathcal{B}$-block permutation matrix (Definition D.1.18). Then, there exists a $\mathcal{B}$-respecting permutation π such that for all $c \in \mathcal{C}$ and all distinct $B, B' \in \mathcal{B}$, $M(c)_{\pi(B'),B} = 0$.*

Proof. The reason this result is not trivial, is that, even if $M(c)$ is a $\mathcal{B}$-block permutation for all c, the permutation might change for different c. The goal of this lemma is to show that, if $\mathcal{C}$ is connected and the map $M(\cdot)$ is continuous, then one can find a single permutation that works for all $c \in \mathcal{C}$.

First, since $\mathcal{C}$ is connected and M is continuous, its image, $M(\mathcal{C})$, must be connected (by [Munkres, 2000, Theorem 23.5]).

Second, from the hypothesis of the lemma, we know that

$$M(\mathcal{C}) \subset \mathcal{A} := \left(\bigcup_{\pi \in \mathfrak{S}(\mathcal{B})} \mathbb{R}^{d \times d}_{S_{\mathcal{B}}} P_{\pi} \right) \setminus \{\text{singular matrices}\}\,, \tag{D.39}$$

where $\mathfrak{S}(\mathcal{B})$ is the set of $\mathcal{B}$-respecting permutations and $\mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi = \{MP_\pi \mid M \in \mathbb{R}^{d\times d}_{S_\mathcal{B}}\}$. We can rewrite the set $\mathcal{A}$ above as

$$\mathcal{A} = \bigcup_{\pi \in \mathfrak{S}(\mathcal{B})} \left(\mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi \setminus \{\text{singular matrices}\}\right) , \tag{D.40}$$

We now define an equivalence relation $\sim$ over $\mathcal{B}$-respecting permutation: $\pi \sim \pi'$ iff for all $B \in \mathcal{B}$, $\pi(B) = \pi'(B)$. In other words, two $\mathcal{B}$-respecting permutations are equivalent if they send every block to the same block (note that they can permute elements of a given block differently). We notice that

$$\pi \sim \pi' \implies \mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi = \mathbb{R}^{d\times d}_{S_\mathcal{B}} P_{\pi'} \,. \tag{D.41}$$

Let $\mathfrak{S}(\mathcal{B})/\sim$ be the set of equivalence classes induce by $\sim$ and let Π stand for one such equivalence class. Thanks to (D.41), we can define, for all $\Pi \in \mathfrak{S}(\mathcal{B})/\sim$, the following set:

$$V_\Pi := \mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi \setminus \{\text{singular matrices}\}, \text{ for some } \pi \in \Pi \,, \tag{D.42}$$

where the specific choice of $\pi \in \Pi$ is arbitrary (any $\pi' \in \Pi$ would yield the same definition, by (D.41)). This construction allows us to write

$$\mathcal{A} = \bigcup_{\Pi \in \mathfrak{S}(\mathcal{B})/\sim} V_\Pi \,, \tag{D.43}$$

We now show that $\{V_\Pi\}_{\Pi \in \mathfrak{S}(\mathcal{B})/\sim}$ forms a partition of $\mathcal{A}$. Choose two distinct equivalence classes of permutations Π and Π' and let $\pi \in \Pi$ and $\pi' \in \Pi'$ be representatives. We note that

$$\mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi \cap \mathbb{R}^{d\times d}_{S_\mathcal{B}} P_{\pi'} \subset \{\text{singular matrices}\} \,, \tag{D.44}$$

since any matrix that is both in $\mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi$ and $\mathbb{R}^{d\times d}_{S_\mathcal{B}} P_{\pi'}$ must have at least one row filled with zeros. This implies that

$$V_\Pi \cap V_{\Pi'} = \emptyset \,, \tag{D.45}$$

which shows that $\{V_\Pi\}_{\Pi \in \mathfrak{S}(\mathcal{B})/\sim}$ is indeed a partition of $\mathcal{A}$.

Each V_Π is closed in $\mathcal{A}$ (wrt the relative topology) since

$$V_\Pi = \mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi \setminus \{\text{singular matrices}\} = \mathcal{A} \cap \underbrace{\mathbb{R}^{d\times d}_{S_\mathcal{B}} P_\pi}_{\text{closed in } \mathbb{R}^{d\times d}} \,. \tag{D.46}$$

Moreover, V_Π is open in $\mathcal{A}$, since

$$V_\Pi = \mathcal{A} \setminus \underbrace{\bigcup_{\Pi' \neq \Pi} V_{\Pi'}}_{\text{closed in } \mathcal{A}} . \tag{D.47}$$

Thus, for any $\Pi \in \mathfrak{S}(\mathcal{B})/\sim$, the sets V_Π and $\bigcup_{\Pi' \neq \Pi} V_{\Pi'}$ forms a *separation* (see [Munkres, 2000, Section 23]). Since $M(\mathcal{C})$ is a connected subset of $\mathcal{A}$, it must lie completely in V_Π or $\bigcup_{\Pi' \neq \Pi} V_{\Pi'}$, by [Munkres, 2000, Lemma 23.2]. Since this is true for all Π, it must follow that there exists a Π^* such that $M(\mathcal{C}) \subset V_{\Pi^*}$, which completes the proof. □

Theorem 3.4.6 (Global disentanglement via additive decoders)**.** *Suppose that all the assumptions of Theorem 3.4.4 hold, along with the extra assumptions stated below:*

- *Support of the true latents $\mathcal{Z}^{\text{train}}$ is path-connected (Definition D.1.3)*
- *Block-specific decoders $g^{(B)}$ and $\hat{g}^{(B)}$ are injective for all blocks $B \in \mathcal{B}$*

Then for optimal solution $(\hat{f}, \hat{g})$ under the reconstruction loss we have $\hat{g}$ is (globally) $\mathcal{B}$-disentangled w.r.t. g (Definition 3.4.2). Further, for all $B \in \mathcal{B}$, we have the following:

$$\hat{g}^{(B)}(z_B) = g^{(\pi(B))}(\bar{v}_{\pi(B)}(z_B)) + c^{(B)}, \text{ for all } z_B \in \hat{\mathcal{Z}}^{\text{train}}_B , \tag{3.7}$$

where the functions $\bar{v}_{\pi(B)}$ are from Definition 3.4.2 and the vectors $c^{(B)} \in \mathbb{R}^{d_x}$ are constants such that $\sum_{B \in \mathcal{B}} c^{(B)} = 0$. We also have that the functions $\bar{v}_{\pi(B)} : \hat{\mathcal{Z}}^{\text{train}}_B \to \mathcal{Z}^{\text{train}}_{\pi(B)}$ are C^2-diffeomorphisms and have the following form:

$$\bar{v}_{\pi(B)}(z_B) = (g^{\pi(B)})^{-1}(\hat{g}^{(B)}(z_B) - c^{(B)}), \text{ for all } z_B \in \hat{\mathcal{Z}}^{\text{train}}_B . \tag{3.8}$$

Proof. **Step 1 - Showing the permutation π does not change for different z.** Theorem 3.4.4 showed local $\mathcal{B}$-disentanglement, i.e. for all $z \in \hat{\mathcal{Z}}^{\text{train}}$, $Dv(z)$ has a $\mathcal{B}$-block permutation structure. The first step towards showing global disentanglement is to show that this block structure is the same for all $z \in \hat{\mathcal{Z}}^{\text{train}}$ (*a priori*, π could be different for different z). Since v is C^2, its Jacobian $Dv(z)$ is continuous. Since $\mathcal{Z}^{\text{train}}$ is path-connected, $\hat{\mathcal{Z}}^{\text{train}}$ must also be since both sets are diffeomorphic. By Lemma D.1.19, this means the $\mathcal{B}$-block permutation structure of $Dv(z)$ is the same for all $z \in \hat{\mathcal{Z}}^{\text{train}}$ (implicitly using the fact that path-connected implies connected). In other words, there exists a permutation π respecting $\mathcal{B}$ such that, for all $z \in \hat{\mathcal{Z}}^{\text{train}}$ and all distinct $B, B' \in \mathcal{B}$, $D_B v_{\pi(B')}(z) = 0$.

Step 2 - Linking object-specific decoders. We now show that, for all $B \in \mathcal{B}$, $\hat{g}^{(B)}(z_B) = g^{(\pi(B))}(v_{\pi(B)}(z)) + \boldsymbol{c}^{(B)}$ for all $z \in \hat{\mathcal{Z}}^{\text{train}}$. To do this, we rewrite (D.10) as

$$D\hat{g}^{(J)}(z_J) = \sum_{B \in \mathcal{B}} Dg^{(B)}(v_B(z)) D_J v_B(z)\,, \tag{D.48}$$

but because $B \neq \pi(J) \implies D_J v_B(z) = 0$ (block-permutation structure), we get

$$D\hat{g}^{(J)}(z_J) = Dg^{(\pi(J))}(v_{\pi(J)}(z)) D_J v_{\pi(J)}(z)\,. \tag{D.49}$$

The above holds for all $J \in \mathcal{B}$. We simply change J by B in the following equation.

$$D\hat{g}^{(B)}(z_B) = Dg^{(\pi(B))}(v_{\pi(B)}(z)) D_B v_{\pi(B)}(z)\,. \tag{D.50}$$

Now notice that the r.h.s. of the above equation is equal to $D(g^{(\pi(B))} \circ v_{\pi(B)})$. We can thus write

$$D\hat{g}^{(B)}(z_B) = D(g^{(\pi(B))} \circ v_{\pi(B)})(z)\,, \text{for all } z \in \hat{\mathcal{Z}}^{\text{train}}\,. \tag{D.51}$$

Now choose distinct $z, z^0 \in \hat{\mathcal{Z}}^{\text{train}}$. Since $\mathcal{Z}^{\text{train}}$ is path-connected, $\hat{\mathcal{Z}}^{\text{train}}$ also is since they are diffeomorphic. Hence, there exists a continuously differentiable function $\boldsymbol{\phi} : [0, 1] \to \hat{\mathcal{Z}}^{\text{train}}$ such that $\boldsymbol{\phi}(0) = z^0$ and $\boldsymbol{\phi}(1) = z$. We can now use (D.51) together with the gradient theorem, a.k.a. the fundamental theorem of calculus for line integrals, to show the following

$$\int_0^1 D\hat{g}^{(B)}(\boldsymbol{\phi}_B(z)) \cdot \boldsymbol{\phi}_B(t)dt = \int_0^1 D(g^{(\pi(B))} \circ v_{\pi(B)})(\boldsymbol{\phi}(z)) \cdot \boldsymbol{\phi}(t)dt \tag{D.52}$$

$$\hat{g}^{(B)}(z_B) - \hat{g}^{(B)}(z_B^0) = g^{(\pi(B))} \circ v_{\pi(B)}(z) - g^{(\pi(B))} \circ v_{\pi(B)}(z^0) \tag{D.53}$$

$$\hat{g}^{(B)}(z_B) = g^{(\pi(B))} \circ v_{\pi(B)}(z) + \underbrace{(\hat{g}^{(B)}(z_B^0) - g^{(\pi(B))} \circ v_{\pi(B)}(z^0))}_{\text{constant in } z} \tag{D.54}$$

$$\hat{g}^{(B)}(z_B) = g^{(\pi(B))} \circ v_{\pi(B)}(z) + c^{(B)}\,, \tag{D.55}$$

which holds for all $z \in \hat{\mathcal{Z}}^{\text{train}}$.

We now show that $\sum_{B \in \mathcal{B}} c^{(B)} = 0$. Take some $z^0 \in \hat{\mathcal{Z}}^{\text{train}}$. Equations (D.9) & (D.55) tell us that

$$\sum_{B\in\mathcal{B}} g^{(B)}(v_B(z^0)) = \sum_{B\in\mathcal{B}} \hat{g}^{(B)}(z^0_B) \tag{D.56}$$
$$= \sum_{B\in\mathcal{B}} g^{(\pi(B))}(v_{\pi(B)}(z^0)) + \sum_{B\in\mathcal{B}} c^{(B)} \tag{D.57}$$
$$= \sum_{B\in\mathcal{B}} g^{(B)}(v_B(z^0)) + \sum_{B\in\mathcal{B}} c^{(B)} \tag{D.58}$$
$$\implies \sum_{B\in\mathcal{B}} c^{(B)} = 0 \tag{D.59}$$

Step 3 - From local to global disentanglement. By assumption, the functions $g^{(B)} : \mathcal{Z}^{\text{train}}_B \to \mathbb{R}^{d_x}$ are injective. This will allow us to show that $v_{\pi(B)}(z)$ depends only on z_B. We proceed by contradiction. Suppose there exists $(z_B, z_{B^c}) \in \hat{\mathcal{Z}}^{\text{train}}$ and $z^0_{B^c}$ such that $(z_B, z^0_{B^c}) \in \hat{\mathcal{Z}}^{\text{train}}$ and $v_{\pi(B)}(z_B, z_{B^c}) \neq v_{\pi(B)}(z_B, z^0_{B^c})$. This means

$$g^{(\pi(B))} \circ v_{\pi(B)}(z_B, z_{B^c}) + c^{(B)} = \hat{g}^{(B)}(z_B) = g^{(\pi(B))} \circ v_{\pi(B)}(z_B, z^0_{B^c}) + c^{(B)}$$

$$g^{(\pi(B))}(v_{\pi(B)}(z_B, z_B)) = g^{(\pi(B))}(v_{\pi(B)}(z_B, z^0_B))$$

which is a contradiction with the fact that $g^{(\pi(B))}$ is injective. Hence, $v_{\pi(B)}(z)$ depends only on z_B. We also get an explicit form for $v_{\pi(B)}$:

$$(g^{\pi(B)})^{-1}(\hat{g}^{(B)}(z_B) - c^{(B)}) = v_{\pi(B)}(z) \text{ for all } z \in \mathcal{Z}^{\text{train}}\,. \tag{D.60}$$

We define the map $\bar{v}_{\pi(B)}(z_B) := (g^{\pi(B)})^{-1}(\hat{g}^{(B)}(z_B) - \boldsymbol{c}^{(B)})$ which is from $\hat{\mathcal{Z}}^{\text{train}}_B$ to $\mathcal{Z}^{\text{train}}_{\pi(B)}$. This allows us to rewrite (D.55) as

$$\hat{g}^{(B)}(z_B) = g^{(\pi(B))} \circ \bar{v}_{\pi(B)}(z_B) + c^{(B)}\,,\ \text{for all } z_B \in \mathcal{Z}^{\text{train}}_B\,. \tag{D.61}$$

Because $\hat{g}^{(B)}$ is also injective, we must have that $\bar{v}_{\pi(B)} : \hat{\mathcal{Z}}^{\text{train}}_B \to \mathcal{Z}^{\text{train}}_{\pi(B)}$ is injective as well.

We now show that $\bar{v}_{\pi(B)}$ is surjective. Choose some $z_{\pi(B)} \in \mathcal{Z}^{\text{train}}_{\pi(B)}$. We can always find $z_{\pi(B)^c}$ such that $(z_{\pi(B)}, z_{\pi(B)^c}) \in \mathcal{Z}^{\text{train}}$. Because $v : \hat{\mathcal{Z}}^{\text{train}} \to \mathcal{Z}^{\text{train}}$ is surjective (it is a diffeomorphism), there exists a $z^0 \in \hat{\mathcal{Z}}^{\text{train}}$ such that $v(z^0) = (z_{\pi(B)}, z_{\pi(B)^c})$. By (D.60), we have that

$$\bar{v}_{\pi(B)}(z^0_B) = v_{\pi(B)}(z^0)\,. \tag{D.62}$$

which means $\bar{v}_{\pi(B)}(z^0_B) = z_{\pi(B)}$.

We thus have that $\bar{v}_{\pi(B)}$ is bijective. It is a diffeomorphism because

$$\det D\bar{v}_{\pi(B)}(z_B) = \det D_B v_{\pi(B)}(z) \neq 0 \;\forall z \in \hat{\mathcal{Z}}^{\text{train}} \tag{D.63}$$

where the first equality holds by (D.60) and the second holds because $\boldsymbol{v}$ is a diffeomorphism and has block-permutation structure, which means it has a nonzero determinant everywhere on $\hat{\mathcal{Z}}^{\text{train}}$ and is equal to the product of the determinants of its blocks, which implies each block $D_B v_{\pi(B)}$ must have nonzero determinant everywhere.

Since $\bar{v}_{\pi(B)} : \hat{\mathcal{Z}}^{\text{train}}_B \to \mathcal{Z}^{\text{train}}_{\pi(B)}$ bijective and has invertible Jacobian everywhere, it must be a diffeomorphism. □

D.1.5 Injectivity of object-specific decoders v.s. injectivity of their sum

We want to explore the relationship between the injectivity of individual object-specific decoders $g^{(B)}$ and the injectivity of their sum, i.e. $\sum_{B\in\mathcal{B}} g^{(B)}$.

We first show the simple fact that having each $g^{(B)}$ injective is not sufficient to have $\sum_{B\in\mathcal{B}} g^{(B)}$ injective. Take $g^{(B)}(z_B) = W^{(B)} z_B$ where $W^{(B)} \in \mathbb{R}^{d_x \times |B|}$ has full column-rank for all $B \in \mathcal{B}$. We have that

$$\sum_{B\in\mathcal{B}} g^{(B)}(z_B) = \sum_{B\in\mathcal{B}} W^{(B)} z_B = [W^{(B_1)} \;\cdots\; W^{(B_\ell)}]z\,, \tag{D.64}$$

where it is clear that the matrix $[W^{(B_1)} \;\cdots\; W^{(B_\ell)}] \in \mathbb{R}^{d_x \times d_z}$ is not necessarily injective even if each $W^{(B)}$ is. This is the case, for instance, if all $W^{(B)}$ have the same image.

We now provide conditions such that $\sum_{B\in\mathcal{B}} g^{(B)}$ injective implies each $g^{(B)}$ injective. We start with a simple lemma:

Lemma D.1.20. *If $g \circ h$ is injective, then h is injective.*

Proof. By contradiction, assume that h is not injective. Then, there exists distinct $x_1, x_2 \in \text{Dom}(h)$ such that $h(x_1) = h(x_2)$. This implies $g \circ h(x_1) = g \circ h(x_2)$, which violates injectivity of $g \circ h$. □

The following Lemma provides a condition on the domain of the function $\sum_{B\in\mathcal{B}} g^{(B)}$, $\mathcal{Z}^{\text{train}}$, so that its injectivity implies injectivity of the functions $g^{(B)}$.

Lemma D.1.21. *Assume that, for all $B \in \mathcal{B}$ and for all distinct $z_B, z'_B \in \mathcal{Z}^{\text{train}}_B$, there exists z_{B^c} such that $(z_B, z_{B^c}), (z'_B, z_{B^c}) \in \mathcal{Z}^{\text{train}}$. Then, whenever $\sum_{B\in\mathcal{B}} g^{(B)}$ is injective, each $g^{(B)}$ must be injective.*

Proof. Notice that $g(z) := \sum_{B\in\mathcal{B}} g^{(B)}(z_B)$ can be written as $g := \text{SumBlocks} \circ \bar{g}(z)$ where

$$\bar{g}(z) := \begin{bmatrix} g^{(B_1)}(z_{B_1}) \\ \vdots \\ g^{(B_\ell)}(z_{B_\ell}) \end{bmatrix}, \text{ and } \text{SumBlocks}(x^{(B_1)}, \dots, x^{(B_\ell)}) := \sum_{B\in\mathcal{B}} x^{(B)} \tag{D.65}$$

Since g is injective, by Lemma D.1.20 $\bar{g}$ must be injective.

We now show that each $g^{(B)}$ must also be injective. Take $z_B, z'_B \in \mathcal{Z}^{\text{train}}_B$ such that $g^{(B)}(z_B) = g^{(B)}(z'_B)$. By assumption, we know there exists a z_{B^c} s.t. (z_B, z_{B^c}) and (z'_B, z_{B^c}) are in $\mathcal{Z}^{\text{train}}$. By construction, we have that $\bar{g}((z_B, z_{B^c})) = \bar{g}((z'_B, z_{B^c}))$. By injectivity of $\bar{g}$, we have that $(z_B, z_{B^c}) \neq (z'_B, z_{B^c})$, which implies $z_B \neq z'_B$, i.e. $g^{(B)}$ is injective. □

D.1.6 Proof of Corollary 3.4.8: Cartesian-Product Extrapolation

Corollary 3.4.8 (Cartesian-product extrapolation)**.** *Suppose all the assumptions of Theorem 3.4.6 hold. Then we have the following:*

$$\sum_{B\in\mathcal{B}} \hat{g}^{(B)}(z_B) = \sum_{B\in\mathcal{B}} g^{(\pi(B))}(\bar{v}_{\pi(B)}(z_B))\ \forall z \in \text{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\text{train}}) \tag{3.10}$$

Furthermore, if $\text{CPE}_{\mathcal{B}}(\mathcal{Z}^{\text{train}}) \subseteq \mathcal{Z}^{\text{test}}$*, then* $\hat{g}(\text{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\text{train}})) \subseteq g(\mathcal{Z}^{\text{test}})$*.*

Proof. Pick $z \in \text{CPE}(\hat{\mathcal{Z}}^{\text{train}})$. By definition, this means that, for all $B \in \mathcal{B}$, $z_B \in \hat{\mathcal{Z}}^{\text{train}}_B$. We thus have that, for all $B \in \mathcal{B}$,

$$\hat{g}^{(B)}(z_B) = g^{(\pi(B))} \circ \bar{v}_{\pi(B)}(z_B) + c^{(B)} \tag{D.66}$$

We can thus sum over B to obtain

$$\sum_{B\in\mathcal{B}} \hat{g}^{(B)}(z_B) = \sum_{B\in\mathcal{B}} g^{(\pi(B))} \circ \bar{v}_{\pi(B)}(z_B) + \underbrace{\sum_{B\in\mathcal{B}} c^{(B)}}_{=0} \tag{D.67}$$

Since $z \in \text{CPE}(\hat{\mathcal{Z}}^{\text{train}})$ was arbitrary, we have

$$\text{for all } z \in \text{CPE}(\hat{\mathcal{Z}}^{\text{train}}),\ \sum_{B\in\mathcal{B}} \hat{g}^{(B)}(z_B) = \sum_{B\in\mathcal{B}} g^{(\pi(B))} \circ \bar{v}_{\pi(B)}(z_B) \tag{D.68}$$

$$\hat{g}(z) = g \circ \bar{v}(z) \tag{D.69}$$

where $\bar{v} : \mathrm{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\mathrm{train}}) \to \mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}})$ is defined as

$$\bar{v}(z) := \begin{bmatrix} \bar{v}_{B_1}(z_{\pi^{-1}(B_1)}) \\ \vdots \\ \bar{v}_{B_\ell}(z_{\pi^{-1}(B_\ell)}) \end{bmatrix} \tag{D.70}$$

The map $\bar{v}$ is a diffeomorphism since each $\bar{v}_{\pi(B)}$ is a diffeomorphism from $\hat{\mathcal{Z}}^{\mathrm{train}}_B$ to $\mathcal{Z}^{\mathrm{train}}_{\pi(B)}$.

By (D.69) we get

$$\hat{g}(\mathrm{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\mathrm{train}})) = g \circ \bar{v}(\mathrm{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\mathrm{train}})) \tag{D.71}$$

and since the map $\bar{\boldsymbol{v}}$ is surjective we have $\bar{v}(\mathrm{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\mathrm{train}})) = \mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}})$ and thus

$$\hat{g}(\mathrm{CPE}_{\mathcal{B}}(\hat{\mathcal{Z}}^{\mathrm{train}})) = g(\mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}})) \tag{D.72}$$

Hence if $\mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}}) \subseteq \mathcal{Z}^{\mathrm{test}}$, then $g(\mathrm{CPE}_{\mathcal{B}}(\mathcal{Z}^{\mathrm{train}})) \subseteq g(\mathcal{Z}^{\mathrm{test}})$. □

D.2 Experiments

D.2.1 Training Details

Loss Function. We use the standard reconstruction objective of mean squared error loss between the ground truth data and the reconstructed/generated data.

Hyperparameters. For both the ScalarLatents and the BlockLatents dataset, we used the Adam optimizer with the hyperparameters defined below. Note that we maintain consistent hyperparameters across both the Additive decoder and the Non-Additive decoder method.

ScalarLatents Dataset.

- Batch Size: 64
- Learning Rate: 1×10^{-3}
- Weight Decay: 5×10^{-4}
- Total Epochs: 4000

BlockLatents Dataset.

- Batch Size: 1024
- Learning Rate: 1×10^{-3}
- Weight Decay: 5×10^{-4}
- Total Epochs: 6000

Model Architecture. We use the following architectures for Encoder and Decoder across both the datasets (ScalarLatents, BlockLatents). Note that for the ScalarLatents dataset we train with latent dimension $d_z = 2$, and for the BlockLatents dataset we train with latent dimension $d_z = 4$, which corresponds to the dimensionalities of the ground-truth data generating process for both datasets.

Encoder Architecture:

- RestNet-18 Architecture till the penultimate layer (512 dimensional feature output)
- Stack of 5 fully-connected layer blocks, with each block consisting of Linear Layer (dimensions: 512×512), Batch Normalization layer, and Leaky ReLU activation (negative slope: 0.01).
- Final Linear Layer (dimension: $512 \times d_z$) followed by Batch Normalization Layer to output the latent representation.

Decoder Architecture (Non-additive):

- Fully connected layer block with input as latent representation, consisting of Linear Layer (dimension: $d_z \times 512$), Batch Normalization layer, and Leaky ReLU activation (negative slope: 0.01).
- Stack of 5 fully-connected layer blocks, with each block consisting of Linear Layer (dimensions: 512×512), Batch Normalization layer, and Leaky ReLU activation (negative slope: 0.01).
- Series of DeConvolutional layers, where each DeConvolutional layer is follwed by Leaky ReLU (negative slope: 0.01) activation.
  - DeConvolution Layer (c_{in}: 64, c_{out}: 64, kernel: 4; stride: 2; padding: 1)
  - DeConvolution Layer (c_{in}: 64, c_{out}: 32, kernel: 4; stride: 2; padding: 1)
  - DeConvolution Layer (c_{in}: 32, c_{out}: 32, kernel: 4; stride: 2; padding: 1)
  - DeConvolution Layer (c_{in}: 32, c_{out}: 3, kernel: 4; stride: 2; padding: 1)

Decoder Architecture (Additive): Recall that an additive decoder has the form $g(z) = \sum_{B \in \mathcal{B}} g^{(B)}(z_B)$. Each $g^{(B)}$ has the same architecture as the one presented above for the non-additive case, but the input has dimensionality $|B|$ (which is 1 or 2, depending on the dataset). Note that we do not share parameters among the functions $g^{(B)}$.

D.2.2 Datasets Details

We use the moving balls environment from Ahuja et al. [2022] with images of dimension $64 \times 64 \times 3$, with latent vector (z) representing the position coordinates of each balls. We consider only two balls. The rendered images have pixels in the range [0, 255].

ScalarLatents Dataset. We fix the x-coordinate of each ball to 0.25 and 0.75. The only factors varying are the y-coordinates of both balls. Thus, $z \in \mathbb{R}^2$ and $\mathcal{B} = \{\{1\}, \{2\}\}$ where z_1 and z_2 designate the y-coordinates of both balls. We sample the y-coordinate of the first ball from a continuous uniform distribution as follows: $z_1 \sim$ Uniform(0, 1). Then we sample the y-coordinate of the second ball as per the following scheme:

$$z_2 \sim \begin{cases} \text{Uniform}(0,1) & \text{if } z_1 \leq 0.5 \\ \text{Uniform}(0,0.5) & \text{else} \end{cases}$$

Hence, this leads to the L-shaped latent support, i.e., $\mathcal{Z}^{\text{train}} := [0,1] \times [0,1] \setminus [0.5,1] \times [0.5,1]$.

We use $50k$ samples for the test dataset, while we use $20k$ samples for the train dataset along with $5k$ samples (25% of the train sample size) for the validation dataset.

BlockLatents Dataset. For this dataset, we allow the balls to move in both the x, y directions, so that $z \in \mathbb{R}^4$ and $\mathcal{B} = \{\{1,2\}, \{3,4\}\}$. For the case of **independent latents**, we sample each latent component independently and identically distributed according to a uniform distribution over $(0,1)$, i.e. $z_i \sim$ Uniform(0, 1). We rejected the images that present occlusion, i.e. when one ball hides another one.[*]

For the case of **dependent latents**, we sample the latents corresponding to the first ball similarly from the same continuous uniform distribution, i.e, $z_1, z_2 \sim$ Uniform (0, 1). However, the latents of the second ball are a function of the latents of the first ball, as described in what follows:

[*]Note that, in the independent latents case, the latents are not actually independent because of the rejection step which prevents occlusion from happening.

$$z_3 \sim \begin{cases} \text{Uniform}(0, 0.5) & \text{if } 1.25 \times (z_1^2 + z_2^2) \geq 1.0 \\ \text{Uniform}(0.5, 1) & \text{if } 1.25 \times (z_1^2 + z_2^2) < 1.0 \end{cases}$$

$$z_4 \sim \begin{cases} \text{Uniform}(0.5, 1) & \text{if } 1.25 \times (z_1^2 + z_2^2) \geq 1.0 \\ \text{Uniform}(0, 0.5) & \text{if } 1.25 \times (z_1^2 + z_2^2) < 1.0 \end{cases}$$

Intuitively, this means the second ball will be placed in either the top-left or the bottom-right quadrant based on the position of the first ball. We also exclude from the dataset the images presenting occlusion. Note that our dependent BlockLatent setup is same as the non-linear SCM case from Ahuja et al. [Ahuja et al., 2023].

We use $50k$ samples for both the train and the test dataset, along with $12.5k$ samples (25% of the train sample size) for the validation dataset.

Disconnected Support Dataset. For this dataset, we have setup similar to the **ScalarLatents** dataset; we fix the x-coordinates of both balls to 0.25 and 0.75 and only vary the y-coordinates so that $z \in \mathbb{R}^2$. We sample the y-coordinate of the first ball (z_1) from Uniform(0, 1). Then we sample the y-coordinate of the second ball (z_2) from either of the following continuous uniform distribution with equal probability; Uniform(0, 0.25) and Uniform(0.75, 1). This leads to a disconnected support given by $\mathcal{Z}^{\text{train}} := [0, 1] \times [0, 1] \setminus [0.25, 0.75] \times [0.25, 0.75]$.

We use $50k$ samples for the test dataset, while we use $20k$ samples for the train dataset along with $5k$ samples (25% of the train sample size) for the validation dataset.

D.2.3 Evaluation Metrics

Recall that, to evaluate disentanglement, we compute a matrix of scores $(s_{B,B'}) \in \mathbb{R}^{\ell \times \ell}$ where ℓ is the number of blocks in $\mathcal{B}$ and $s_{B,B'}$ is a score measuring how well we can predict the ground-truth block z_B from the learned latent block $\hat{z}_{B'} = \hat{g}_{B'}(x)$ outputted by the encoder. The final Latent Matching Score (LMS) is computed as $\text{LMS} = \arg\max_{\pi \in \mathfrak{S}_\mathcal{B}} \frac{1}{\ell} \sum_{B \in \mathcal{B}} s_{B,\pi(B)}$, where $\mathfrak{S}_\mathcal{B}$ is the set of permutations respecting $\mathcal{B}$ (Definition 3.4.1). These scores are always computed on the test set.

Metric $\text{LMS}_{\text{Spear}}$: As mentioned in the main paper, this metric is used for the **Scalar-Latents** dataset where each block is 1-dimensional. Hence, this metric is almost the same as the mean correlation coefficient (MCC), which is widely used in the nonlinear ICA literature [Hyvärinen and Morioka, 2016, 2017, Hyvärinen et al., 2019, Khemakhem et al., 2020b, Lachapelle et al., 2022b], with the only difference that we use Spearman correlation instead of Pearson correlation as a score $s_{B,B'}$. The Spearman correlation can capture nonlinear monotonous relations, unlike Pearson which can only capture linear dependencies. We favor Spearman over Pearson because our identifiability result (Theorem 3.4.6) guarantees we can recover the latents only up to permutation and element-wise invertible transformations, which can be nonlinear.

Metric LMS$_{\text{tree}}$: This metric is used for the **BlockLatents** dataset. For this metric, we take $s_{B,B'}$ to be the R^2 score of a Regression Tree with maximal depth of 10. For this, we used the class `sklearn.tree.DecisionTreeRegressor` from the `sklearn` library. We learn the parameters of the Decision Tree using the train dataset and then use it to evaluate LMS$_{\text{tree}}$ metric on the test dataset. For the additive decoder, it is easy to compute this metric since the additive structure already gives a natural partition $\mathcal{B}$ which matches the ground-truth. However, for the non-additive decoder, there is no natural partition and thus we cannot compute LMS$_{\text{tree}}$ directly. To go around this problem, for the non-additive decoder, we compute LMS$_{\text{tree}}$ for all possible partitions of d_z latent variables into blocks of size $|B| = 2$ (assuming all blocks have the same dimension), and report the best LMS$_{\text{tree}}$. This procedure is tractable in our experiments due to the small dimensionality of the problem we consider.

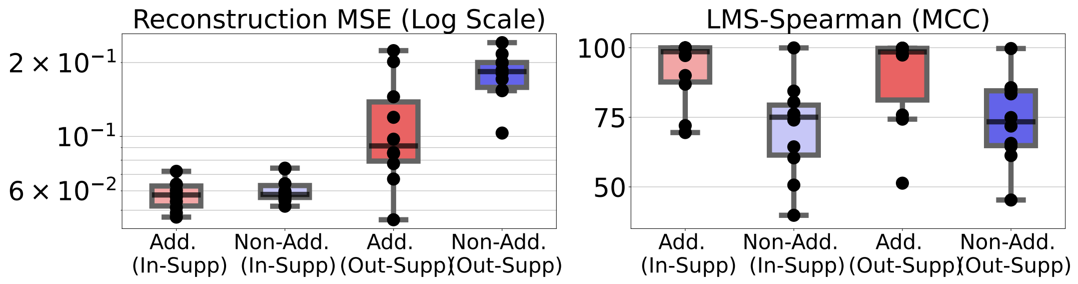
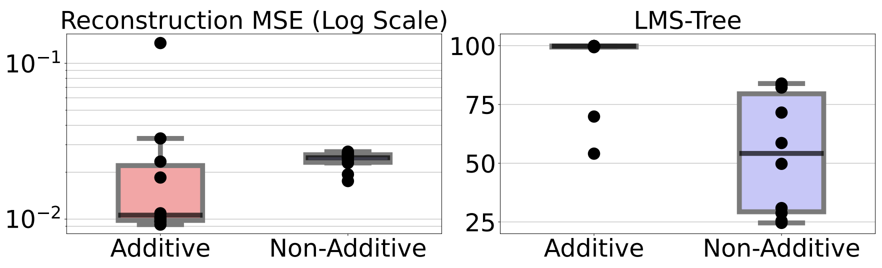


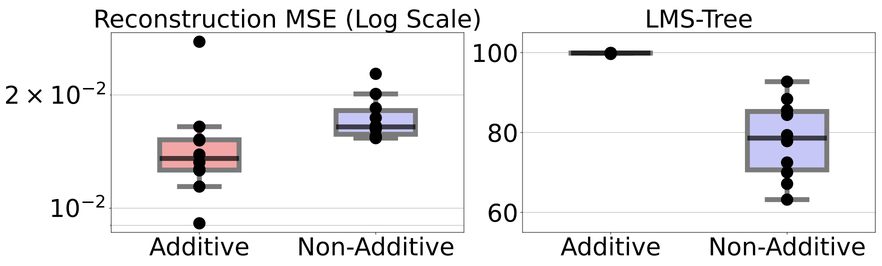
Figure D.1: Reconstruction mean squared error (MSE) (↓) and Latent Matching Score (LMS) (↑) over 10 different random initializations for **ScalarLatents** dataset.

D.2.4 Boxplots for main experiments (Table 3.1)

Since the standard error in the main results (Table 3.1) was high, we provide boxplots in Figures D.1 & D.2 to have a better visibility on what is causing this. We observe that the high standard error for the Additive approach was due to bad performance for a few bad random initializations for the ScalarLatents dataset; while we have nearly perfect latent identification for the others. Figure D.6e shows the latent space learned by the worst case seed, which somehow learned a disconnected support even if the ground-truth support was



(a) Independent Latent Case



(b) Dependent Latent Case

Figure D.2: Reconstruction mean squared error (MSE) (↓) and Latent Matching Score (LMS) (↑) for 10 different initializations for **BlockLatents** dataset.

connected. Similarly, for the case of Independent BlockLatents, there are only a couple of bad random initializations and the rest of the cases have perfect identification.

D.2.5 Additional Results: BlockLatents Dataset

To get a qualitative understanding of latent identification in the BlockLatents dataset, we plot the response of each predicted latent as we change a particular ground-truth latent factor. We describe the following cases of changing the ground-truth latents:

- **Ball 1 moving along x-axis:** We sample 10 equally spaced points for z_1 from $[0, 1]$; while keeping other latents fixed as follows: $z_2 = 0.25, z_3 = 0.50, z_4 = 0.75$. We will never have occlusion since the balls are separated along the y-axis $z_4 - z_2 > 0$.

- **Ball 2 moving along x-axis:** We sample 10 equally spaced points for z_3 from $[0, 1]$; while keeping other latents fixed as follows: $z_1 = 0.50, z_2 = 0.25, z_4 = 0.75$. We will never have occlusion since the balls are separated along the y-axis $z_4 - z_2 > 0$.

- **Ball 1 moving along y-axis:** We sample 10 equally spaced points for z_2 from $[0, 1]$; while keeping other latents fixed as follows: $z_1 = 0.25, z_3 = 0.75, z_4 = 0.50$. We will never have occlusion since the balls are separated along the x-axis $z_3 - z_1 > 0$.

- **Ball 2 moving along y-axis:** We sample 10 equally spaced points for z_4 from $[0, 1]$; while keeping other latents fixed as follows: $z_1 = 0.25, z_2 = 0.50, z_3 = 0.75$. We will never have occlusion since the balls are separated along the x-axis $z_3 - z_1 > 0$.

Figure 3.5 in the main paper presents the latent responses plot for the median LMS_{tree} case among random initializations. In Figure D.3, we provide the results for the case of best and the worst LMS_{tree} among random seeds. We find that Additive Decoder fails for only for the worst case random seed, while Non-Additive Decoder fails for all the cases.

Additionally, we provide the object-specific reconstructions for the Additive Decoder in Figure D.4. This helps us better understand the failure of Additive Decoder for the worst case random seed (Figure D.4c), where the issue arises due to bad reconstruction error.

D.2.6 Disconnected Support Experiments

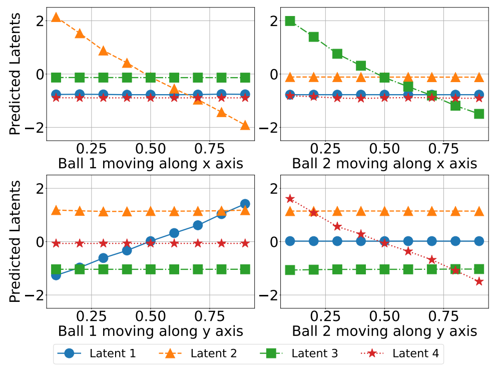
Since path-connected latent support is an important assumption for latent identification with additive decoders (Theorem 3.4.6), we provide results for the case where the assumption is not satisfied. We experiment with the **Disconnected Support** dataset (Section D.2.2) and find that we obtain much worse $\text{LMS}_{\text{Spear}}$ as compared to the case of training with L-shaped support in the **ScalarLatents** dataset. Over 10 different random initializations, we find mean $\text{LMS}_{\text{Spear}}$ performance of 69.5 with standard error of 6.69.

For better qualitative understanding, we provide visualization of the latent support and the extrapolated images for the median $\text{LMS}_{\text{Spear}}$ among 10 random seeds in Figure D.5. Somewhat surprisingly, the representation appears to be aligned in the sense that the first predicted latent corresponds to the blue ball while the second predicted latent correspond to the red ball. Also surprisingly, extrapolation occurs (we can see images of both balls high). That being said, we observe that the relationship between the predicted latent 2 ($\hat{z}_2$) and y-coordinate of second (red) ball is not monotonic, which explains why the Spearman correlation is so low (Spearman correlation scores are high when there is a monotonic relationship between both variables).

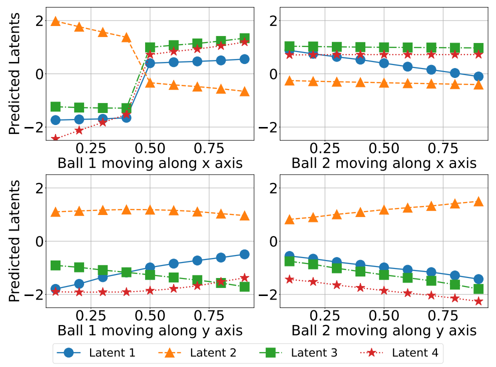
D.2.7 Additional Results: ScalarLatents Dataset

To get a qualitative understanding of extrapolation, we plot the latent support on the test dataset and sample a grid of equally spaced points from the support of each predicted latent on the test dataset. The grid represents the cartesian-product of the support of predicted latents and would contain novel combinations of latents that were unseen during training. We show the reconstructed images for each point from the cartesian-product grid to see whether the model is able to reconstruct well the novel latent combinations.

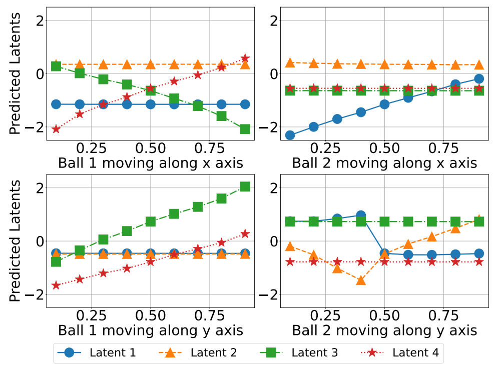
Figure 3.4 in the main paper presents visualizations of the latent support and the extrapolated images for the median $\text{LMS}_{\text{Spear}}$ case among random seeds. In Figure D.6, we provide the results for the case of best and the worst $\text{LMS}_{\text{Spear}}$ among random seeds. We find that even for the best case (Figure D.6b), Non-Additive Decoder does not generate good quality extrapolated images, while Additive Decoder generates extrapoalted images for the best and median case. The worst-case run for the Additive Decoder has disconnected support, which explains why it is not able to extrapolate.



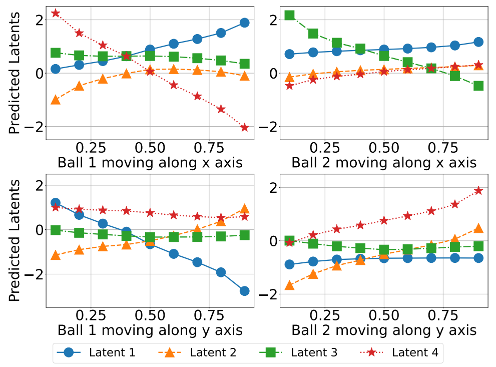
(a) Additive Decoder (Best) (LMS_{Tree} : 99.9)



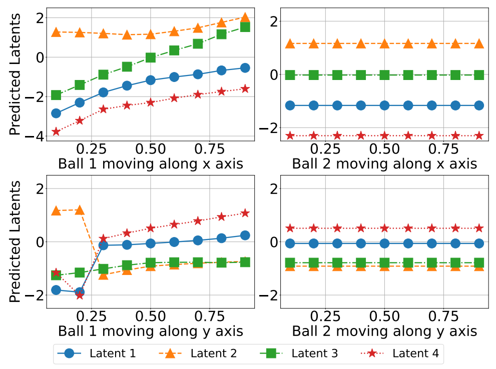
(b) Non-Additive Decoder (Best) (LMS_{Tree} : 83.9)



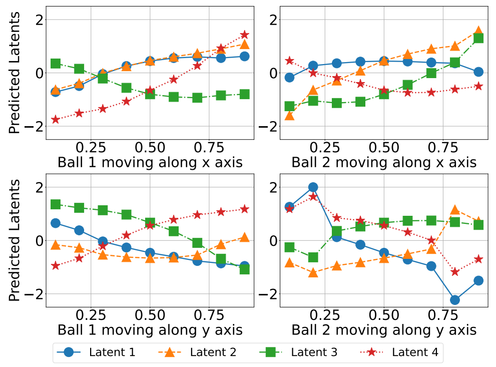
(c) Additive Decoder (Median) (LMS_{Tree} : 99.8)



(d) Non-Additive Decoder (Median) (LMS_{Tree} : 58.6)

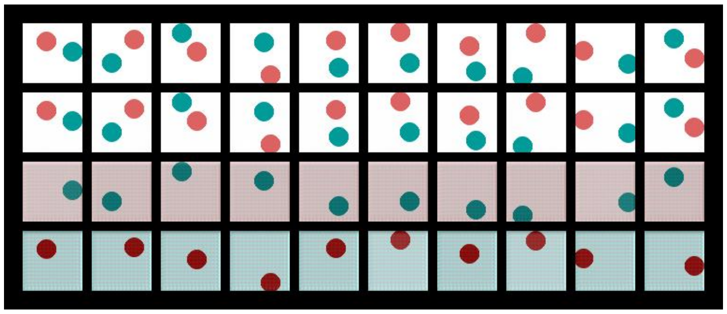


(e) Additive Decoder (Worst) (LMS_{Tree} : 54.1)

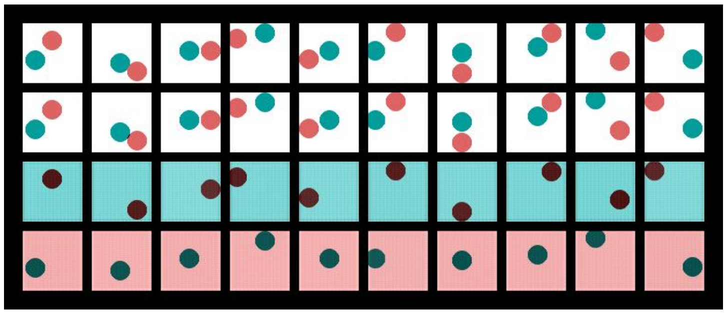


(f) Non-Additive Decoder (Worst) (LMS_{Tree} : 24.6)

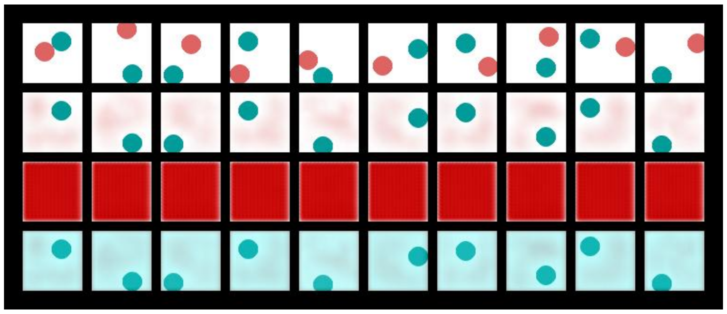
Figure D.3: Latent responses for the cases with the **best**/**median**/**worst** LMS_{Tree} among runs performed on the **BlockLatent** dataset with independent latents. In each plot, we report the latent factors predicted from multiple images where one ball moves along only one axis at a time.



(a) Additive Decoder (Best)



(b) Additive Decoder (Median)



(c) Additive Decoder (Worst)

Figure D.4: Object-specific renderings with the **best/median/worst** LMS$_{\text{tree}}$ among runs performed on the **BlockLatents** dataset with independent latents. In each plot, the first row is the original image, the second row is the reconstruction and the third and fourth rows are the output of the object-specific decoders. In the best and median cases, each object-specific decoder corresponds to one and only one object, e.g. the third row of the best case always corresponds to the red ball. However, in the worst case, there are issues with reconstruction as only one of the balls is generated. Note that the visual artefacts are due to the additive constant indeterminacy we saw in Theorem 3.4.6, which cancel each other as is suggested by the absence of artefacts in the reconstruction.

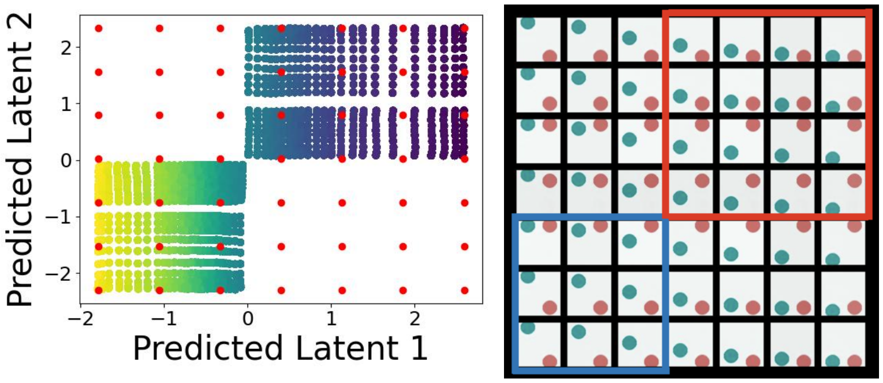
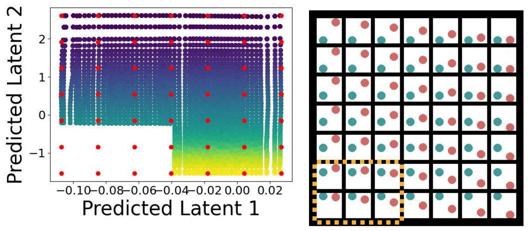
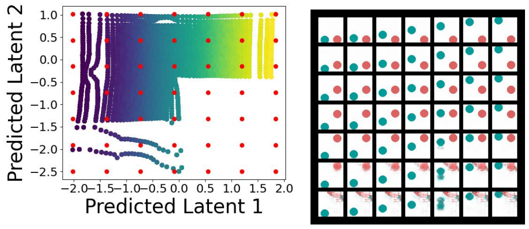


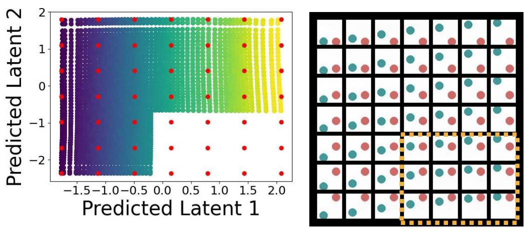
Figure D.5: Learned latent space, $\hat{\mathcal{Z}}^{\text{train}}$, and the corresponding reconstructed images of the additive decoder with the **median** $\text{LMS}_{\text{Spear}}$ among runs performed on the **Disconnected Support** dataset. The red dots correspond to latent factors used to generate the images.



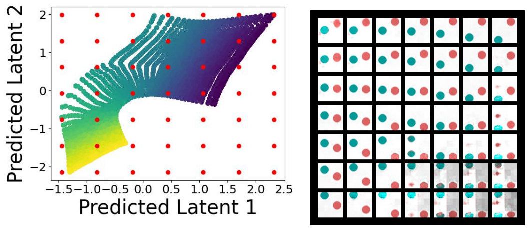
(a) Additive Decoder (Best) ($\text{LMS}_{\text{Spear}}$: 99.9)



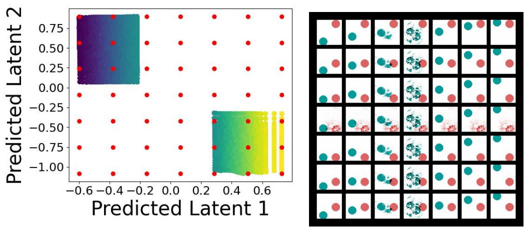
(b) Non-Additive Decoder (Best) ($\text{LMS}_{\text{Spear}}$: 99.9)



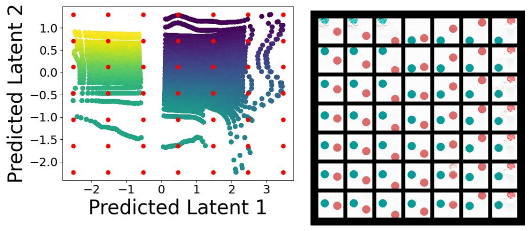
(c) Additive Decoder (Median) ($\text{LMS}_{\text{Spear}}$: 99.9)



(d) Non-Additive Decoder (Median) ($\text{LMS}_{\text{Spear}}$: 76.1)



(e) Additive Decoder (Worst) ($\text{LMS}_{\text{Spear}}$: 69.5)



(f) Non-Additive Decoder (Worst) ($\text{LMS}_{\text{Spear}}$: 39.8)

Figure D.6: Figure (a, c, e) shows the learned latent space, $\hat{\mathcal{Z}}^{\text{train}}$, and the corresponding reconstructed images of the additive decoder with the **best**/**median**/**worst** $\text{LMS}_{\text{Spear}}$ among runs performed on the **ScalarLatents** dataset. Figure (b, d, f) shows the same thing for the non-additive decoder. The red dots correspond to latent factors used to generate the images and the yellow square highlights extrapolated images.

Appendix E

Supplementary Material: Compositional Generalization with Additive Energy Models

E.1 Evaluating additive functions on affine-hull

We are given an additive function $g(z) = \sum_i g_i(z_i)$ and we want to evaluate the function on $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ where $z' = \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z \tau(z)$ and $\tau(z) = [\tau(z_1), \cdots, \tau(z_{d_z})] \in \mathbb{R}^{d_z \times m}$.

Define $\boldsymbol{g} = [g_1(z_1 = 1), \cdots, g_1(z_1 = m), \cdots, g_{d_z}(z_{d_z} = 1), \cdots, g_{d_z}(z_{d_z} = m)]$. Essentially $\boldsymbol{g} \in \mathbb{R}^{d_z \times m}$ is a concatenation of different components of g_i, where each component g_i is evaluated on the different values the factor z_i can take.

It is easy to observe that $g(z) = \sum_i g_i(z_i) =< \boldsymbol{g}, \tau(z) >$. Using this equality, we can show the following:

$$\begin{aligned} g(z') &=< \boldsymbol{g}, \tau(z') > \\ &=< \boldsymbol{g}, \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z \tau(z) > \\ &= \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z < \boldsymbol{g}, \tau(z) > \\ &= \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z g(z) \end{aligned}$$

Hence Proved.

E.2 Proof for Theorem 4.2.7

In order to prove this theorem we first establish some basic lemmas.

Lemma E.2.1. *If we consider binary factors, $m = 2$, then there are four possible concatenated one-hot vectors $\tau(z)$, i.e., $\tau(0,0)$, $\tau(0,1)$, $\tau(1,0)$, and $\tau(1,1)$ denoted by t_1, t_2, t_3, t_4. Each t_i can be expressed as an affine combination of the remaining.*

Proof.

$$(+1) \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + (-1) \cdot \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + (+1) \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{E.1}$$

$$(-1) \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + (+1) \cdot \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + (+1) \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{E.2}$$

$$(+1) \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} + (+1) \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + (-1) \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \tag{E.3}$$

$$(-1) \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} + (+1) \cdot \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + (+1) \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \tag{E.4}$$

□

We now interpret Lemma E.2.1 from a geometric perspective on. Consider a 2×2 subgrid of the grid G. In Figure E.1, we show a 2×2 subgrid and illustrate Lemma E.2.1. Lets define the following:

- $t_1 = [0, \cdots, 1_i, \cdots 0, 0, \cdots, 1_j, \cdots 0]$
- $t_2 = [0, \cdots, 1_{i+1}, \cdots 0, 0, \cdots, 1_j, \cdots 0]$
- $t_3 = [0, \cdots, 1_i, \cdots 0, 0, \cdots, 1_{j+1}, \cdots 0]$
- $t_4 = [0, \cdots, 1_{i+1}, \cdots 0, 0, \cdots, 1_{j+1}, \cdots 0]$

Observe that using Lemma E.2.1, we get $t_4 = t_2 + t_3 - t_1$. Similarly, we can express every other t_i in terms of rest of t_j's. While the above observation is made with contiguous points on the grid, the observation generalizes as follows. We consider each point on the grid as a vertex. Each vertex $v = (i, j)$ shares an edge with another vertex if the Hamming distance equals one and there is no edge otherwise (similar to graph construction in Definition 4.2.2). We can now consider any 2×2 subgrid which does not necessarily consist of contiguous points. In Figure E.2, we show the graph for this general subgrid. Even in such a case, we continue to be able apply Lemma E.2.1 following the same exact proof recipe.

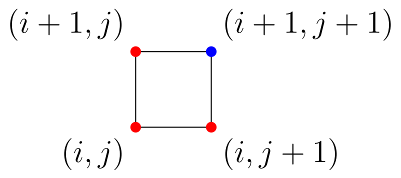


Figure E.1: The three red corners correspond to the observed and blue corner is the one we extrapolate to.

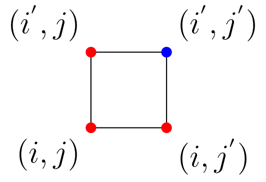


Figure E.2: The three red corners correspond to the observed and blue corner is the one we extrapolate to.

Define the observed set of groups as N. Suppose that their affine span contains all the points in a subgrid S of size $p \times n$. Next, we show that if we add a new point $g = (g_1, g_2)$, which shares either the x or the y-coordinate with from outside this subgrid, then the affine span of $N \cup \{g\}$ is a larger subgrid of size either $p \times n + 1$ or $(p+1) \times n$. Define C_x as the set of points that share the same x coordinate as g and same y coordinate as S, $C_x = \{(g_x, 1), (g_x, 2), \cdots, (g_x, n)\}$. Define C_y as the set of points that share the same y coordinate as g and same x coordinate as S, $C_y = \{(1, g_y), (2, g_y), \cdots, (p, g_y)\}$.

Theorem E.2.2. *If the affine span of the observed set N contains a subgrid S of size $p \times n$. If the new point g shares the x-coordinate with a point in S, then the the affine span of $N \cup \{g\}$ contains $S \cup C_y$.*

Proof. Consider a subgrid $S = \{x_1, \cdots, x_p\} \times \{y_1, \cdots, y_n\}$. Without loss of generality, we can permute the points and make the subgrid contiguous as follows $S = \{1, \cdots, p\} \times \{1, \cdots, n\}$.

We observe a new group g, which shares x coordinate with one of the points in S. Without loss of generality let this point be $(1, n+1)$ (we can always permute the columns and rows to achieve such a configuration). Consider the triplet – $(z_1, z_2, z_3) = ((1, n), (2, n), (1, n+1))$. We use Lemma E.2.1 to infer that the fourth point $z_4 = (2, n+1)$ on this 2×2 subgrid can be obtained as an affine combination of this triplet, i.e., $\tau(z_4) = \alpha\tau(z_1) + \beta\tau(z_2) + \gamma\tau(z_3)$. Also, we know z_1, z_2, z_3 can be written as an affine combination of seen points in N as follows $\tau(z_1) = \sum_{k\in N} a_k\tau(z_{\theta_k})$, $\tau(z_2) = \sum_{k\in N} b_k\tau(z_{\theta_k})$, and $\tau(z_3) = \sum c_k\tau(z_{\theta_k})$. Observe that

$$\begin{aligned}\tau(z_4) &= \alpha\tau(z_1) + \beta\tau(z_2) + \gamma\tau(z_3) = \alpha(\sum a_k\tau(z_{\theta_k})) + \beta(\sum b_k\tau(z_{\theta_k})) + \gamma(\sum c_k\tau(z_{\theta_k}))\\ &= \sum_{k\in N}(\alpha a_k + \beta b_k)\tau(z_{\theta_k}) + \gamma\tau(z_3)\end{aligned} \tag{E.5}$$

Observe that $\sum_k(\alpha a_k + \beta b_k) = (\alpha\sum_k a_k + \beta\sum_k b_k) = \alpha + \beta$. Since $\alpha+\beta+\gamma$ Thus z_4 is an affine combination of points in $N \cup \{g\}$. Thus we have shown the claim for the point $(2, n+1)$. We can repeat this claim for point $(3, n+1)$ and so on until we reach $(m, n+1)$ beyond which there would be no points in S that are expressed as affine combination of N. We can make this argument formal through induction. We have already shown the base case above. Suppose all the points $(j, n+1)$ in $j \leq i < m$ are in the affine span of $N \cup \{z_3\}$. Consider the point $z_4 = (i+1, n+1)$. Construct the triplet $(z_1, z_2, z_3) = ((i, n), (i, n+1), (i+1, n))$. Again from Lemma E.2.1, it follows that $\tau(z_4) = \alpha\tau(z_1) + \beta\tau(z_2) + \gamma\tau(z_3)$. We substitute z_1, z_2 and z_3 with their corresponding affine combinations. $\tau(z_4) = \alpha\sum_{k\in N\cup\{g\}} a_k\tau(z_{\theta_k}) + \beta\sum_{k\in N\cup\{g\}} b_k\tau(z_{\theta_k}) + \gamma\sum_{k\in N\cup\{g\}} c_k\tau(z_{\theta_k})$. Since $\sum_{k\in N\cup\{g\}} \alpha a_k + \beta b_k + \gamma c_k = 1$.

$\square$

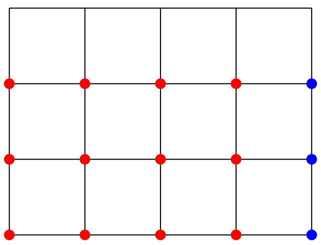


Figure E.3: A 4×5 grid. Red points form S, a new point g in blue is observed and all other points in blue are in the affine span of $S \cup \{g\}$

We now describe a simple procedure that helps us understand how many groups we need to see before we are guaranteed that the affine span of seen points span the whole grid. We first construct a determinstic procedure.

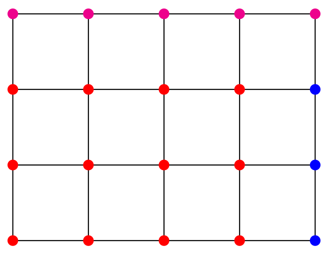


Figure E.4: A 4×5 grid. Red points and blue points form S, a new point g in magenta is observed and all other points in green are in the affine span of $S \cup \{g\}$

- We start with a base set of three points, $B = \{(1,1),(1,2),(2,1)\}$. From Lemma E.2.1, the affine span contains $(2,2)$.
- For each $i \in \{2, \cdots, m-1\}$ add points $\{(1, i+1)\} \cup \{(i+1, 1)\}$ to the set. From Theorem E.2.2, it follows that affine span of B contains $(i+1 \times i+1)$ subgrid.

Observe that from the above deterministic procedure shows that we can span the entire m^2 grid with $2m-1$ points. We illustrate this procedure in Figure E.3 and Figure E.4.

We now discuss how the randomized version of this procedure can also allow us to span the entire grid with $\mathcal{O}(m \log(m))$ points, where m denotes the total values taken by each z_i. Set $S = \emptyset, B = \emptyset$ and $\mathsf{Flag} = x$. We use S_x to denote the distinct set of x-coordinates that appear in S and same goes for S_y.

- Sample a group g from G uniform at random. $B = B \cup \{g\}, S = S \cup \{g\}$
- While $S \neq G$, sample a group g from G uniform at random. $B = B \cup \{g\}$
  - If Flag equals x, g shares the x-coordinate with point in S and is not in S, then update $S = S \cup (S_x \times \{g_y\})$ and $\mathsf{Flag} = y$.
  - If Flag equals y, g shares the y-coordinate with point in S and is not in S, then update $S = S \cup (\{g_x\} \times S_y)$ and $\mathsf{Flag} = x$.

In the above procedure, in every step in the while loop a group g is sampled. Note that when Flag flips from x to y, then following Theorem E.2.2, the set S belongs to the affine span of B. We can say the same when Flag flips from y to x. In the next theorem, we will show that the while loop terminates after $8cm\log(m)$ steps with a high probability and the affine span contains the entire grid G. We follow this strategy. We count the time it takes for Flag to flip from x to y (from y to x) as it grows the size of S from a $k \times k$ subgrid to $k \times (k+1)$ ($k \times (k+1)$ subgrid to $(k+1) \times (k+1)$) subgrid.

Theorem 4.2.7. *Assume 2-d factors, i.e., $d_z = 2$. If the number of sampled factors is more than $8c * m\log(m)$, then $\mathsf{Aff}(\mathcal{Z}^{\text{train}}) = [m] \times [m]$ with probability $\geq 1 - \frac{1}{c}$.*

Proof. We take the first group g that is sampled. Without loss of generality, we say this group is $(1, 1)$.

Define an event A_1^k: Sampled g shares first (x-coordinate) with S (size $k \times k$) and is not in S. This can be interpreted as the probability for Flag to flip from x to y. $P(A_1^k) = \dfrac{(k)(m-k)}{m^2}$.

Define an event A_2^k: Sampled g shares first (y-coordinate) with S (size $k \times (k+1)$) and is not in S. This can be interpreted as the probability for Flag to flip from y to x. $P(A_2^k) = \dfrac{(k+1)(m-k)}{m^2}$.

Define T_1^k as the number of groups that need to be sampled before A_1^k occurs. Define T_2^k as the number of groups that need to be sampled before A_2^k occurs.

Define $T_{\mathsf{sum}} = \sum_{k=1}^{m-1}(T_1^k + T_2^k)$. T_{sum} is the total number of groups sampled before the affine span of the observed groups spans the grid $G : [m] \times [m]$.

We now derive a bound on $\mathbb{E}[T_{\mathsf{sum}}] = \sum_{k=1}^{m-1}(\mathbb{E}[T_1^k] + \mathbb{E}[T_2^k])$. We first simplify $\sum_{k=1}^{m-1}\mathbb{E}[T_1^k]$ first as follows.

$$\begin{aligned}
\sum_{k=1}^{m-1}\mathbb{E}[T_1^k] &= \sum_{k=1}^{m-1}\frac{1}{P(A_1^k)} \\
&= \sum_{k=1}^{m-1} m^2/(k(m-k)) \\
&= 2\sum_{k=1}^{(m-1)/2} m^2/(k(m-k)) \\
&= 2m\sum_{k=1}^{(m-1)/2}\left[\frac{1}{k} + \frac{1}{m-k}\right] \\
&\approx 4m\log((m-1)/2)
\end{aligned} \tag{E.6}$$

Similarly, we obtain a similar bound for $\sum_{k=1}^{m-1}\mathbb{E}[T_2^k]$ as follows.

$$
\begin{aligned}
\sum_{k=1}^{m-1}(\mathbb{E}[T_2^k] &= \sum_{k=1}^{m-1}\frac{1}{P(A_2^k)} \\
&= \sum_{k=1}^{m-1} m^2/((k+1)(m-k)) \\
&= 2\sum_{k=1}^{(m-1)/2} m^2/((k+1)(m-k)) \\
&\leq 2m\sum_{k=1}^{(m-1)/2}\Big[\frac{1}{k+1}+\frac{1}{m-k}\Big] \\
&\approx 4m\log((m-1)/2)
\end{aligned}
\tag{E.7}
$$

Therefore, $\mathbb{E}[T_{\mathsf{sum}}] \approx 8m\log(m/2)$. From Markov inequality, it immediately follows that $P(T_{\mathsf{sum}} \leq 8cm\log(m/2)) \geq 1-\frac{1}{c}$. □

E.3 Proof for Theorem 4.3.2: Generative Case

Theorem 4.3.2. *[Affine Hull Extrapolation with Additive Energy Models] Under Assumption 4.3.1, if the learned additive energy model $\hat{p}(x|z)$ matches the true additive energy model $p(x|z)$ on training data, i.e., $\hat{p}(x|z) = p(x|z)\ \forall z \in \mathcal{Z}^{\text{train}}$, then we would have affine hull extrapolation, $\hat{p}(x|z) = p(x|z)\ \forall z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.*

Proof. We start by expanding the expressions for log densities (true and estimated) below.

$$
\begin{aligned}
-\log\Big[\hat{p}(x|z)\Big] &= \langle 1, \hat{\boldsymbol{E}}(x,z)\rangle + \log(\hat{B}(z)) \\
-\log\Big[p(x|z)\Big] &= \langle 1, \boldsymbol{E}(x,z)\rangle + \log(B(z))
\end{aligned}
\tag{E.8}
$$

We equate the densities for the training attributes $z \in \mathcal{Z}^{\text{train}}$, which implies the following:

$$
\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle = \langle 1, \boldsymbol{E}(x,z)\rangle + C(z)\ \ \forall z \in \mathcal{Z}^{\text{train}}, \tag{E.9}
$$

where $C(z) = \log\big(B(z)/\hat{B}(z)\big)$.

Now, lets consider $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$, which implies $\tau(z') = \sum_{z\in\mathcal{Z}^{\text{train}}}\alpha_z\tau(z)$. Using similar arguments as in Appendix E.1, we can show that $\langle 1, \hat{\boldsymbol{E}}(x,z')\rangle = \sum_{z\in\mathcal{Z}^{\text{train}}}\alpha_z\,\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle$. We further use this decomposition as follows:

$$\begin{aligned}
\langle 1, \hat{\boldsymbol{E}}(x, z')\rangle &= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \langle 1, \hat{\boldsymbol{E}}(x, z)\rangle \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z(\langle 1, \boldsymbol{E}(x, z)\rangle + C(z)) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \langle 1, \boldsymbol{E}(x, z)\rangle + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z C(z) \\
&= \langle 1, \boldsymbol{E}(x, z')\rangle + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z C(z) \\
&= \langle 1, \boldsymbol{E}(x, z')\rangle + H(\mathcal{Z}^{\text{train}}, \{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})
\end{aligned} \tag{E.10}$$

From this we can infer the following:

$$\begin{aligned}
\hat{p}(x|z') &= \frac{1}{\hat{B}(z')} \exp\Big(- \langle 1, \hat{\boldsymbol{E}}(x, z')\rangle \Big) \\
&= \frac{1}{\hat{B}(z')} \exp\Big(- \langle 1, \boldsymbol{E}(x, z')\rangle - H(\mathcal{Z}^{\text{train}}, \{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})\Big)
\end{aligned} \tag{E.11}$$

We now use the fact that density $\hat{p}(x|z')$ integrates to one for continuous random variables (or alternatively the probability sums to one for discrete random variables), and simplify as follows:

$$\begin{aligned}
& \int \hat{p}(x|z')dx = 1 \\
\implies & \int \frac{1}{\hat{B}(z')} \exp\Big(- \langle 1, \boldsymbol{E}(x, z')\rangle - H(\mathcal{Z}^{\text{train}}, \{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})\Big)dx = 1 \\
\implies & \frac{\exp\Big(- H(\mathcal{Z}^{\text{train}}, \{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})\Big)}{\hat{B}(z')} \int \exp\Big(- \langle 1, \boldsymbol{E}(x, z')\rangle \Big)dx = 1
\end{aligned}$$

Using the definition of partition function $B(z') = \int \exp\Big(- \langle 1, \boldsymbol{E}(x, z')\rangle \Big)dx$, we have the following

$$\frac{1}{\hat{B}(z')} \exp\Big(- H(\mathcal{Z}^{\text{train}}, \{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})\Big) = \frac{1}{B(z')} \tag{E.12}$$

We substitute (E.12) into (E.11) to obtain

$$\hat{p}(x|z') = \frac{1}{B(z')} \exp\Big(- \langle 1, \boldsymbol{E}(x, z')\rangle \Big) = p(x|z') \tag{E.13}$$

Hence, $\hat{p}(x|z') = p(x|z) \ \ \forall z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ whenever $\hat{p}(x|z) = p(x|z) \ \ \forall z \in \mathcal{Z}^{\text{train}}$. $\square$

E.4 Proof for Theorem 4.3.5: Discriminative Case

Theorem 4.3.5. *Consider the data generation process in Definition 4.3.3 and the proposed estimator $\hat{p}(z|x)$ (4.4). If $\hat{p}(z|x) = p(z|x)\ \forall z \in \mathcal{Z}^{\text{train}}$, then we can solve the compositional classification task with $\tilde{p}(z|x)$ (4.5), i.e., $\tilde{p}(z|x) = p(z|x)\ \forall z \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.*

Proof. By def. of additive energy model (4.3), we have $-\langle 1, \boldsymbol{E}(x,z)\rangle = \log\big(p(x|z)\big) + \log B(z)$. Consider $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$, then we have $z' = \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z z$

First we show that the partition function at $\tilde{z}$ can be expressed as affine combination of partition of the individual points and a correction term. By the definition of partition function, $B(z') = \int \exp(-\langle 1, \boldsymbol{E}(x,z')\rangle)dx$, which is further simplified as follows:

$$\begin{aligned}
\log B(z') &= \log\Big(\int \exp\big(-\langle 1, \boldsymbol{E}(x,z')\rangle\big)dx\Big) \\
&= \log\Big(\int \exp\big(-\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \langle 1, \boldsymbol{E}(x,z)\rangle\big)dx\Big) \\
&= \log\Big(\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z\big(\log p(x|z) + \log B(z)\big)\Big)dx\Big) \\
&= \log\Big(\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log B(z)\Big)\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z)\Big)dx\Big) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log B(z) + \log\Big(\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z)\Big)dx\Big)
\end{aligned} \tag{E.14}$$

Denote $R(\{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}}) = \log\Big(\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z)\Big)dx\Big)$

We now simplify $\log p(x|z')$ as follows for all $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.

$$\begin{aligned}
&\log p(x|z') = -\langle 1, \boldsymbol{E}(x,z')\rangle - \log B(z') \\
&= -\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \langle 1, \boldsymbol{E}(x,z)\rangle - \log B(z') \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z\big(\log p(x|z) + \log B(z)\big) - \log B(z') \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z) + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log B(z) - \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log\big(B(z)\big) - R(\{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}}) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z) - R(\{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})
\end{aligned} \tag{E.15}$$

Note: The above expression shows how density $p(x|z')$ for novel points under the true additive energy model can be written as a function of density for training points

$\{p(x|z)|z \in \mathcal{Z}^{\text{train}}\}$. We now try to obtain a similar expression for $p(z'|x)$ by first simplifying $\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z)$ in terms of $p(z|x)$.

$$\begin{aligned}
\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z) &= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log \Big(\frac{p(z|x)p(x)}{p(z)}\Big) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z) + \log p(x) \times \sum_{z\ \in\mathcal{Z}^{\text{train}}} \alpha_z \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z) + \log p(x)
\end{aligned} \tag{E.16}$$

Similarly, $R(\{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}})$ can be phrased in terms of $p(z|x)$ as follows.

$$\begin{aligned}
&R(\{\alpha_z\}_{z\in\mathcal{Z}^{\text{train}}}) \\
&= \log\Big(\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(x|z)\Big)dx\Big) \\
&= \log\Big(\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z) + \log p(x)\Big)dx\Big) \\
&= \log\Big(\exp\Big(-\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z)\Big)\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)p(x)dx\Big) \\
&= -\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z) + \log\Big(\int \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)p(x)dx\Big) \\
&= -\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z) + \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)
\end{aligned} \tag{E.17}$$

Using (E.16), (E.17) to simplify (E.15) we obtain.

$$
\begin{aligned}
\log p(x|z') &= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) + \log p(x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)\\
\log\Big(\frac{p(z'|x)p(x)}{p(z')}\Big) &= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) + \log p(x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)\\
\log p(z'|x) - \log p(z') &= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)\\
\log p(z'|x) &= \log p(z') + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)\\
p(z'|x) &= \mathsf{Softmax}\Big(\log p(z') + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)\Big)
\end{aligned}
\tag{E.18}
$$

Remark. Note that we do not need to use Softmax in the equation above and we can exactly write the following:

$$p(z'|x) = \exp\Big(\log p(z') + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)\Big)$$

Essentially, the denominator in Softmax would sum to one in this case. The reason why we still wrote the above expression with Softmax is to handle the general case where the prior distribution can change $p(z)$, and in that case the expression can written as follows.

$$\log p(z'|x) = \log\frac{p^{\text{test}}(x)}{p(x)} + \log p(z') + \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) - \log\Big(\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]\Big)$$

where the term $\log\frac{p^{\text{test}}(x)}{p(x)}$ was not present earlier as we were not considering shifts in the prior distribution. For this general case, the Softmax equality written above would still be true as the term $\log\frac{p^{\text{test}}(x)}{p(x)}$ does not depend on z. Therefore, we choose to write the Softmax variant as its more general and would be true in the case where the prior distribution undergoes a shift at test time. To remind the reader, the connection of Affine extrapolation with extrapolation to test distribution is made by assuming $\mathcal{Z}^{\text{test}} \subseteq \mathsf{Aff}(\mathcal{Z}^{\text{train}})$.

The above equation (E.18) expresses $p(\tilde{z}|x)$ entirely in terms of the distributions over training support $\mathcal{Z}^{\text{train}}$. Hence, if the learner can successfully extrapolate to $p(z'|x)$ for all $z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ if it estimates $p(z|x)$ correctly for all $z \in \mathcal{Z}^{\text{train}}$.

We use the form of the proposed classifier (4.4) to estimate $p(z|x)$ over training support in the equation above. Note that since the learner has done estimation perfectly on training support, i.e., $p(x|z) = \hat{p}(x|z)\ \forall z \in \mathcal{Z}^{\text{train}}$, we can plug the learner's estimate of $\hat{p}(x|z)$ (4.4) in the equation above. First we simplify the term $\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)$ as follows.

$$\begin{aligned}
&\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log \hat{p}(z|x) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \Big(\log \operatorname{Softmax}\Big(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle - \log \hat{M}(z) + \log p(z)\Big)\Big) \\
&= \sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \Big(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle - \log \hat{M}(z) + \log p(z)\Big) \\
&\qquad - \log\Big(\sum_{\tilde{z}\in\mathcal{Z}^{\text{train}}} \exp\Big(-\langle 1, \hat{\boldsymbol{E}}(x,\tilde{z})\rangle - \log \hat{M}(\tilde{z}) + \log p(\tilde{z})\Big)\Big)
\end{aligned} \tag{E.19}$$

If we drop the latter term, then softmax in equation (E.18) still remains invariant as this term does not depend on z. Hence, later when we will substitute this into (E.18), we can ignore the last term.

We now turn to simplify $\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big]$ using the expression above.

$$\begin{aligned}
&\mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log p(z|x)\Big)\Big] \\
&= \mathbb{E}_x\Big[\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log \hat{p}(z|x)\Big)\Big] \\
&= \mathbb{E}_x\left[\frac{\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \Big(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle - \log \hat{M}(z) + \log p(z)\Big)\Big)}{\sum_{\tilde{z}\in\mathcal{Z}^{\text{train}}} \exp\Big(-\langle 1, \hat{\boldsymbol{E}}(x,\tilde{z})\rangle - \log \hat{M}(\tilde{z}) + \log p(\tilde{z})\Big)}\right] \\
&= \mathbb{E}_x\left[\frac{\exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \Big(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle\Big)}{\sum_{\tilde{z}\in\mathcal{Z}^{\text{train}}} \exp\Big(-\langle 1, \hat{\boldsymbol{E}}(x,\tilde{z})\rangle - \log \hat{M}(\tilde{z}) + \log p(\tilde{z})\Big)}\right] \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log\big(\frac{p(z)}{\hat{M}(z)}\big)\Big) \\
&= \mathbb{E}_x\left[\frac{\exp\Big(-\langle 1, \hat{\boldsymbol{E}}(x,z')\rangle\Big)}{\sum_{\tilde{z}\in\mathcal{Z}^{\text{train}}} \exp\Big(-\langle 1, \hat{\boldsymbol{E}}(x,\tilde{z})\rangle - \log \hat{M}(\tilde{z}) + \log p(\tilde{z})\Big)}\right] \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \log\big(\frac{p(z)}{\hat{M}(z)}\big)\Big) \\
&= \hat{Q}(z') \exp\Big(\sum_{z\in\mathcal{Z}^{\text{train}}} \alpha_z \Big(-\log \hat{M}(z) + \log p(z)\Big)\Big)
\end{aligned} \tag{E.20}$$

In the above simplification, we use $\hat{Q}(z')$ from (4.3.1). We now substitute (E.20), (E.19) into (E.18) to obtain.

$$\begin{aligned} p(z'|x) &= \mathsf{Softmax}\Big(\log p(z') + \sum_{z \in \mathcal{Z}^{\text{train}}} \alpha_z \Big(-\langle 1, \hat{\boldsymbol{E}}(x,z)\rangle \Big) - \log \hat{Q}(z') \Big) \\ &= \mathsf{Softmax}\Big(\log p(z') - \langle 1, \hat{\boldsymbol{E}}(x,z')\rangle - \log \hat{Q}(z') \Big) \end{aligned} \tag{E.21}$$

Note that the RHS is equal to our estimator $\tilde{p}(z'|x)$ (4.5). Therefore, we have established that $p(z'|x) = \tilde{p}(z'|x) \ \ \forall z' \in \mathsf{Aff}(\mathcal{Z}^{\text{train}})$ whenever $p(z|x) = \hat{p}(z|x) \ \ \forall z \in \mathcal{Z}^{\text{train}}$. Hence proved. $\square$

Group (y, a)	Training	Validation	Test
(0, 0)	3498	467	2255
(0, 1)	184	466	2255
(1, 0)	56	133	642
(1, 1)	1057	133	642

Table E.1: Statistics for each group across the different splits for **Waterbirds** benchmark.

Group (y, a)	Training	Validation	Test
(0, 0)	71629	8535	9767
(0, 1)	66874	8276	7535
(1, 0)	22880	2874	2480
(1, 1)	1387	182	180

Table E.2: Statistics for each group across the different splits for **CelebA** benchmark.

Group (y, a)	Training	Validation	Test
(0, 0)	784	127	273
(0, 1)	507	75	191
(1, 0)	196	33	65
(1, 1)	789	114	345

Table E.3: Statistics for each group across the different splits for **MetaShift** benchmark.

E.5 Experiments

E.5.1 Dataset Details

Waterbirds [Wah et al., 2011]. The task is to classify land birds ($y = 0$) from water birds ($y = 1$), where the spurious attributes are land background ($a = 0$) and water background ($a = 1$). Table E.1 provides details regarding the statics of each factor $z = (y, a)$ in the dataset.

CelebA [Liu et al., 2015]. The task is to classify blond hair ($y = 1$) from non-blond hair ($y = 0$), where the spurious attribute is gender, female ($a = 0$) and male ($a = 1$). Table E.2 provides details regarding the statics of each factor $z = (y, a)$ in the dataset.

MetaShift [Liang and Zou, 2022]. The task is to classify cats ($y = 0$) from dogs ($y = 1$), where the spurious attribute is background, indoor ($a = 0$) and outdoor ($a = 1$). Table E.3 provides details regarding the statics of each factor $z = (y, a)$ in the dataset.

E.5.2 Additional Results

Table E.4 and Table E.5 present the results for the CelebA and MetaShift benchmark respectively. We similar trends as with the Waterbirds benchmark (Table 4.1) that the proposed AddEnergy method outperforms the baselines for worst group accuracy prediction in almost all the scenarios.

Removed (y, a)	Method	Average Acc	Worst Group Acc
(0, 0)	ERM	0.74 (0.01)	0.34 (0.0)
(0, 0)	GroupDRO	0.83 (0.0)	0.73 (0.02)
(0, 0)	AddEnergy	0.86 (0.0)	0.82 (0.0)
(0, 1)	ERM	0.91 (0.0)	0.59 (0.01)
(0, 1)	GroupDRO	0.81 (0.01)	0.68 (0.02)
(0, 1)	AddEnergy	0.83 (0.03)	0.74 (0.02)
(1, 0)	ERM	0.87 (0.0)	0.10 (0.0)
(1, 0)	GroupDRO	0.88 (0.0)	0.73 (0.02)
(1, 0)	AddEnergy	0.73 (0.13)	0.69 (0.13)
(1, 1)	ERM	0.94 (0.0)	0.15 (0.01)
(1, 1)	GroupDRO	0.93 (0.0)	0.42 (0.01)
(1, 1)	AddEnergy	0.87 (0.0)	0.78 (0.01)

Table E.4: Results for compositional generalization on the **CelebA** benchmark. The first column describes the factors that were dropped during training. The performance for both the metrics is denoted as mean ± standard error over 3 random seeds on the test dataset.

Removed (y, a)	Method	Average Acc	Worst Group Acc
(0, 0)	ERM	0.87 (0.0)	0.82 (0.0)
(0, 0)	GroupDRO	0.85 (0.01)	0.81 (0.01)
(0, 0)	AddEnergy	0.85 (0.0)	0.80 (0.01)
(0, 1)	ERM	0.85 (0.01)	0.45 (0.04)
(0, 1)	GroupDRO	0.89 (0.0)	0.72 (0.0)
(0, 1)	AddEnergy	0.91 (0.0)	0.78 (0.01)
(1, 0)	ERM	0.89 (0.0)	0.48 (0.0)
(1, 0)	GroupDRO	0.89 (0.0)	0.60 (0.01)
(1, 0)	AddEnergy	0.86 (0.0)	0.71 (0.0)
(1, 1)	ERM	0.82 (0.0)	0.64 (0.01)
(1, 1)	GroupDRO	0.85 (0.0)	0.77 (0.0)
(1, 1)	AddEnergy	0.89 (0.01)	0.76 (0.03)

Table E.5: Results for compositional generalization on the **MetaShift** benchmark. The first column describes the factors that were dropped during training. The performance for both the metrics is denoted as mean $\pm$ standard error over 3 random seeds on the test dataset.