Incorporating Interventional Independence Improves Robustness against Interventional Distribution Shift

Anonymous authors
Paper under double-blind review

Abstract

We consider the problem of learning robust discriminative representations of latent variables that are causally related to each other via a directed graph. In addition to passively collected observational data, the training dataset also includes interventional data obtained through targeted interventions on some of these latent variables to learn representations that are robust against the resulting interventional distribution shifts. However, existing approaches treat interventional data like observational data, even when the underlying causal model is known, and ignore the independence relations that arise from these interventions. Since these approaches do not fully exploit the causal relational information resulting from interventions, they learn representations that produce large disparities in predictive performance on observational and interventional data. This performance disparity worsens when the number of interventional data samples available for training is limited. In this paper, (1) we first identify a strong correlation between this performance disparity and adherence of the representations to the statistical independence conditions induced by the underlying causal model during interventions. (2) For linear models, we derive sufficient conditions on the proportion of interventional data in the training dataset, for which enforcing statistical independence between representations corresponding to the intervened node and its non-descendants during interventions lowers the test-time error on interventional data. Combining these insights, (3) we propose RepLIn, a training algorithm to explicitly enforce this statistical independence during interventions. We demonstrate the utility of RepLIn on a synthetic dataset and on real image and text datasets on facial attribute classification and toxicity detection, respectively. Our experiments show that RepLIn is scalable with the number of nodes in the causal graph and is suitable to improve the robustness of representations against interventional distribution shifts of both continuous and discrete latent variables compared to the ERM baselines.

1 Introduction

We consider the problem of learning robust discriminative representations corresponding to latent random variables for downstream prediction tasks from their observable data. These latent variables usually correspond to semantic concepts such as the color of an object, the level of glucose in the blood, and a person's age. The relationship between these latent variables can be modeled using directed acyclic graphs (DAGs) called *causal graphs*. Causal modeling allows manually altering the causal graph and observing its effects on the data. E.g., intervene on the amount of insulin (parent variable) in the blood by consuming an insulin inhibitor and then measuring the glucose level (child variable) in the blood. This procedure is known as a *causal intervention*, and the data collected through this procedure is called *interventional data*. In contrast, data passively collected without intervention is known as *observational data*. Several types of interventions are possible on a causal graph, of which we are interested in *hard interventions* where we manually set the

value of one or more variables. Intervening on a graph node renders it statistically independent of its *parent nodes* in the causal graph[1]. See (Peters et al., 2017, Chapter 6) and (Pearl, 2009, Chapter 3).

Suppose the latent variables are A and B, such that A causes B ($A \to B$) during observations. An attribute-specific representation $\boldsymbol{F}_A$ corresponding to A learned by a model from observational training data alone may contain information about its child node B due to the association between A and B. For instance, consider a computer-aided diagnosis system that inputs a chest X-ray image and outputs two representations corresponding to air sac inflammation (A) to predict pneumonia and fluid accumulation around lungs (B) to check for pleural effusion, respectively. This design makes the system modular and interpretable. These representations will be used by separate predictors for the corresponding diagnosis. The causal relation between these medical conditions is as follows: pneumonia can lead to excess fluid accumulation, although similar fluid accumulation can occur due to factors unrelated to pneumonia. For instance, suppose that fluid accumulation happened as a side effect of some medication. In this case, it is possible that the representation corresponding to predicting pneumonia incorporates information about excess fluid accumulation to aid pneumonia diagnosis, although the presence of fluid accumulation does not mean pneumonia. Moreover, the practitioner does not know what conditions the patient has when their chest X-ray image is fed into the system, prohibiting a system design that yields a separate representation for each medical condition. To avoid such catastrophic mistakes, these representations must be designed explicitly to include only the information corresponding to their diagnostic purpose. In other words, these models must be made robust against *interventional distribution shifts*.

To improve the robustness of the learned representations, interventional data samples are included in the training data to learn models that are robust to interventional distribution shifts. For example, in (Sauer & Geiger, 2021; Gao et al., 2023), interventional data was generated to train image classification models invariant to texture and background. In (Arjovsky et al., 2019; Heinze-Deml & Meinshausen, 2021), interventional data is treated merely as data sourced from different domains or *environments*, and they do not consider the explicit statistical independence relations that arise from interventions[2]. As we demonstrate, ignoring these independence relations may result in representations that are still susceptible to interventional distribution shifts during inference. Additionally, performing interventions is often challenging, thus limiting the amount of interventional data available for training. This furthers the need for a causally motivated learning strategy that exploits the limited amount of interventional training data.

We first consider a simple case study in which we observe that models that do not learn independent representations during interventions show a performance drop on interventional data. We then derive sufficient conditions on the proportion of interventional data during training, under which enforcing linear independence between interventional features of linear models during training can reduce test-time error on interventional data. Motivated by these theoretical insights, we propose "Representation Learning from Interventional Data" (**RepLIn**), an algorithm to train models with improved robustness against interventional distribution shifts. We confirm the utility of RepLIn on a variety of synthetic (Sec. 5.1) and real datasets (Secs. 5.2 and 5.3) on various modalities, and demonstrate its scalability to the number of nodes (Sec. 6.2).

To summarize our contributions,

- We demonstrate a positive correlation between the accuracy drop during interventional distribution shift and the dependence between representations corresponding to the label node and its children. We refer to this as "interventional feature dependence" (Sec. 3.3).
- We theoretically explain why linear ERM models are susceptible to interventional distribution shifts in the regime of linear causal models. In the same setting, we theoretically and empirically show that enforcing linear independence between interventional features improves robustness when sufficient interventional data is available during training and establish the sufficient condition (Sec. 3.4).

[1]For ease of use, we refer to "statistical independence" as "independence", and "hard interventions" as "interventions". We will also use "features" and "representations" interchangeably to denote the vector representations of the data learned by a model.

[2]The distribution shift due to differing environments is more general than interventional distribution shift. However, this work argues against an agnostic approach for robustness against interventional distribution shift.

- We propose a novel training algorithm that combines these insights and demonstrates that this model minimizes the drop in accuracy under interventional distribution shifts by explicitly enforcing independence between interventional features (Sec. 4).

2 Related Works

Identifiable Causal Representation Learning (ICRL) (Locatello et al., 2019; Schölkopf et al., 2021; Hyvärinen et al., 2024) seek to learn representations of the underlying causal model under certain assumptions (Hyvärinen et al., 2024), and are important to interpretable representation learning. Interventions have also been used in ICRL works (Lippe et al., 2022b; 2023; Ahuja et al., 2023; Squires et al., 2022; von Kügelgen et al., 2023; Zhang et al., 2023; Jiang & Aragam, 2023; Buchholz et al., 2023; Varıcı et al., 2024b; Bing et al., 2024; Lachapelle et al., 2024), with the key underlying idea that the variables that become independent during a known intervention can be identifiably learned. In contrast to these works, we are interested in a broader class of discriminative representation learning when some underlying causal relations are known. Instead of learning the entire causal model, we seek to exploit the known independence relations from interventions to learn discriminative representations that are robust against these interventions. Moreover, the representations learned by ICRL methods usually have permutation ambiguity. That is, the representations are disentangled but not mapped to the semantic attribute that we wish to predict in a downstream task. As we will describe later, our approach overcomes this ambiguity by explicitly learning attribute-specific representations. We provide a detailed review of ICRL in App. C.

Interventional data is key in causal discovery (Eberhardt et al., 2005; Yu et al., 2019; Ke et al., 2019; Lippe et al., 2022a; Wang et al., 2022b) as one can only retrieve causal relations up to a Markov equivalent graph without interventions or assumptions on the causal model. For example, known interventional targets have been used for unsupervised causal discovery of linear causal models (Subramanian et al., 2022), interventional and observational data have been leveraged for training a supervised model for causal discovery (Ke et al., 2022), and interventions with unknown targets were used for differentiable causal discovery (Brouillard et al., 2020). Interventional data also find applications in reinforcement learning (Gasse et al., 2021; Ding et al., 2022a) and recommendation systems (Zhang et al., 2021; Krauth et al., 2022; Luo et al., 2024). While this body of work focuses on discovering causal relations in the data, our work considers how to leverage known causal relations to learn data representations that are robust to distribution shifts induced by interventions.

Our setting also differs from that of domain generalization (DG). In DG, the learning objective is a predictor for an attribute of interest that is robust/invariant to changes in the domain/environment (Mahajan et al., 2021; Wang et al., 2022a; Ding et al., 2022b). Here, there is no interest in learning representations for the domain, and multiple factors could be jointly treated as a single domain. Moreover, there is also no requirement that the learned predictor for the attribute of interest is free of domain information (Rosenfeld et al., 2022). Therefore, the learned representations obtained from domain generalization may not be trustworthy for modular applications such as the medical diagnosis system described in Sec. 1. A more detailed discussion is provided in App. D.

Training with group-imbalanced data leads to models that suffer from group-bias during inference. In such cases, resampling the data according to the inverse sample frequency can improve generalization and robustness. Studies such as (Gulrajani & Lopez-Paz, 2021; Idrissi et al., 2022) have shown that ERM with resampling is effective against spurious correlations and is a strong baseline for domain generalization. Recent works such as dynamic importance reweighting (Fang et al., 2020), SRDO (Shen et al., 2020), and MAPLE (Zhou et al., 2022) *learn to resample* using a separate validation set that acts as a proxy for the test set. However, learning such a resampling requires a large dataset of both *observational* and *interventional* data, which is often not practically feasible. In contrast, we will exploit known independence relations during interventions to improve robustness to interventional distributional shifts.

3 The Learning from Interventional Data Problem

Notation: Random variables and random vectors are denoted by regular (e.g., A) and bold (e.g., $\boldsymbol{a}$) serif characters, respectively. The distribution of a random variable A is denoted by P_A.

We now formally define the problem of interest in this paper, namely *learning discriminative representations to predict latent variables that are robust against interventional distribution shifts*[3], in general terms, and examine a specific case study in Sec. 3.1. The learning problem is characterized by a DAG $\mathcal{G}$ that causally relates the attributes of interest $A_1, \ldots, A_m$, and B. Let $\mathbf{Pa}_B = \{A_1, \ldots, A_m\}$ denote the parents of the attribute B. These attributes along with other unobserved exogenous variables $\boldsymbol{U}$, generate the observable data $\boldsymbol{X}$, i.e., $\boldsymbol{X} = g_{\boldsymbol{X}}(B, A_1, \ldots, A_m, \boldsymbol{U})$. During interventions, the variable B is set to values drawn from a known distribution independent of $\mathbf{Pa}_B$. Therefore, the post-intervention variable B (denoted by $\tilde{B}$) is statistically independent of its parents, i.e., $\tilde{B} \perp\!\!\!\perp \mathbf{Pa}_B$, as shown in Fig. 1. Although $g_{\boldsymbol{X}}$ is not affected by this intervention, the distribution of $\boldsymbol{X}$ (now denoted by $\boldsymbol{X}'$) will change since it is a function of B. Note that to learn representations that are robust against distribution shift due to intervention on B, our setting only provides us information about B and its parents in the causal graph, and not of any causal relations between $A_1, \ldots, A_m$. We also do not place restrictions on the functional form of causal relations between $A_1, \ldots, A_m, B$, and $\boldsymbol{X}$, or on their marginal distributions. For training, data samples from both observational and interventional distributions are available, i.e., $\mathcal{D}^{\text{train}} = \mathcal{D}^{\text{obs}} \cup \mathcal{D}^{\text{int}}$ where $\mathcal{D}^{\text{obs}} \sim P(\boldsymbol{X}, B, A_1, \ldots, A_m)$ and $\mathcal{D}^{\text{int}} \sim P(\boldsymbol{X}', \tilde{B}, A_1, \ldots, A_m)$. However, the number of interventional training samples is much less compared to the number of observational training samples, i.e., $|\mathcal{D}^{\text{obs}}| \ll |\mathcal{D}^{\text{int}}|$ Given $\mathcal{D}^{\text{train}}$ and $\mathcal{G}$, the goal is to learn attribute-specific discriminative representations $\boldsymbol{F}_B = h_B(\boldsymbol{X})$ and $\boldsymbol{F}_{A_i} = h_{A_i}(\boldsymbol{X})$ that are robust against distribution shifts due to intervention on B.

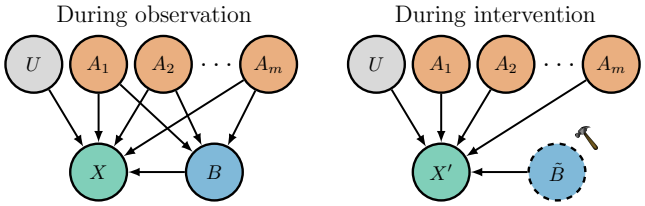
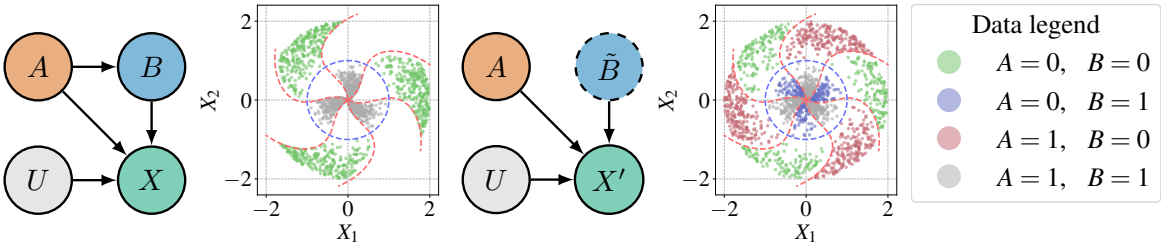


Figure 1: **Causal graph modification due to intervention:** During observation, B is the effect of its parent variables $\mathbf{Pa}_B = \{A_1, \ldots, A_m\}$. When we intervene on B, it becomes statistically independent of its parents.

3.1 Does Accuracy Drop during Interventions Correlate with Interventional Feature Dependence?

In this section, we will design a case study using a synthetic dataset and establish a correlation between the accuracy drop on interventional data and the statistical dependence between the attribute representations under intervention.



(a) Observational graph and data (b) Interventional graph and data

Figure 2: **An illustration of Windmill Dataset:** A and B are binary random variables that are causally linked to each other and $\boldsymbol{X}$, as shown in (a). By intervening on B as shown in (b), we make $A \perp\!\!\!\perp \tilde{B}$. $\boldsymbol{X} = g_{\boldsymbol{X}}(A, B, \boldsymbol{U})$ where $\boldsymbol{U}$ denotes unobserved noise variables. The true decision boundaries for predicting A and B from $\boldsymbol{X}$ are shown in red and blue dashed lines, respectively. See App. I for a detailed description.

Problem Setting: Consider the causal graph shown in Fig. 2a. Here, A and B are binary random variables that generate the observed data $\boldsymbol{X} \in \mathbb{R}^2$. $\boldsymbol{X}$ is also affected by an unobserved noise variable $\boldsymbol{U}$. Functionally, $\boldsymbol{X} = g_{\boldsymbol{X}}(A, B, \boldsymbol{U})$. A itself could be a function of unobserved random factors that are of no predictive interest to us. Therefore, we model $A \sim \text{Bernoulli}(0.6)$. The distribution of B is only affected by A, as denoted by the arrow between them. Analytically, $B \coloneqq A$, where $\coloneqq$ indicates the causal assignment operator, following

[3]We use "discriminative" to explicitly state that the purpose of these representations is robust prediction and not data generation. Information loss with improved robustness is therefore acceptable.

(Peters et al., 2017). Visually, the observed data looks like a windmill. The value of A determines the windmill's blade, and B determines the radial distance. The precise angle and radial distance of the points are sampled from a noise distribution independent of A and B. We also shear the windmill blades according to a sinusoidal function of the radial distance. In Fig. 2b, we intervene on B, modeled as $\tilde{B} \sim \text{Bernoulli}(0.5)$. This induces a change in the distribution of B and subsequently that of $\boldsymbol{X}$. Since the intervention is independent of A, $\tilde{B}$ is also independent of A, denoted by removing the arrow between A and $\tilde{B}$. Note that $g_{\boldsymbol{X}}$ is unaffected by this intervention. The exact mathematical formulation of the data-generating process is provided in App. I.

Learning task: The task is to accurately predict A and B from $\boldsymbol{X}$ at test time. We have N samples for training, where βN are interventional and $(1-\beta)N$ are observational with $0 < \beta \ll 1$. For this demonstration, we set $N = 40,000$, $\beta = 0.01$ to get 39,600 observational and 400 interventional samples. We train a feed-forward network with two hidden layers to learn representations $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$ corresponding to A and B, respectively. We normalize them by dividing each by their corresponding L_2 norm. Separate linear classifiers predict A and B from $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$ respectively. By construction, $g_{\boldsymbol{X}}$ in the data-generating process is a one-to-one mapping. Therefore, predicting A and B from $\boldsymbol{X}$ accurately is possible. However, the true decision boundary for A is more complex than that of B[4]. Therefore, the model may rely on information from B to predict A due to their association during observation, similar to the concept of simplicity bias from (Shah et al., 2020). As a result, $\boldsymbol{F}_A$ may contain information about B even during interventions when $A \perp\!\!\!\perp B$. Following the standard ERM framework, the cross-entropy errors in predicting A and B from $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$, respectively, provide the training signal. The statistical loss function can be written as $\mathcal{L}_{\text{total}}(f) = \mathbb{E}_{P_{\text{train}}}[\mathcal{L}_{\text{pred}}(f, \boldsymbol{X})]$. The training distribution is a mixture of observational and interventional distributions with $(1-\beta)$ and β acting as the corresponding mixture weights. Thus, $\mathcal{L}_{\text{total}}(f) = (1-\beta)\mathbb{E}_{P_{\text{obs}}}\left[\mathcal{L}_{\text{pred}}(f, \boldsymbol{X}^{\text{obs}})\right] + \beta\mathbb{E}_{P_{\text{int}}}[\mathcal{L}_{\text{pred}}(f, \boldsymbol{X}^{\text{int}})]$.

ERM version	Accuracy in predicting A			Accuracy in predicting B			NHSIC
	Observation	Intervention	Relative drop	Observation	Intervention	Relative drop	
Vanilla	99.98 ± 0.01	60.15 ± 3.12	0.40 ± 0.03	100.00 ± 0.00	99.99 ± 0.01	0	0.72 ± 0.06
w/ Resampling	94.53 ± 1.14	70.20 ± 3.73	0.26 ± 0.03	100.00 ± 0.00	99.99 ± 0.01	0	0.64 ± 0.08

Table 1: The relative drop in accuracy in predicting A correlates well with a gap in the measure of dependence between the learned representations on interventional data.

Observations: Tab. 1 shows the accuracy of ERM in predicting A and B on observational and interventional data during validation. Ideally, *we expect no drop in accuracy from observation to intervention* if the learned representations are robust against interventional distribution shift. However, we observe that ERM performs only slightly better than random chance in predicting A on interventional data. As a remedy, we modify the vanilla ERM method to sample observational and interventional data in separate batches, and thus prevent the gradients from interventional training samples being obfuscated by those from observational training samples, which are likely to be more in number in a given batch. This is equivalent to sampling interventional data $\left(\frac{1-\beta}{\beta}\right)$-times as observational data. Therefore, we refer to this version as "ERM-Resampled". The equivalent loss for a learning function f in ERM-Resampled is $\mathcal{L}_{\text{total}}(f) = \mathbb{E}_{P_{\text{obs}}}\left[\mathcal{L}_{\text{pred}}(f, \boldsymbol{X}^{\text{obs}})\right] + \mathbb{E}_{P_{\text{int}}}[\mathcal{L}_{\text{pred}}(f, \boldsymbol{X}^{\text{int}})]$. Note that β does not appear in $\mathcal{L}_{\text{total}}(f)$ due to resampling. Although ERM-Resampled performs better than vanilla ERM, we observe that ERM-Resampled still exhibits a large drop in predictive accuracy between observational and interventional data during inference. Also, we observe the drop in observational accuracy of ERM-Resampled in predicting A as it improved interventional accuracy. As we will show in Sec. 3.4, the reduced observational accuracy is due to the removal of spurious information previously exploited to boost its observational accuracy.

3.2 Measuring Statistical Dependence Between Interventional Features

The key consequence of hard interventions in causal graphs is that the variable being intervened upon becomes independent of all its non-descendants. Since the predictive accuracy on the parent node is affected

[4]We informally define "complexity" as the minimum polynomial degree required to approximate the decision boundary.

by intervention, we hypothesize that *the representation corresponding to the parent node remains dependent on the child node during intervention, even when their underlying latent variables in the causal graph become independent*. To verify our hypothesis, we measure the dependence between the representations. We choose to measure the dependence between the representations instead of between the representations and the latent attributes because we aim to learn robust representations for every attribute.

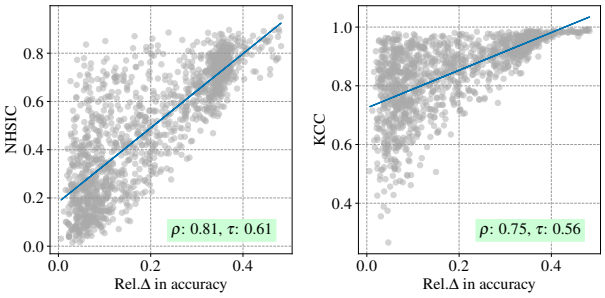
Dependence Measure: We use HSIC (Gretton et al., 2005) to measure dependence between a pair of high-dimensional continuous random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$. Empirical HSIC between N i.i.d. samples $\mathcal{X} = \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N}$ and $\mathcal{Y} = \left\{\boldsymbol{y}^{(i)}\right\}_{i=1}^{N}$ from $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively, can be computed as $\mathrm{HSIC}(\mathcal{X}, \mathcal{Y}) = \frac{1}{(N-1)^2}\mathrm{Trace}\left[\boldsymbol{K}_X \boldsymbol{H} \boldsymbol{K}_Y \boldsymbol{H}\right]$, where $\boldsymbol{H}$ is the $N \times N$ centering matrix, and $\boldsymbol{K}_X, \boldsymbol{K}_Y \in \mathbb{R}^{N \times N}$ are Gram matrices whose $(i,j)^{\text{th}}$ entries are $k_X\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)$ and $k_Y\left(\boldsymbol{y}^{(i)}, \boldsymbol{y}^{(j)}\right)$, respectively. Here, k_X and k_Y are the kernel functions associated with a universal kernel (e.g., RBF kernel). Since HSIC is unbounded, we normalize it as $\mathrm{NHSIC}(\mathcal{X}, \mathcal{Y}) = \frac{\mathrm{HSIC}(\mathcal{X},\mathcal{Y})}{\sqrt{\mathrm{HSIC}(\mathcal{X},\mathcal{X})\,\mathrm{HSIC}(\mathcal{Y},\mathcal{Y})}}$, following (Cortes et al., 2012; Cristianini et al., 2001). We also use random Fourier features (Rahimi & Recht, 2007) to improve computational efficiency.

Observations: Tab. 1 compares NHSIC values between the features $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$ learned by ERM and ERM-Resampled on interventional data from Windmill dataset. We observe that features learned by ERM had more statistical dependence during interventions than those by vanilla ERM, indicating a larger violation of the underlying statistical independence relations in the causal graph during interventions. Interestingly, the relative drop in accuracy also increases with the statistical dependence between interventional features.

3.3 Strength of Correlation between Drop in Accuracy and Interventional Features Dependence

How strong is the observed correlation between the dependence of features and the drop in accuracy? For a given combination of predictive task and dataset, does it hold for a variety of hyperparameter settings? To answer these questions, we train several models under the ERM-Resampled setting described in Sec. 3.1. The representations are learned using feed-forward networks, each with one to six hidden layers and with 20 to 200 hidden units. We also use early-stopping in our training, as it was noted as an effective regularizer (Sagawa et al., 2020). Early-stopping is executed in our experiments by choosing an arbitrary number of training epochs for each run. ~~We also randomly set the number of training epochs to use early-stopping as a regularizer.~~ We measure the robustness of a model to interventional distribution shift using the relative drop in accuracy between observational and interventional data: $\mathrm{Rel.}\Delta = \frac{\text{Obs acc.} - \text{Int acc.}}{\text{Obs acc.}}$. Similar experiments were reported in (Sreekumar & Boddeti, 2023), although their primary research question concerned the effect of data and model complexities on spurious correlations. In the following experiment, we expand their setting to deeper models and more variety in hyperparameters while foregoing the variation in data complexity.

In Fig. 3, we plot the relative drop in accuracy against the interventional feature dependence. In addition to NHSIC, we also use kernel canonical correlation (KCC) (Bach & Jordan, 2002) to measure the dependence. We observe that all models with a high relative drop in accuracy also have a large interventional feature dependence (see top-right regions of the plots). However, the corollary is not true – a large interventional feature dependence does not mean a relative drop in accuracy. Therefore, we conclude for this case study that *a relative drop in accuracy is always accompanied by interventional feature dependence*. The strength of the correlation between the relative drop in accuracy and interventional feature dependence is quantitatively measured using Spearman rank correlation coefficient (ρ) (Spearman, 1904) and Kendall rank corre-



(a) Rel.Δ against NHSIC (b) Rel.Δ against KCC

Figure 3: Across models with different capacities, a relative drop in accuracy is always accompanied by interventional feature dependence, while the corollary does not hold. Interventional feature dependence is measured using NHSIC and KCC.

lation coefficient (τ) (Kendall, 1938). In Fig. 3a, $\rho = 0.81$ and $\tau = 0.61$ when the dependence is measured using NHSIC, indicating that the correlation we noted in Sec. 3.2 can be observed for a wide range of hyper-parameters. When KCC is used for measuring dependence between interventional representations, $\rho = 0.75$ and $\tau = 0.56$ as shown in Fig. 3b.

Note that the correlation measures are affected by the choice of measure of dependence. Both NHSIC and KCC satisfy the postulates for an appropriate measure of dependence in (Rényi, 1959) and measure dependence from the spectrum of the cross-covariance operator between RKHSs. However, NHSIC measures the Hilbert-Schmidt norm of the cross-covariance operator while KCC measures its spectral norm (largest singular value). As a result, KCC is more suited for independence tests where the presence of dependence is more important than its overall strength. Informally, KCC is a "harsher" measure of dependence compared to NHSIC. Another commonly used metric to measure and enforce independence between representations is maximum mean discrepancy (MMD) (Gretton et al., 2012). MMD was originally proposed as a tool to check if two samples came from the same distribution or not. When used to measure dependence between two random variables, MMD and HSIC are equivalent. Given two random variables between which we wish to compute the dependence, HSIC between these random variables is equivalent to MMD between the joint distribution and the product of marginals of these variables (Schrab, 2025). Moreover, they have similar computational costs. For the remainder of this work, we will use NHSIC for training and analysis, and reserve KCC for evaluation.

3.4 Will Minimizing Dependence between Interventional Features Improve Robustness?

In Sec. 3.3, we showed that strong interventional feature dependence always accompanies a large relative drop in accuracy. Based on this correlation, we may ask the following question: *will minimizing interventional feature dependence improve the robustness to interventional distribution shifts?* We consider a linear causal model to answer this question theoretically. The detailed proof of each step is provided in App. B.

Causal Model: We use the causal model shown in Fig. 2a with A and B being continuous random variables. A and B are causally related during observation as $B := w_{AB}A$. The observed data signal $\boldsymbol{X}$ is generated from A and B as $\boldsymbol{X} := \begin{bmatrix} X_A \\ X_B \end{bmatrix} + \boldsymbol{U}$, where $X_A := w_A A$ and $X_B := w_B B$. $\boldsymbol{U} := \begin{bmatrix} U_A \\ U_B \end{bmatrix}$ is exogenous noise. U_A and U_B are independent of A and B respectively. We intervene on B as shown in Fig. 2b, severing the causal relation between A and B. The intervened variable is denoted as B' and $B' \perp\!\!\!\perp A$.

Learning model: Similar to the case study, the task is to predict the latent variables A and B from observed data signal $\boldsymbol{X}$. The training dataset is sampled from a training distribution P_{train} that contains observational and interventional samples. We model P_{train} as a mixture of observation distribution P_{obs} and interventional distribution P_{int} with $(1-\beta)$ and β acting as the mixture weights, i.e., $P_{\text{train}} = (1-\beta)P_{\text{obs}} + \beta P_{\text{int}}$. We use linear models to learn attribute-specific representations $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$, from which predictions $\hat{A}$ and $\hat{B}$, respectively, are made using the classifiers. The linear models are parameterized by $\boldsymbol{\Theta}^{(A)}$ and $\boldsymbol{\Theta}^{(B)}$, and the classifiers are parameterized by $\boldsymbol{c}^{(A)}$ and $\boldsymbol{c}^{(B)}$.

Statistical Risk: The parameter matrix of the linear feature extractor described before can be written in terms of its constituent parameter vectors as $\boldsymbol{\Theta}^{(A)} = \begin{bmatrix} \boldsymbol{\theta}_A^{(A)\top} \\ \boldsymbol{\theta}_B^{(A)\top} \end{bmatrix}$. Assuming zero mean for all latent variables[5], the statistical squared error of an arbitrary model in predicting A from an interventional test sample $\boldsymbol{X}$ is,

$$E_A = \underbrace{\left(1 - w_A \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_A^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_{\boldsymbol{U}_A}^2}_{E_A^{(1)}} + \underbrace{\left(w_B \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{B'}^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{\boldsymbol{U}_B}^2}_{E_A^{(2)}} \tag{1}$$

where $\rho_A^2 = \mathbb{E}_{P_{\text{int}}}\left[A^2\right]$, $\rho_{B'}^2 = \mathbb{E}_{P_{\text{int}}}\left[B'^2\right]$, $\rho_{\boldsymbol{U}_A}^2 = \mathbb{E}_{P_{\text{int}}}\left[U_A^2\right]$, and $\rho_{\boldsymbol{U}_B}^2 = \mathbb{E}_{P_{\text{int}}}\left[U_B^2\right]$. The statistical risk can be split into two components: (1) $E_A^{(1)}$ in terms of A and U_A, and (2) $E_A^{(2)}$ in terms of B and U_B. $E_A^{(2)} \neq 0$ when $\boldsymbol{\theta}_B^{(A)} \neq \boldsymbol{0}$. A non-zero $\boldsymbol{\theta}_B^{(A)}$ indicates that the representation $\boldsymbol{F}_A$ is a function of X_B, i.e., it learns

[5]The zero mean assumption is to make our calculations easier. This will not affect our conclusion from the proof, as we can always learn the mean of the data separately.

a spurious correlation with B. Thus the prediction $\hat{A}$ is susceptible to interventions on B. In contrast, a robust model will have $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}$, and thus $E_A^{(2)} = 0$. Derivation of Eq. (1) is provided in App. B.1.

Optimal ERM model: The optimal ERM model is obtained by minimizing the expected risk in predicting the latent attributes. Since parameters are not shared between the prediction of $\boldsymbol{a}$ and $\boldsymbol{b}$, we can consider their optimization separately. We consider the optimization of parameters for predicting $\boldsymbol{a}$ since we are interested in the performance drop while predicting A from interventional data.

$$\boldsymbol{\Theta}^{(A)*}, \boldsymbol{c}^{(A)*} = \underset{\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}}{\operatorname{argmin}} \, \mathbb{E}_{P_{\text{train}}} \left[\left(A - \boldsymbol{c}^{(A)\top} \boldsymbol{\Theta}^{(A)\top} \boldsymbol{X} \right)^2 \right] \tag{2}$$

For a given training error, there is no unique solution for $\boldsymbol{\Theta}^{(A)}$ and $\boldsymbol{c}^{(A)}$. Therefore, we can equivalently optimize for $\boldsymbol{\psi}_A = \boldsymbol{c}^{(A)\top} \boldsymbol{\Theta}^{(A)\top}$. We write $\boldsymbol{\psi}_A = \begin{bmatrix} \psi_1 \\ \psi_2 \end{bmatrix}$ where $\psi_1 = \boldsymbol{c}^{(A)\top} \boldsymbol{\theta}_A^{(A)}$ and $\psi_2 = \boldsymbol{c}^{(A)\top} \boldsymbol{\theta}_B^{(A)}$. The learning objective in Eq. (2) then reduces to,

$$\boldsymbol{\psi}_A^* = \underset{\boldsymbol{\psi}_A}{\operatorname{argmin}} \, \mathbb{E}_{P_{\text{train}}} \left[\left(A - \boldsymbol{\psi}_A \boldsymbol{X} \right)^2 \right] \tag{3}$$

We solve Eq. (3) by setting the gradients to zero. To check the robustness of the optimal ERM model, we can verify whether $\psi_2^* = 0$ or not since a robust model will have $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}$. Solving Eq. (3), we get:

$$\psi_2^* = \frac{-(1-\beta) w_B w_{AB} \sigma_A^2 \sigma_{U_A}^2}{T} \neq 0 \tag{4}$$

where T is a non-zero scalar. This implies that $E_A^{(2)} \neq 0$ in optimal ERM models. *Therefore, optimal ERM are not robust against interventional distribution shift.* Also, note that a robust model is not a minimizer of prediction loss on the training distribution as the minimizer in Eq. (4) leads to non-zero $\boldsymbol{\theta}_B^{(A)}$. This can explain the drop in observational accuracy of ERM-Resampled as it improved the interventional accuracy in predicting A in Sec. 3.1 and is also illustrated later in App. H. The detailed derivation is provided in App. B.2.

Minimizing linear dependence: In Sec. 3.3, we showed that dependence between interventional features correlated positively with the drop in accuracy on interventional data. We will now verify if minimizing dependence between interventional features can minimize the drop in accuracy in a linear setting. The interventional features are given by $\boldsymbol{F}_A = \boldsymbol{\Theta}^{(A)\top} \boldsymbol{X}$ and $\boldsymbol{F}'_B = \boldsymbol{\Theta}^{(B)\top} \boldsymbol{X}$.

$$\boldsymbol{F}_A = \boldsymbol{\Theta}^{(A)\top} \boldsymbol{X} = X_A \boldsymbol{\theta}_A^{(A)} + X_B \boldsymbol{\theta}_B^{(A)}$$
$$\boldsymbol{F}'_B = \boldsymbol{\Theta}^{(B)\top} \boldsymbol{X} = X_A \boldsymbol{\theta}_A^{(B)} + X_B \boldsymbol{\theta}_B^{(B)}$$

Since both the data generation process and the learned model are linear, it is sufficient to minimize the linear interventional dependence between representations instead of the full statistical dependence that we described in Sec. 3.2. Following the definition of HSIC (Gretton et al., 2005), the linear dependence in interventional features can be defined as follows[6],

$$\operatorname{Dep}\left(\boldsymbol{F}_A, \boldsymbol{F}'_B\right) = \left\| \mathbb{E}_{P_{\text{int}}} \left[\boldsymbol{F}_A \boldsymbol{F}_B'^{\top} \right] \right\|_F^2 \tag{5}$$

Leveraging the independence relations during interventions, we can expand Eq. (5) as,

$$\left\| \mathbb{E}_{P_{\text{int}}} \left[\boldsymbol{F}_A \boldsymbol{F}_B'^{\top} \right] \right\|_F^2 = \left\| (w_A^2 \rho_A^2 + \rho_{\boldsymbol{U}_A}^2) \boldsymbol{\theta}_A^{(A)} \boldsymbol{\theta}_A^{(B)\top} + (w_B^2 \rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2) \boldsymbol{\theta}_B^{(A)} \boldsymbol{\theta}_B^{(B)\top} \right\|_F^2 \tag{6}$$

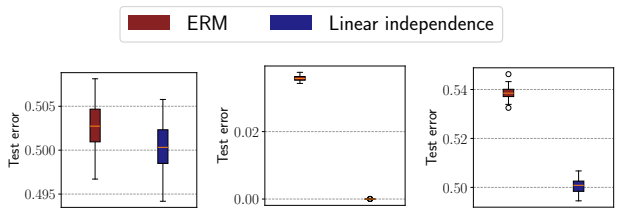
The dependence loss is thus the Frobenius norm of a sum of rank-one matrices. There are three classes of solutions that minimize Eq. (6): (1) $\boldsymbol{\theta}_A^{(A)} = \boldsymbol{\theta}_B^{(A)} = \boldsymbol{\theta}_A^{(B)} = \boldsymbol{\theta}_B^{(B)} = \mathbf{0}$, (2) $\boldsymbol{\theta}_A^{(A)} = \pm\gamma \boldsymbol{\theta}_B^{(A)}$ and $\gamma \boldsymbol{\theta}_A^{(B)} = \mp \boldsymbol{\theta}_B^{(B)}$

[6]For a complete definition of the dependence, refer to App. B.3.

for some scalar $\gamma \neq 0$, and (3) $\boldsymbol{\theta}_A^{(A)} = \mathbf{0}$ or $\boldsymbol{\theta}_A^{(B)} = \mathbf{0}$, and $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}$ or $\boldsymbol{\theta}_B^{(B)} = \mathbf{0}$. However, all except two of these solutions produce trivial features and increase the classification error. The only remaining non-degenerate solutions are: (S1) $\boldsymbol{\theta}_A^{(A)} = \mathbf{0}, \boldsymbol{\theta}_B^{(B)} = \mathbf{0}$, and (S2) $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}, \boldsymbol{\theta}_A^{(B)} = \mathbf{0}$. Note that (S2) corresponds to a robust model. Since both (S1) and (S2) minimize Eq. (5), the solution that minimizes the prediction error on both A and B during training will prevail.

Proposition 1. *The total training error for (S1) is strictly greater than that of (S2) when the following conditions are satisfied: (1)* $\beta \geq 1 - \frac{1}{|w_{AB}|}$, *(2)* $\beta \geq \min\left(\frac{\rho_A^2}{2\rho_{B'}^2 + \rho_A^2}, \frac{\rho_{U_A}^2}{w_A^2 w_{AB}^2 \rho_A^2}\right)$.

Proposition 1 states that a robust model is guaranteed when a certain minimum amount of interventional data is available during training. Note that Proposition 1 describes sufficient conditions for (S1) to have a larger training error than (S2). In other words, this result conveys that, given a certain proportion of interventional data in the training set, explicitly enforcing independence between learned representations can provably improve robustness against interventional distributional shifts. More importantly, it *does not* mean the contrary for smaller values of β. In practice, β could be smaller. For instance, in Tab. 2, explicitly enforcing independence using our proposed approach improves robustness even for $\beta = 1\%$. Refer to App. B.3 for a detailed derivation and experimental verification of Proposition 1.



(a) $E_A^{(1)}$ (b) $E_A^{(2)}$ (c) E_A

Figure 4: Robust models achieve $E_A^{(2)} = 0$ in Eq. (1). ERM models have a non-zero $\boldsymbol{\theta}_B^{(A)}$ resulting in $E_A^{(2)} \neq 0$. Minimizing linear independence on interventional features results in orthogonal interventional feature spaces where $\boldsymbol{\theta}_B^{(A)} = \boldsymbol{\theta}_A^{(B)} = \mathbf{0}$. Thus, they result in robust models with $E_A^{(2)} = 0$.

Experimental verification: To experimentally verify the theoretical results, we simulate the causal model by setting $w_A = w_B = w_{AB} = 1$. The random variables A, B, U_A, and U_B are sampled from independent normal distributions with zero mean and unit variance. We generate $N = 50000$ data points for training with $\beta = 0.5$. The classifiers use 2-dimensional features learned by linear feature extractors to predict A and B. The experiment is repeated with 50 seeds. In Eq. (1), the statistical risk was shown to be composed of $E_A^{(1)}$ and $E_A^{(2)}$, plotted in Figs. 4a and 4b respectively. An ideal robust model will achieve $E_A^{(2)} = 0$. As expected, both models have similar $E_A^{(1)}$. However, linear independence models minimize $E_A^{(2)}$, resulting in a lower total error E_A shown in Fig. 4c.

4 RepLIn: Enforcing Statistical Independence between Interventional Features

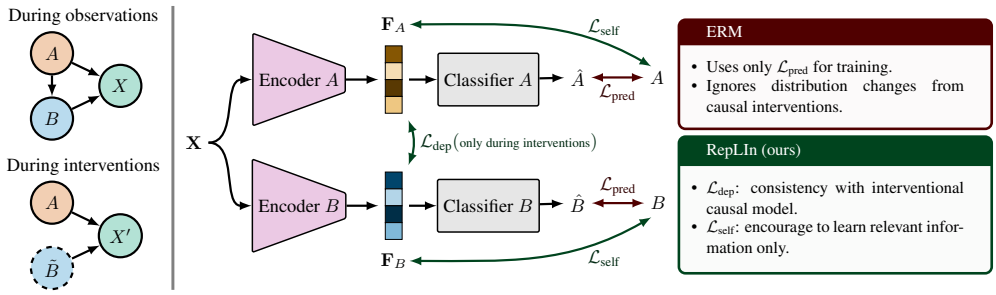


Figure 5: Schematic illustration of **RepLIn** for a causal graph with two attributes ($A \to B$) and $\boldsymbol{X} = f(A, B, \boldsymbol{U})$. Encoders learn representations $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$ corresponding to A and B, which are then used by their corresponding classifiers to predict $\hat{A}$ and $\hat{B}$ respectively. On interventional samples, we minimize $\mathcal{L}_{\text{dep}}$ between the features to ensure their independence. On all samples, we minimize $\mathcal{L}_{\text{self}}$ to encourage the representations to learn only the relevant information.

As noted in the previous section, there is a strong correlation between the drop in accuracy during interventions and interventional feature dependence. We also showed theoretically that minimizing linear dependence between interventional features can improve test time error on interventional data for linear models. Based on this observation, we propose "Representation Learning from Interventional data" (RepLIn) to learn discriminative representations that are robust against interventional distribution shifts.

To enforce independence between interventional features, we propose to use dependence-guided regularization denoted as $\mathcal{L}_{\text{dep}}$ over the prediction loss function (e.g., cross-entropy for classification tasks) used in ERM. We refer to this regularization as "dependence loss" and is defined for the general case in Sec. 3 as $\mathcal{L}_{\text{dep}} = \sum_{i=1}^{n} \text{NHSIC}(\boldsymbol{F}_{A_i}^{\text{int}}, \boldsymbol{F}_{B}^{\text{int}})$. We minimize the dependence loss *only* for the interventional samples in our training set since congruent statistical independence occurs in the data space only during interventions.

However, $\mathcal{L}_{\text{dep}}$ alone is insufficient since learning irrelevant features can minimize $\mathcal{L}_{\text{dep}}$. To avoid such pathological scenarios and encourage the model to learn only relevant information, we introduce another loss that maximizes the dependence between a feature and its corresponding label. We employ this "self-dependence loss" on both observational and interventional data and define it as $\mathcal{L}_{\text{self}} = 1 - \frac{\text{NHSIC}(\boldsymbol{F}_B, B) + \sum_{i=1}^{n} \text{NHSIC}(\boldsymbol{F}_{A_i}, A_i)}{2(n+1)}$. Employing $\mathcal{L}_{\text{self}}$ in addition to $\mathcal{L}_{\text{pred}}$ ensures that the representations contain as much information about the modeled latent variables and not just the information required to predict the given downstream task. In contrast to $\mathcal{L}_{\text{dep}}$, we use linear kernels in $\mathcal{L}_{\text{self}}$ to maximize a lower estimate of the dependence between the representations and the labels. Using linear kernels in HSIC amounts to $k_P\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \boldsymbol{x}^{(i)\top}\boldsymbol{x}^{(j)}$ in Sec. 3.2. Since kernel approaches typically require much computation and memory, we use random Fourier features (Rahimi & Recht, 2007) to compute NHSIC values.

In summary, RepLIn optimizes the following total loss: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{dep}}\mathcal{L}_{\text{dep}} + \lambda_{\text{self}}\mathcal{L}_{\text{self}}$, where λ_{dep} and λ_{self} are weights that control the contribution of the respective losses. The impact of the choice of these hyperparameters is discussed in App. H. A pictorial overview of the RepLIn pipeline is shown in Fig. 5.

5 Experimental Evaluation

In this section, we compare the performance of RepLIn to the baselines on a synthetic dataset (Windmill) and real image and text datasets (CelebA and CivilComments). We use the Windmill dataset introduced in Sec. 3.1 to verify the effectiveness of RepLIn and evaluate its broader applicability to practical scenarios through the facial attribute prediction task on the CelebA dataset. Our experiments are designed to validate the following hypothesis: *Does explicitly minimizing the interventional feature dependence improve interventional accuracy?*

Training Hyperparameters and Baselines: We consider vanilla **ERM** and **ERM-Resampled** (Chawla et al., 2002; Cateni et al., 2014) as our primary baselines since they are the most commonly used training algorithms. Additionally, ERM-Resampled has been shown to be a strong baseline for group-imbalanced training and domain generalization (Idrissi et al., 2022; Gulrajani & Lopez-Paz, 2021). On Windmill dataset, we also consider the following SOTA algorithms in domain generalization: **IRMv1** (Arjovsky et al., 2019), **Fish** (Shi et al., 2022), **GroupDRO** (Sagawa et al., 2020), **SAGM** (Wang et al., 2023), **DiWA** (Rame et al., 2022), and **TEP** (Qiao & Peng, 2024). The latter two are weight-averaging methods, for which we train 20 independent models per seed. We study two variants of our method: RepLIn and RepLIn-Resampled. The latter variant uses the resampling strategy from ERM-Resampled. In each method, attribute-specific representations are extracted from the input data, which feed into the classifiers to get the final prediction. Since ICRL methods learn attribute-identifiable representations only up to permutation invariance, and therefore, cannot be used with attribute-specific classifiers, we do not include them as baselines. All baselines use the same architecture to learn representations and linear layers to make the final prediction from these representations. The values of λ_{dep} and λ_{self} in RepLIn variants are tuned and kept fixed for all values of β. A detailed description of the datasets and the training settings is provided in App. A.

Evaluation Metrics: Our primary interest is in investigating the accuracy drop when predicting the variables that are unaffected by interventions. Ideally, if the learned features respect causal relations during interventions, we expect no change in the prediction accuracy of parent variables of the intervened variable between observational and interventional distributions. To measure the change, we use the relative drop in accuracy defined in Sec. 3.3: $\text{Rel.}\Delta = \frac{\text{Obs acc.} - \text{Int acc.}}{\text{Obs acc.}}$. Since we optimize NHSIC during training, we use NKCC from Sec. 3.3 to evaluate the dependence between the features on interventional data during testing. We repeat each experiment with five different random seeds and report the mean and standard deviation.

5.1 Windmill dataset

We first evaluate our method on Windmill dataset that helped us identify the relation between the performance gap in predicting A on observational and interventional data in Sec. 3.1. As a reminder, the causal graph consists of two binary random variables A and B, where $A \to B$ during observations. We intervene by setting $B \sim \text{Bernoulli}(0.5)$, breaking the dependence between A and B. The proportion of interventional samples in the training data varies from $\beta = 0.01$ to $\beta = 0.5$.

Accuracy on interventional data. The relative drop in accuracy is shown in parentheses.

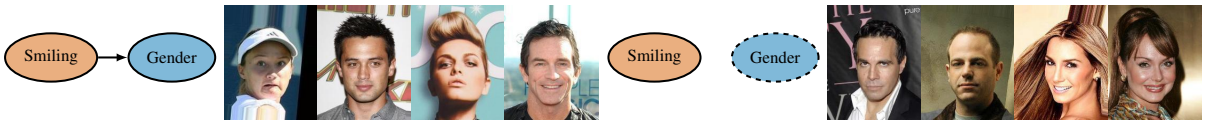
Method	$\beta = 0.5$		$\beta = 0.3$		$\beta = 0.1$		$\beta = 0.05$		$\beta = 0.01$	
ERM	$76.87_{\pm 1.08}$	$(0.18_{\pm 0.01})$	$69.86_{\pm 3.19}$	$(0.29_{\pm 0.04})$	$62.78_{\pm 1.77}$	$(0.37_{\pm 0.02})$	$59.52_{\pm 1.30}$	$(0.40_{\pm 0.01})$	$60.15_{\pm 3.12}$	$(0.40_{\pm 0.03})$
ERM-Res.	$73.70_{\pm 3.19}$	$(0.22_{\pm 0.04})$	$71.19_{\pm 3.23}$	$(0.24_{\pm 0.03})$	$73.62_{\pm 1.54}$	$(0.22_{\pm 0.02})$	$71.03_{\pm 2.83}$	$(0.25_{\pm 0.03})$	$70.20_{\pm 3.73}$	$(0.26_{\pm 0.03})$
IRMv1	$78.24_{\pm 0.79}$	$(0.16_{\pm 0.01})$	$74.83_{\pm 1.74}$	$(0.20_{\pm 0.02})$	$78.61_{\pm 2.24}$	$(0.16_{\pm 0.02})$	$76.28_{\pm 1.87}$	$(0.18_{\pm 0.02})$	$71.75_{\pm 2.03}$	$(0.24_{\pm 0.02})$
Fish	$77.23_{\pm 2.24}$	$(0.19_{\pm 0.02})$	$77.23_{\pm 1.32}$	$(0.19_{\pm 0.01})$	$78.24_{\pm 2.09}$	$(0.18_{\pm 0.02})$	$76.42_{\pm 1.95}$	$(0.20_{\pm 0.02})$	$\mathbf{73.92_{\pm 2.53}}$	$(0.23_{\pm 0.03})$
GroupDRO	$80.10_{\pm 1.66}$	$(\mathbf{0.02_{\pm 0.01}})$	$80.96_{\pm 1.33}$	$(\mathbf{0.04_{\pm 0.02}})$	$80.35_{\pm 1.01}$	$(\mathbf{0.06_{\pm 0.02}})$	$\mathbf{77.40_{\pm 1.16}}$	$(\mathbf{0.08_{\pm 0.01}})$	$71.86_{\pm 1.60}$	$(\mathbf{0.22_{\pm 0.02}})$
SAGM	$76.43_{\pm 2.37}$	$(0.19_{\pm 0.02})$	$79.05_{\pm 2.23}$	$(0.17_{\pm 0.02})$	$76.96_{\pm 4.36}$	$(0.18_{\pm 0.03})$	$79.86_{\pm 1.81}$	$(0.16_{\pm 0.02})$	$72.81_{\pm 3.10}$	$(0.23_{\pm 0.03})$
DiWA	$76.61_{\pm 2.15}$	$(0.19_{\pm 0.02})$	$76.71_{\pm 0.59}$	$(0.19_{\pm 0.01})$	$76.09_{\pm 0.69}$	$(0.20_{\pm 0.01})$	$75.83_{\pm 1.83}$	$(0.20_{\pm 0.02})$	$73.39_{\pm 1.31}$	$(\mathbf{0.22_{\pm 0.01}})$
TEP	$58.68_{\pm 4.72}$	$(0.06_{\pm 0.19})$	$60.42_{\pm 1.30}$	$(0.09_{\pm 0.06})$	$56.07_{\pm 3.35}$	$(-0.04_{\pm 0.42})$	$58.52_{\pm 4.36}$	$(0.01_{\pm 0.25})$	$59.23_{\pm 1.13}$	$(0.18_{\pm 0.11})$
RepLIn	$\mathbf{87.94_{\pm 1.46}}$	$(0.08_{\pm 0.02})$	$\mathbf{87.76_{\pm 2.30}}$	$(0.10_{\pm 0.02})$	$\mathbf{83.23_{\pm 2.67}}$	$(0.16_{\pm 0.03})$	$73.63_{\pm 2.43}$	$(0.25_{\pm 0.02})$	$67.52_{\pm 2.30}$	$(0.32_{\pm 0.03})$
RepLIn-Res.	$\mathbf{88.46_{\pm 0.96}}$	$(\mathbf{0.07_{\pm 0.01}})$	$\mathbf{88.05_{\pm 1.04}}$	$(\mathbf{0.08_{\pm 0.01}})$	$\mathbf{87.91_{\pm 1.36}}$	$(\mathbf{0.08_{\pm 0.01}})$	$\mathbf{86.38_{\pm 0.85}}$	$(\mathbf{0.10_{\pm 0.01}})$	$\mathbf{78.41_{\pm 1.27}}$	$(\mathbf{0.18_{\pm 0.02}})$

Table 2: **Results on Windmill dataset:** We evaluate the variants of RepLIn (highlighted in green) against the baselines on two metrics: interventional accuracy and relative accuracy drop on interventional data compared to observational. As the proportion of interventional data during training (β) decreases, the problem becomes more challenging. Compared to the baselines, RepLIn maintains its interventional accuracy. A similar trend is observed in the relative accuracy drop, where RepLIn significantly outperforms most baselines. The **best** and the **second-best** results are shown in different colors. "Res." stands for "Resampled".

Tab. 2 compares the interventional accuracy in predicting A for various amounts of interventional training data. We make the following observations: **(1)** RepLIn outperforms every baseline in interventional accuracy for all values of β. This clearly demonstrates the advantage of exploiting the underlying causal relations when learning from interventional data, instead of treating it as a separate domain. **(2)** Comparing ERM and RepLIn with their resampling variants, we observe that resampling is a generally useful technique with large gains when β is very small (for example, consider results with $\beta \leq 0.05$). We are also interested in the relative drop in accuracy between observational and interventional data (Rel.Δ). From Tab. 2, we observe that GroupDRO has the lowest Rel.Δ among the considered methods for $\beta \geq 0.05$, and achieves its best results when more interventional data is available during training. However, this improvement comes at the cost of lower interventional accuracy – over ≈ 7 percentage points difference compared to RepLIn. Meanwhile, the relative drop in accuracy of RepLIn is comparable to GroupDRO at larger values of β and has the least relative drop in accuracy at lower values of β. DiWA and TEP were provided with the same pool of models trained with minor variations in their hyperparameters. We do not consider the negative Rel.Δ of TEP since (1) it achieves very low interventional accuracy, performing barely above random chance, and (2) due to the high standard deviation of Rel.Δ. We discuss in Sec. 6.1 how the representations learned by RepLIn are less affected by interventional shifts. As mentioned in Sec. 3.1, interventional robustness may be at odds with observational accuracy as removing spurious information from representations may hurt performance on observational data. We provide the results on observational data in App. E.

5.2 Facial Attribute Prediction

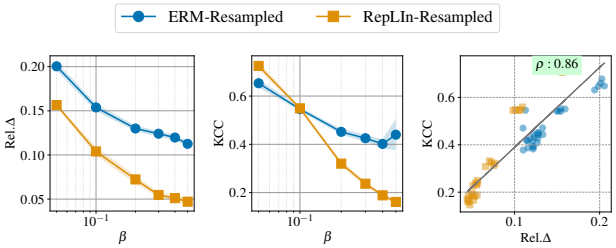
We verify the utility of RepLIn for predicting facial attributes on the CelebA dataset (Liu et al., 2015). Images in the CelebA dataset are annotated with 40 labeled binary attributes. We consider two of these attributes – `smiling` and `gender` – as random variables affecting each other causally. Since the true underlying relation between smile and gender is unknown, we adopt the resampling procedure from (Wang & Boddeti, 2022) to induce a desired causal relation between the attributes (`smiling` $\rightarrow$ `gender`) and obtain samples. Specifically, to simulate this causal relation, we sample `smiling` from Bernoulli(0.6) first and then sample `gender` according to a probability distribution conditioned on the sampled `smiling` variable. We then sample a face image whose attribute labels match the sampled values. We model the diversity in the images due to unobserved noise variables. Note that, unlike in Windmill, the noise variables in this experiment *may be* causally related to the attributes that we wish to predict, adding to the challenges in the dataset. The causal model for this experiment and some sample images are shown in Fig. 7. We first extract features from



(a) Observational causal graph and samples (b) Interventional causal graph and samples

Figure 7: Causal model for CelebA before and after intervention along with sample images from these models

the face images using a ResNet-50 (He et al., 2016) model pre-trained on the ImageNet dataset (Deng et al., 2009). Then, similar to the architecture used for the Windmill experiments, we employ a shallow MLP to act on these features, followed by a linear classifier to predict the attributes. Our loss functions act upon the features of the MLP. We use 30,000 samples for training and 15,000 for testing. We use the relative drop in interventional accuracy as the primary metric and compare RepLIn-Resampled against ERM-Resampled. We also verify if the correlation between interventional feature dependence and the relative drop in accuracy observed in Sec. 3.3 on Windmill experiments holds in a more practical scenario.



(a) Accuracy (b) Dependence (c) Correlation

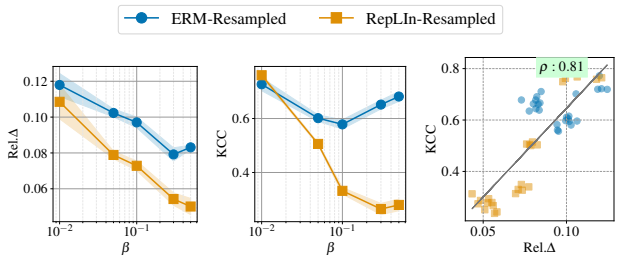
Figure 8: **Facial Attribute Prediction:** (a) RepLIn has a lower relative drop in accuracy compared to ERM-Resampled. (b) Minimizing interventional feature dependence during training generalizes to testing. (c) Interventional feature dependence correlates positively with the relative drop in accuracy.

Fig. 8 reports the experimental results on facial attribute prediction for various amounts of interventional training data. We make the following observations: **(1)** as the proportion of interventional data increases, the relative drop in accuracy in all methods decreases, **(2)** across all proportions of interventional data, RepLIn consistently outperforms the baseline by $4\% - 7\%$ lower relative drop in accuracy despite the potential challenges due to noise variable being causally related to the attributes of interest, **(3)** relative drop in accuracy and interventional feature dependence show strong positive correlation ($\rho = 0.86$), and **(4)** the interventional feature dependence of RepLIn steadily decreases as the amount of interventional data increases.

5.3 Toxicity Prediction in Text

We further evaluate RepLIn on a text classification task on the CivilComments dataset (Borkan et al., 2019). CivilComments consists of comments from online forums, and we use a subset of this dataset labeled with identity attributes (such as "Male", "White", "LGBTQ", etc.) and toxicity scores by humans. The task is to classify each comment as toxic or not. Previous works have identified gender bias in toxicity classifier

models (Dixon et al., 2018; Park et al., 2018; Nozza et al., 2019). Therefore, we will simulate a causal model in the training dataset between the attribute "female" and toxicity, similar to Sec. 5.2. During observation, both attributes assume the same binary value. During interventions, toxicity takes value independent of "female". Input text comments are sampled according to these attributes. Similar to our facial attribute prediction experiments, we first extract features from the comments using BERT (Devlin et al., 2019) and train the models on these features. Our model architecture consists of a linear layer to learn representations and a linear layer to predict toxicity.



(a) Accuracy (b) Dependence (c) Correlation

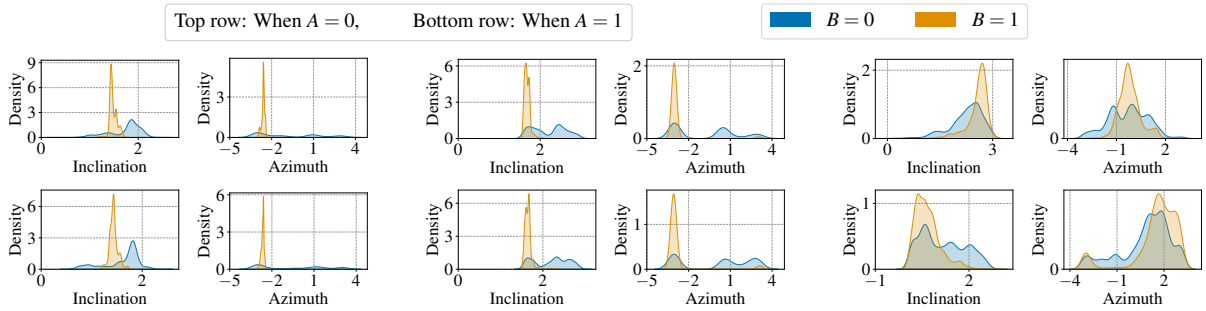
Figure 9: **Toxicity Prediction in Text:** (a) RepLIn has lower interventional accuracy drop compared to ERM-Resampled; (b) Minimizing $\mathcal{L}_{\text{dep}}$ during training gives us representations that are independent during interventions; (c) The strong correlation between accuracy drop and interventional feature dependence further corroborates our hypothesis in Sec. 3.2.

Fig. 9 compares the performance of RepLIn against ERM-Resampled. Fig. 9b shows that enforcing independence between interventional features minimizes the interventional feature dependence during testing, although its effectiveness drops as β approaches 0.01. Yet, RepLIn outperforms the baseline in terms of the accuracy drop during interventions (Fig. 9a). We also note that RepLIn becomes increasingly efficient in minimizing the interventional feature dependence as β increases.

6 Discussion

6.1 How are representations learned by RepLIn different from those by ERM?

In this section, we qualitatively and quantitatively inspect the differences between the interventional features learned by RepLIn and baselines to understand how RepLIn improves robustness against interventional distribution shift.



(a) ERM (b) ERM-Resampled (c) RepLIn-Resampled

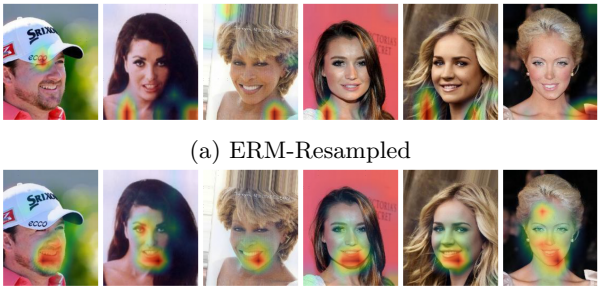
Figure 10: Visualization of interventional features learned by various methods on WINDMILL dataset.

Windmill dataset: Robust representations of A change with A but not B. We quantify this change using the Jensen-Shannon (JS) divergence between the distributions of $\boldsymbol{F}_A^{\text{int}}$ for a fixed value of A and changing values of B. Tab. 3 shows the JS divergence between $P(\boldsymbol{F}_A^{\text{int}}|B = 0, A = a)$ and $P(\boldsymbol{F}_A^{\text{int}}|B = 1, A = a)$ (obtained through binning) for multiple baselines trained on WINDMILL dataset. JS divergence for an ideal robust model must be zero. We observe that $\boldsymbol{F}_A^{\text{int}}$ learned by RepLIn achieves the lowest JS divergence, indicating that $\boldsymbol{F}_A^{\text{int}}$ learned by RepLIn contains the least information about B among the baselines.

Method	ERM	ERM-Resampled	IRMv1	Fish	GroupDRO	RepLIn	RepLIn-Resampled
When $A = 0$	0.45 ± 0.058	0.423 ± 0.105	0.333 ± 0.122	0.341 ± 0.111	0.365 ± 0.066	$\mathbf{0.15 \pm 0.03}$	$\mathbf{0.188 \pm 0.032}$
When $A = 1$	0.499 ± 0.07	0.456 ± 0.11	0.405 ± 0.111	0.37 ± 0.116	0.431 ± 0.048	$\mathbf{0.183 \pm 0.058}$	$\mathbf{0.168 \pm 0.047}$
Average	0.475 ± 0.063	0.439 ± 0.105	0.369 ± 0.116	0.355 ± 0.113	0.398 ± 0.055	$\mathbf{0.166 \pm 0.035}$	$\mathbf{0.178 \pm 0.036}$

Table 3: **Jensen-Shannon (JS) divergence:** The distribution of $\boldsymbol{F}_A^{\text{int}}$ must be invariant to the value assumed by B since $A \perp\!\!\!\perp B$ during interventions. Therefore, JS divergence between $P(\boldsymbol{F}_A^{\text{int}}|B = 0, A = a)$ and $P(\boldsymbol{F}_A^{\text{int}}|B = 1, A = a)$ of a robust model must be zero, for $a \in \{0, 1\}$. We compare the JS divergence between interventional features of the baselines for $\beta = 0.5$. Among the baselines, RepLIn achieves the lowest values of Jensen-Shannon divergence. The lowest and the second lowest scores are highlighted in color.

We can qualitatively examine the learned representations of the baselines by visualizing the spherical angles subtended by the 3-dimensional features on a unit radius sphere. We compare the distributions of inclination and azimuth angles of $\boldsymbol{F}_A^{\text{int}}$ learned by RepLIn-Resampled against the ERM baselines in Fig. 10. Each row shows the distribution of the spherical angles for different values of A. Distributions for different values of B have separate colors. These feature distributions for a robust model must change with A but not B. We observed from the figure that the feature distributions of the baselines are affected by B and not A due to the dependence between $\boldsymbol{F}_A^{\text{int}}$ and B. However, the feature distributions learned by RepLIn change with A and overlap significantly when B takes different values. Thus, models learned by RepLIn perform more similarly to a robust model. Visualizations of the feature distributions of other baselines are provided in App. G.



(a) ERM-Resampled

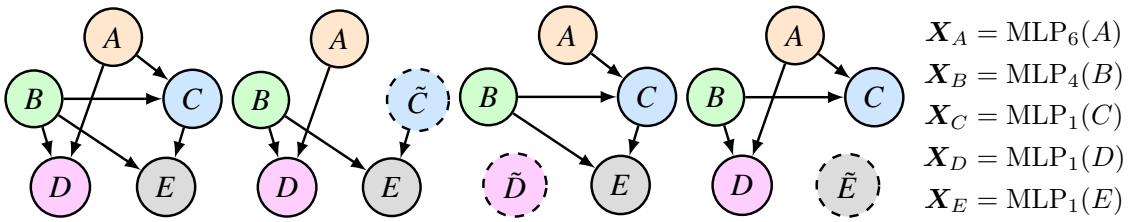
(b) RepLIn-Resampled

Figure 11: Consider these sample face images where the subjects are smiling. The ERM baseline misclassified these samples as not smiling, while RepLIn classified them correctly. We use GradCAM visualizations to identify the regions in the input images that the models used to make their predictions. The ERM model relied on factors such as hair and the presence of a hat that may correlate with gender to predict whether the subjects are smiling. In contrast, RepLIn attended to the lip regions to make predictions.

CelebA dataset: To analyze the high-dimensional features learned on CelebA, we employ Grad-CAM (Selvaraju et al., 2017) and inspect their output attention maps. We consider some samples with `smiling` $= 1$ that were misclassified by ERM-Resampled but were correctly classified by RepLIn-Resampled. Fig. 11 shows the attention maps from models trained on datasets with 50% interventional data. A robust model would attend to facial regions near the lips to make predictions about smiling. Observe that RepLIn-Resampled tends to focus more on the region around the lips (associated with smiling) while ERM-Resampled attends to other regions of the image, such as hair and cap. This supports the trustworthiness of representations learned by RepLIn. More GradCAM visualizations, including the samples accurately classified by ERM, but not RepLIn are shown in App. F.

6.2 Scalability with number of nodes

Practical causal graphs can include many latent variables. The variable for which we wish to learn robust representations may have several child nodes, depending on the density of the causal graph. Therefore, it is imperative that RepLIn is scalable with both the number of intervened nodes and their parents. To verify this scalability, we use the causal graph shown in Fig. 12a with five latent variables. It consists of two binary source nodes A and B, and three binary derived nodes C, D, and E. During observations, A and B are sampled from independent Bernoulli(0.5) distributions. During observation, the remaining nodes take the following logical expressions: $C := A$ `or` B, $D := A$ `and` B, and $E :=$ `not` B `and` C. Like our previous experiments, the training dataset has interventional data samples collected by intervening on nodes



(a) Observational (b) Intervening on C (c) Intervening on D (d) Intervening on E (e) Generating $\boldsymbol{X}$

Figure 12: **5-variable causal graph**: We construct a 5-variable causal graph to demonstrate the scalability of our method with the number of nodes. To collect interventional data, we intervene on C, D, and E separately and measure the performance drop in predicting A and B during these interventions. Nodes in the graphs with dashed borders indicate intervened nodes. Note that we do not intervene on multiple targets at a time. The input data signal $\boldsymbol{X}$ is constructed as a concatenation of individual input signals, each being a function of a latent variable, i.e., $\boldsymbol{X} = \begin{bmatrix}\boldsymbol{X}_A^\top & \boldsymbol{X}_B^\top & \boldsymbol{X}_C^\top & \boldsymbol{X}_D^\top & \boldsymbol{X}_E^\top\end{bmatrix}^\top$ Here, MLP_l indicates a randomly initialized MLP with l linear layers, each followed by a ReLU. We also add Gaussian noise sampled from $\mathcal{N}(0, 0.01)$ to the output of the MLP.

C, D, and E separately, in addition to the observational data. The changes in the causal graph due to these interventions are shown in Figs. 12b to 12d. Each intervened variable assumes values from a Bernoulli(0.5) distribution independent of their parents. Each latent variable $*$ is passed through a randomly initialized MLP with noise added to its output to get a corresponding observed signal $\boldsymbol{X}_*$. These individual signals are concatenated to obtain the observed input signal $\boldsymbol{X}$, as shown in Fig. 12e. The task is to predict the latent variables from the input signal $\boldsymbol{X}$.

Each training batch comprises only observational or interventional data after intervention on a single target. Therefore, our method only enforces the independence relations from at most one interventional target in each batch. The validation and test sets consist of samples collected during interventions on C, D, or E. Since we are interested in the robustness of the model against interventional distribution shift, our primary metrics will be the predictive accuracy for A and B during interventions on C, D, and E.

Interventional target	Method	Predictive accuracy on A				Predictive accuracy on B			
		$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.05$	$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.05$
C	ERM-Resampled	79.71 ± 0.30	76.22 ± 0.42	$\mathbf{73.97 \pm 0.39}$	73.56 ± 0.36	87.60 ± 0.06	85.45 ± 0.23	$\mathbf{83.89 \pm 0.33}$	$\mathbf{83.71 \pm 0.40}$
	RepLIn-Resampled	$\mathbf{95.37 \pm 0.97}$	$\mathbf{78.77 \pm 0.54}$	72.15 ± 0.31	$\mathbf{73.74 \pm 0.36}$	$\mathbf{96.72 \pm 0.81}$	$\mathbf{86.16 \pm 0.63}$	82.35 ± 0.95	82.43 ± 0.65
D	ERM-Resampled	79.65 ± 0.43	75.47 ± 0.64	$\mathbf{71.76 \pm 0.35}$	$\mathbf{70.27 \pm 0.34}$	91.05 ± 0.29	90.21 ± 0.27	90.36 ± 0.58	90.55 ± 0.74
	RepLIn-Resampled	$\mathbf{95.49 \pm 1.01}$	$\mathbf{77.76 \pm 0.82}$	71.20 ± 0.82	68.80 ± 0.79	$\mathbf{97.87 \pm 0.31}$	$\mathbf{92.21 \pm 0.48}$	$\mathbf{91.40 \pm 0.79}$	$\mathbf{90.88 \pm 0.89}$
E	ERM-Resampled	86.63 ± 0.33	81.90 ± 0.26	$\mathbf{76.20 \pm 0.84}$	$\mathbf{73.46 \pm 0.37}$	81.12 ± 0.22	78.00 ± 0.48	$\mathbf{74.02 \pm 0.38}$	$\mathbf{72.97 \pm 0.38}$
	RepLIn-Resampled	$\mathbf{96.71 \pm 0.49}$	$\mathbf{84.68 \pm 0.36}$	75.01 ± 0.53	71.52 ± 0.87	$\mathbf{96.89 \pm 0.68}$	$\mathbf{80.88 \pm 0.57}$	72.81 ± 1.13	71.60 ± 0.59

Table 4: **Results on 5-variable causal graph:** We compare the accuracy of RepLIn in predicting the source nodes A and B during interventions on non-source nodes C, D, and E against that of ERM-Resampled. Our approach outperforms the baselines with sufficient interventional data.

Observations: The predictive performances on the test sets are reported in Table 4. We observe that RepLIn significantly improves over the baseline with sufficient interventional data, $\beta > 0.1$. When the proportion of interventional data $\beta \leq 0.1$, RepLIn is comparable with the baseline, suggesting that the benefits of enforcing independence between interventional features extend to larger causal graphs with multiple intervention targets.

6.3 Limitations

RepLIn requires knowledge of interventional data, the intervened node, and its parent variables. RepLIn could be sensitive to inaccurate knowledge about any of these, or lack thereof. For instance, interventional data can be challenging to obtain in safety-critical applications such as drug testing and autonomous driving. In such cases, generative models that accurately model the data generation process could be used to generate

synthetic interventional data. There may be scenarios where only imperfect interventions are possible. During imperfect interventions, the intervened variable would still be partially dependent on its parents, although the strength of this dependence would be lower. We believe that our proposed approach would still be useful in such cases, although the exact outcome would depend on the strength of the post-interventional dependence between the variables and other factors such as the support of the intervened node. RepLIn could also be sensitive to causal graph misspecification involving the intervened node. However, we are only concerned about the misspecification where the edge from a parent to the intervened node is reversed. This misspecification would result in an independent constraint not enforced during training.

7 Conclusion

This paper considered the problem of learning representations that are robust against interventional distribution shifts and proposed a training algorithm for this objective that exploits the statistical independence induced by interventions in the underlying data-generating process. First, we established a strong correlation between the drop in accuracy during interventions and statistical dependence between representations on interventional data. We then showed theoretically that minimizing linear dependence between interventional representations can improve the robustness of a linear model against interventional distribution shift. Building on this result, we proposed RepLIn to learn representations that are robust against interventional distribution shift by explicitly enforcing statistical independence between learned representations on interventional data. Experimental evaluation of RepLIn across different causal graphs on both synthetic and real datasets on image and text modalities showed that RepLIn can improve predictive accuracy during interventions for various proportions of interventional data. RepLIn is also scalable to the number of causal attributes and can be used with continuous and discrete latent variables. We used qualitative and quantitative tools to show that RepLIn is more successful in learning interventional representations that do not contain information about their child nodes during interventions.

References

Kartik Ahuja, Jason S Hartford, and Yoshua Bengio. Weakly supervised representation learning with sparse perturbations. In *Advances in Neural Information Processing Systems*, 2022a. URL `https://papers.nips.cc/paper_files/paper/2022/hash/63d3bae2c1f525745003f679e45bcf7b-Abstract-Conference.html`.

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

Kartik Ahuja, Yixin Wang, Divyat Mahajan, and Yoshua Bengio. Interventional causal representation learning. In *International Conference on Machine Learning*, 2023.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. URL `https://arXiv.org/abs/1907.02893`.

Francis R Bach and Michael I Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3(Jul):1–48, 2002.

Simon Bing, Urmi Ninad, Jonas Wahl, and Jakob Runge. Identifying Linearly-Mixed Causal Representations from Multi-Node Interventions. In *Causal Learning and Reasoning*, 2024.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *ACM Web Conference*, 2019.

Johann Brehmer, Pim De Haan, Phillip Lippe, and Taco Cohen. Weakly supervised causal representation learning. In *Advances in Neural Information Processing Systems*, 2022.

Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. In *Advances in Neural Information Processing Systems*, 2020.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.

Simon Buchholz, Goutham Rajendran, Elan Rosenfeld, Bryon Aragam, Bernhard Schölkopf, and Pradeep Ravikumar. Learning Linear Causal Representations from Interventions under General Nonlinear Mixing. In *Advances in Neural Information Processing Systems*, 2023.

Silvia Cateni, Valentina Colla, and Marco Vannucci. A method for resampling imbalanced datasets in binary classification tasks for real-world problems. *Neurocomputing*, 135:32–41, 2014.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13:795–828, 2012.

Nello Cristianini, John Shawe-Taylor, Andre Elisseeff, and Jaz Kandola. On kernel-target alignment. In *Advances in Neural Information Processing Systems*, 2001.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

Wenhao Ding, Haohong Lin, Bo Li, and Ding Zhao. Generalizing goal-conditioned reinforcement learning with variational causal reasoning. In *Advances in Neural Information Processing Systems*, 2022a.

Yu Ding, Lei Wang, Bin Liang, Shuming Liang, Yang Wang, and Fang Chen. Domain Generalization by Learning and Removing Domain-specific Features. In *Neural Information Processing Systems*, 2022b.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Measuring and mitigating unintended bias in text classification. In *AAAI/ACM Conference on AI, Ethics, and Society*, 2018.

Frederick Eberhardt, Clark Glymour, and Richard Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In *Conference on Uncertainty in Artificial Intelligence*, 2005.

Tongtong Fang, Nan Lu, Gang Niu, and Masashi Sugiyama. Rethinking importance weighting for deep learning under distribution shift. In *Advances in Neural Information Processing Systems*, 2020.

Irena Gao, Shiori Sagawa, Pang Wei Koh, Tatsunori Hashimoto, and Percy Liang. Out-of-domain robustness via targeted augmentations. In *International Conference on Machine Learning*, 2023.

Maxime Gasse, Damien Grasset, Guillaume Gaudron, and Pierre-Yves Oudeyer. Causal reinforcement learning using observational and interventional data. *arXiv preprint arXiv:2106.14421*, 2021.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International Conference on Algorithmic Learning Theory*, 2005.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A Kernel Two-Sample Test. *Journal of Machine Learning Research*, 13(1):723–773, 2012.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=lQdXeXDoWtI`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

Christina Heinze-Deml and Nicolai Meinshausen. Conditional variance penalties and domain shift robustness. *Machine Learning*, 110(2):303–348, 2021.

Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Aapo Hyvarinen, Hiroaki Sasaki, and Richard Turner. Nonlinear ica using auxiliary variables and generalized contrastive learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 2019.

Aapo Hyvärinen, Ilyes Khemakhem, and Ricardo Monti. Identifiability of latent-variable and structural-equation models: from linear to nonlinear. *Annals of the Institute of Statistical Mathematics*, 76(1):1–33, 2024.

Badr Youbi Idrissi, Martin Arjovsky, Mohammad Pezeshki, and David Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. In *Conference on Causal Learning and Reasoning*, 2022. URL `https://proceedings.mlr.press/v177/idrissi22a.html`.

Yibo Jiang and Bryon Aragam. Learning nonparametric latent causal graphs with unknown interventions. In *Advances in Neural Information Processing Systems*, 2023.

Nan Rosemary Ke, Olexa Bilaniuk, Anirudh Goyal, Stefan Bauer, Hugo Larochelle, Bernhard Schölkopf, Michael C Mozer, Chris Pal, and Yoshua Bengio. Learning neural causal models from unknown interventions. *arXiv preprint arXiv:1910.01075*, 2019.

Nan Rosemary Ke, Silvia Chiappa, Jane Wang, Jorg Bornschein, Theophane Weber, Anirudh Goyal, Matthew Botvinic, Michael Mozer, and Danilo Jimenez Rezende. Learning to induce causal structure. In *International Conference on Learning Representattions*, 2022.

Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

David Klindt, Lukas Schott, Yash Sharma, Ivan Ustyuzhaninov, Wieland Brendel, Matthias Bethge, and Dylan Paiton. Towards nonlinear disentanglement in natural data with temporal sparse coding. In *International Conference on Learning Representations*, 2021.

Lingjing Kong, Shaoan Xie, Weiran Yao, Yujia Zheng, Guangyi Chen, Petar Stojanov, Victor Akinwande, and Kun Zhang. Partial identifiability for domain adaptation. In *International Conference on Machine Learning*, 2022. URL `https://proceedings.mlr.press/v162/kong22a.html`.

Karl Krauth, Yixin Wang, and Michael I Jordan. Breaking feedback loops in recommender systems with causal inference. *arXiv preprint arXiv:2207.01616*, 2022.

Sébastien Lachapelle, Pau Rodriguez, Yash Sharma, Katie E Everett, Rémi Le Priol, Alexandre Lacoste, and Simon Lacoste-Julien. Disentanglement via mechanism sparsity regularization: A new principle for nonlinear ica. In *Conference on Causal Learning and Reasoning*, 2022.

Sébastien Lachapelle, Pau Rodríguez López, Yash Sharma, Katie Everett, Rémi Le Priol, Alexandre Lacoste, and Simon Lacoste-Julien. Nonparametric Partial Disentanglement via Mechanism Sparsity: Sparse Actions, Interventions and Sparse Temporal Dependencies. *arXiv preprint arXiv:2401.04890*, 2024.

Phillip Lippe, Taco Cohen, and Efstratios Gavves. Efficient neural causal discovery without acyclicity constraints. In *International Conference on Learning Representations*, 2022a.

Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M Asano, Taco Cohen, and Stratis Gavves. Citris: Causal identifiability from temporal intervened sequences. In *International Conference on Machine Learning*, 2022b.

Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M Asano, Taco Cohen, and Efstratios Gavves. Causal Representation Learning for Instantaneous and Temporal Effects in Interactive Systems. In *International Conference on Learning Representations*, 2023.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *IEEE/CVF International Conference on Computer Vision*, 2015.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, 2019.

Chaochao Lu, Yuhuai Wu, José Miguel Hernández-Lobato, and Bernhard Schölkopf. Invariant causal representation learning for out-of-distribution generalization. In *International Conference on Learning Representations*, 2021.

Huishi Luo, Fuzhen Zhuang, Ruobing Xie, Hengshu Zhu, Deqing Wang, Zhulin An, and Yongjun Xu. A survey on causal inference for recommendation. *The Innovation*, 2024.

Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning*, 2021.

Gemma Elyse Moran, Dhanya Sridhar, Yixin Wang, and David Blei. Identifiable deep generative models via sparse decoding. *Transactions on Machine Learning Research*, 2022. URL `https://openreview.net/forum?id=vd0onGWZbE`.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. Unintended bias in misogyny detection. In *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019.

Ji Ho Park, Jamin Shin, and Pascale Fung. Reducing gender bias in abusive language detection. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Judea Pearl. *Causality*. Cambridge University Press, 2009.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press, 2017.

Fengchun Qiao and Xi Peng. Ensemble pruning for out-of-distribution generalization. In *International Conference on Machine Learning*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Ali Rahimi and Benjamin Recht. Random Features for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems*, 2007.

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization. In *Advances in Neural Information Processing Systems*, 2022.

Alfréd Rényi. On measures of dependence. *Acta Mathematica Academiae Scientiarum Hungarica*, 10:441–451, 1959.

Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv preprint arXiv:2202.06856*, 2022. URL `https://arxiv.org/abs/2202.06856`.

Sorawit Saengkyongam and Ricardo Silva. Learning joint nonlinear effects from single-variable interventions in the presence of hidden confounders. In *Conference on Uncertainty in Artificial Intelligence*, 2020.

Sorawit Saengkyongam, Elan Rosenfeld, Pradeep Ravikumar, Niklas Pfister, and Jonas Peters. Identifying representations for intervention extrapolation. In *International Conference on Learning Representations*, 2024.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations*, 2020.

Axel Sauer and Andreas Geiger. Counterfactual generative networks. In *International Conference on Learning Representations*, 2021.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5): 612–634, 2021.

Antonin Schrab. A Practical Introduction to Kernel Discrepancies: MMD, HSIC & KSD. *arXiv preprint arXiv:2503.04820*, 2025.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE/CVF International Conference on Computer Vision*, 2017.

Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. In *Advances in Neural Information Processing Systems*, 2020.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition workshops*, 2014.

Zheyan Shen, Peng Cui, Tong Zhang, and Kun Kunag. Stable learning via sample reweighting. In *AAAI Conference on Artificial Intelligence*, 2020.

Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *International Conference on Learning Representations*, 2022.

Xiangchen Song, Weiran Yao, Yewen Fan, Xinshuai Dong, Guangyi Chen, Juan Carlos Niebles, Eric Xing, and Kun Zhang. Temporally disentangled representation learning under unknown nonstationarity. In *Advances in Neural Information Processing Systems*, 2023.

Peter Sorrenson, Carsten Rother, and Ullrich Köthe. Disentanglement by nonlinear ica with general incompressible-flow networks (gin). In *International Conference on Learning Representations*, 2020.

Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

Chandler Squires, Anna Seigal, and Caroline Uhler. Linear causal disentanglement via interventions. *arXiv preprint arXiv:2211.16467*, 2022.

Gautam Sreekumar and Vishnu Naresh Boddeti. Spurious correlations and where to find them. In *Spurious Correlations, Invariance and Stability Workshop, International Conference on Machine Learning*, 2023. URL `https://arxiv.org/abs/2308.11043`.

Jithendaraa Subramanian, Yashas Annadani, Ivaxi Sheth, Stefan Bauer, Derek Nowrouzezahrai, and Samira Ebrahimi Kahou. Latent variable models for bayesian causal discovery. In *International Conference on Machine Learning Workshop on Spurious Correlations, Invariance, and Stability*, 2022.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Burak Varıcı, Emre Acartürk, Karthikeyan Shanmugam, Abhishek Kumar, and Ali Tajer. Score-based causal representation learning with interventions. In *Advances in Neural Information Processing Systems Workshop on Causal Representation Learning*, 2023.

Burak Varıcı, Emre Acartürk, Karthikeyan Shanmugam, and Ali Tajer. General identifiability and achievability for causal representation learning. In *International Conference on Artificial Intelligence and Statistics*, 2024a.

Burak Varıcı, Emre Acartürk, Karthikeyan Shanmugam, and Ali Tajer. Linear Causal Representation Learning from Unknown Multi-node Interventions. In *Advances in Neural Information Processing Systems*, 2024b.

Julius Von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. In *Advances in Neural Information Processing Systems*, 2021.

Julius von Kügelgen, Michel Besserve, Wendong Liang, Luigi Gresele, Armin Kekić, Elias Bareinboim, David M Blei, and Bernhard Schölkopf. Nonparametric identifiability of causal representations from unknown interventions. In *Advances in Neural Information Processing Systems*, 2023.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S Yu. Generalizing to Unseen Domains: A Survey on Domain Generalization. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8052–8072, 2022a.

Lan Wang and Vishnu Naresh Boddeti. Do learned representations respect causal relationships? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Pengfei Wang, Zhaoxiang Zhang, Zhen Lei, and Lei Zhang. Sharpness-aware gradient matching for domain generalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Yunxia Wang, Fuyuan Cao, Kui Yu, and Jiye Liang. Efficient causal structure learning from multiple interventional datasets with unknown targets. In *AAAI Conference on Artificial Intelligence*, 2022b.

Xiaojiang Yang, Yi Wang, Jiacheng Sun, Xing Zhang, Shifeng Zhang, Zhenguo Li, and Junchi Yan. Nonlinear ica using volume-preserving transformations. In *International Conference on Learning Representations*, 2021.

Weiran Yao, Yuewen Sun, Alex Ho, Changyin Sun, and Kun Zhang. Learning temporally causal latent processes from general temporal data. In *International Conference on Learning Representations*, 2022.

Kui Yu, Lin Liu, and Jiuyong Li. Learning markov blankets from multiple interventional data sets. *IEEE Transactions on Neural Networks and Learning Systems*, 31(6):2005–2019, 2019.

Jiaqi Zhang, Chandler Squires, Kristjan Greenewald, Akash Srivastava, Karthikeyan Shanmugam, and Caroline Uhler. Identifiability guarantees for causal disentanglement from soft interventions. In *Advances in Neural Information Processing Systems*, 2023.

Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. Causal intervention for leveraging popularity bias in recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

Yujia Zheng, Ignavier Ng, and Kun Zhang. On the identifiability of nonlinear ica: Sparsity and beyond. In *Advances in Neural Information Processing Systems*, 2022.

Xiao Zhou, Yong Lin, Renjie Pi, Weizhong Zhang, Renzhe Xu, Peng Cui, and Tong Zhang. Model agnostic sample reweighting for out-of-distribution learning. In *International Conference on Machine Learning*, 2022.

Roland S Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. In *International Conference on Machine Learning*, 2021.

Appendix

A Implementation details **24**

B Theoretical Motivation for RepLIn **24**

B.1 Statistical Risk in Predicting Interventional Latent Samples 25

B.2 Optimal ERM model 26

B.3 Minimizing Linear Dependence 27

C Review of identifiable causal representation learning **31**

D Differences w.r.t DG/OOD Setting **32**

E Additional Results from Experiments **32**

F More GradCAM Visualization **33**

G Visualization of Feature Distribution Learned on Windmill dataset **34**

H Balancing $\mathcal{L}_{\mathbf{dep}}$ and $\mathcal{L}_{\mathbf{self}}$ during training **34**

H.1 Why do the hyperparameters change between experiments? 37

I Generating Windmill Dataset **39**

A Implementation details

We implement our models using PyTorch (Paszke et al., 2019) and use Adam (Kingma & Ba, 2015) as our optimizer with its default settings. Training hyperparameters for each dataset (such as the number of data points, training epochs, etc.) are shown in Tab. 5. For training stability, we warm up λ_{dep} from 0 to its set value between sN and eN epochs where N is the total number of epochs, and s and e are fractions shown in Tab. 5.

Table 5: List of hyperparameters used for each dataset.

Dataset	#Training samples	Epochs	Batchsize	Initial LR	Scheduler	λ_{dep}	λ_{self}	Start (s)	End (e)
Windmill	40,000	5000	1000	2e-3	MultiStepLR(milestones=[1000], gamma=0.5)	1	1	0.66	0.99
CelebA	30,000	2000	1000	1e-3	MultiStepLR(milestones=[1000], gamma=0.1)	20	2	0.01	0.99

For all methods, we first extract label-specific features from the inputs and pass them through a corresponding classifier to predict the label. The architecture of the feature extractor is the same for all methods on a given dataset, except on the Windmill dataset. The classification layer is a linear layer mapping from feature dimensions to the number of classes. The specific details for each dataset are provided below.

Windmill dataset: For ERM baselines, we use an MLP with two layers of size 40 and 1, with a ReLU activation after each layer (except the last) to extract the features. However, we observed that enforcing independence using 1-dimensional features was difficult. Therefore, we used 2-dimensional features for RepLIn, which were then normalized to lie on a sphere.

CelebA dataset: We first extract features from the raw image using a ResNet-50 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009). Although these features are not optimal for face attribute prediction, they are useful for face verification (Sharif Razavian et al., 2014). Additionally, it makes the binary attribute prediction task more challenging. We extract attribute-specific features from this input using a linear layer that maps it to a 500-dimensional space.

B Theoretical Motivation for RepLIn

In Sec. 3.4, we theoretically motivated RepLIn. This section explains the motivation with detailed proof.

Sketch of proof: First, we estimate the statistical risk in predicting the latent variables from interventional data from representations learned by arbitrary linear feature extractors and classifiers. In this statistical risk, we will identify a term that is the source of performance drop during interventions. We will then show that the optimal ERM models will suffer from this performance drop when trained on a dataset comprising observational and interventional data. Finally, we show that minimizing linear dependence between interventional features can lead to robust linear feature extractors.

Entity	Notation	Examples
Scalar	Regular lowercase characters	a, γ
Random variable	Regular serif uppercase characters	A
Random vector	Bold serif uppercase characters	$\boldsymbol{A}$
Distribution of a random variable A	P with subscript	P_A

Table 6: Mathematical notation used in the proof.

Setup: We follow the same mathematical notation as the main paper, shown in Tab. 6. The input data $\boldsymbol{X}$ is generated as a function of two latent variables of interest, A and B. There are noise variables collectively denoted by $\boldsymbol{U}$ that participate in the data generation but are not of learning interest. Our task is to predict A and B from $\boldsymbol{X}$. A and B are causally related during observation. For ease of exposition, we will consider a simple linear relation $B \coloneqq w_{AB}A$. This causal relation breaks when we intervene on B. The intervened variable is denoted with an added apostrophe (i.e., B'). The data generation process can be written in the form of a structural causal model as follows:

$$\begin{array}{cc} A \sim P_A & X_A \coloneqq w_A A + U_A \\ B' \sim P_{B'} & X_B \coloneqq w_B B + U_B \\ B \coloneqq w_{AB} A \text{ (during observations)} & \\ B \coloneqq B' \text{ (during interventions)} & \boldsymbol{X} = \begin{bmatrix} X_A \\ X_B \end{bmatrix} \\ U_A, U_B \sim P_{\mathsf{U}} & \end{array}$$

Training: The distribution from which training data is sampled is denoted by P_{train}. The training data consists of both observational and interventional samples, which themselves come from distributions P_{obs} and P_{int}. We are interested in the scenario where $(1-\beta)$ proportion of the training data is observational, while the remaining β proportion is interventional, where $0 < \beta < 1$. The training distribution can be represented as a mixture of observational and interventional distributions as follows:

$$P_{\text{train}}(\boldsymbol{X}, A, B) = (1-\beta)P_{\text{obs}}(\boldsymbol{X}, A, B) + \beta P_{\text{int}}(\boldsymbol{X}, A, B)$$

Typically, we assume $\beta \ll 1$. We will also assume that A, B, $\boldsymbol{U}$, and $\boldsymbol{X}$ have zero mean, so that we may use linear models without bias terms to extract representations corresponding to the variables of interest and train linear classifiers on these representations. The corresponding classifiers are parameterized by $\boldsymbol{c}^{(A)}$ and $\boldsymbol{c}^{(B)}$. The predictions are made by the classifiers from the learned representations as $\hat{A} = \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}$ and $\hat{B} = \boldsymbol{c}^{(B)\top}\boldsymbol{\Theta}^{(B)\top}\boldsymbol{X}$. The models are trained by minimizing the mean squared error on the training data, $\mathcal{L}_{\text{MSE}} = \mathbb{E}_{P_{\text{train}}}\left[\left(\left\|A - \hat{A}\right\|_2^2 + \left\|B - \hat{B}\right\|_2^2\right)\right]$.

B.1 Statistical Risk in Predicting Interventional Latent Samples

The model predicts $\hat{A}$ and $\hat{B}$ from $\boldsymbol{X}$ during inference. The statistical squared error in predicting A from interventional samples can be written as,

$$E_A = \mathbb{E}_{P_{\text{int}}}\left[\left(A - \hat{A}\right)^2\right] = \mathbb{E}_{P_{\text{int}}}\left[\left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2\right] \tag{7}$$

The expectation is taken over the interventional distribution over $\boldsymbol{X}, A, B, \boldsymbol{U}$ denoted by P_{int}. $\boldsymbol{\Theta}^{(A)}$ can be written in terms of constituent parameter vectors as $\boldsymbol{\Theta}^{(A)} = \begin{bmatrix} \boldsymbol{\theta}_A^{(A)\top} \\ \boldsymbol{\theta}_B^{(A)\top} \end{bmatrix}$. The predicted latent $\hat{A}$ can hence be written in terms of these vectors as,

$$\begin{aligned}
\hat{A} &= \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X} = \boldsymbol{c}^{(A)\top}\left(X_A\boldsymbol{\theta}_A^{(A)} + X_{B'}\boldsymbol{\theta}_B^{(A)} + \boldsymbol{\Theta}^{(A)\top}\boldsymbol{U}\right) \\
&= w_A A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} + w_B B'\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)} + \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U} \\
\therefore \left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2 &= \left(\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)A + w_B B'\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)} + \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U}\right)^2 \\
&= \left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 A^2 + \left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 B'^2 + \tilde{U}^2 \\
&\quad + 2\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)\left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)AB' \\
&\quad + 2\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)\tilde{U}A + 2\left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)\tilde{U}B'
\end{aligned} \tag{8}$$

$$\begin{aligned}
\therefore E_A = \mathbb{E}_{P_{\text{int}}}&\left[\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 A^2 + \left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 B'^2 + \tilde{U}^2\right] \\
&+ 2\mathbb{E}_{P_{\text{int}}}\left[\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)\left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)AB'\right] \\
&+ 2\mathbb{E}_{P_{\text{int}}}\left[\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)\tilde{U}A + 2\left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)\tilde{U}B'\right]
\end{aligned}$$

where $\tilde{U} = \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U} = \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}U_A + \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}U_B$. $\boldsymbol{U}$ denotes exogenous variables that are independent of A and B. Due to interventions, we also have $A \perp\!\!\!\perp B$. The expectation of AB' will be zero since

they are independent and have zero means marginally. Similarly, the expectation of the products of $\tilde{U}$ with A and B will be zero. Therefore,

$$E_A = \underbrace{\left(1 - w_A \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_A^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_{\boldsymbol{U}_A}^2}_{E_A^{(1)}} + \underbrace{\left(w_B \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{B'}^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{\boldsymbol{U}_B}^2}_{E_A^{(2)}} \tag{9}$$

where $\rho_A^2 = \mathbb{E}_{P_{\text{int}}}\left[A^2\right]$, $\rho_{B'}^2 = \mathbb{E}_{P_{\text{int}}}\left[B'^2\right]$, $\rho_{\boldsymbol{U}_A}^2 = \mathbb{E}_{P_{\text{int}}}\left[U_A^2\right]$, and $\rho_{\boldsymbol{U}_B}^2 = \mathbb{E}_{P_{\text{int}}}\left[U_B^2\right]$.

Statistical risk for a robust model: We are interested in robustness against interventional distribution shifts. The predictions of A by a robust model are unaffected by interventions on its child variable B. If $\hat{A}$ must not depend on B', then the corresponding representation $\boldsymbol{F}_A$ must not depend on it either, i.e. $\boldsymbol{\theta}_B^{(A)}$ must be a zero vector. Eq. (9) has two terms: $E_A^{(1)}$ and $E_A^{(2)}$. Therefore, a robust model will have $E_A^{(2)} = 0$ since $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}$. Therefore, showing that an optimal ERM model has a non-zero $\boldsymbol{\theta}_B^{(A)}$ is sufficient to show that the model is not robust.

B.2 Optimal ERM model

The optimal ERM model can be obtained by minimizing the expected risk in predicting the latent attributes. Since parameters are not shared between the prediction of $\boldsymbol{a}$ and $\boldsymbol{b}$, we can consider their optimization separately. We will consider the optimization of the parameters for predicting $\boldsymbol{a}$ since we are interested in the performance drop in predicting A from interventional data.

$$\boldsymbol{\Theta}^{(A)*}, \boldsymbol{c}^{(A)*} = \underset{\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}}{\text{argmin}}\ \mathbb{E}_{P_{\text{train}}}\left[\left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2\right]$$

where P_{train} is the joint distribution of $(\boldsymbol{X}, A, B)$ during training. As mentioned earlier, P_{train} is a mixture of observational distribution P_{obs} and interventional distribution P_{int}, with $(1-\beta)$ and β acting as the mixture weights. Therefore, the training objective can be rewritten as,

$$\begin{aligned}
\boldsymbol{\Theta}^{(A)*}, \boldsymbol{c}^{(A)*} &= \underset{\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}}{\text{argmin}}\ J(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}) \\
\text{where, } J(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}) &= \left((1-\beta)\mathbb{E}_{P_{\text{obs}}}\left[\left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2\right] + \beta\mathbb{E}_{P_{\text{int}}}\left[\left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2\right]\right)
\end{aligned} \tag{10}$$

Expanding the error term on observational data, we have,

$$\begin{aligned}
\boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X} &= \boldsymbol{c}^{(A)\top}\left(X_A\boldsymbol{\theta}_A^{(A)} + X_B\boldsymbol{\theta}_B^{(A)} + \boldsymbol{\Theta}^{(A)\top}\boldsymbol{U}\right) \\
&= w_A A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} + w_B B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)} + \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U} \\
&= w_A A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} + w_B w_{AB} A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)} + \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U} \\
\therefore \left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2 &= \left(A - w_A A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB} A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)} - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U}\right)^2 \\
&= \left(\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB}\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U}\right)^2 \\
&= \left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB}\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 A^2 + \tilde{U}^2 \\
&\quad - 2\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB}\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)A\tilde{U}
\end{aligned}$$

where $\tilde{U} = \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{U} = U_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} + U_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}$ from App. B.1. Since the exogenous variable $\boldsymbol{U}$ is independent of A and B, the expectation of their products over the observational distribution becomes zero.

Therefore,

$$\mathbb{E}_{P_{\text{obs}}}\left[\left(A - \boldsymbol{c}^{(A)\top}\boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}\right)^2\right] = \left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB}\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \mathbb{E}_{P_{\text{obs}}}\left[A^2\right] + \mathbb{E}_{P_{\text{obs}}}\left[\tilde{U}^2\right]$$
$$= \left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB}\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_A^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_{\boldsymbol{U}_A}^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{\boldsymbol{U}_B}^2 \tag{11}$$

Note that, $\rho_A^2 = \mathbb{E}_{P_{\text{obs}}}\left[A^2\right]$, $\rho_{\boldsymbol{U}_A}^2 = \mathbb{E}_{P_{\text{obs}}}\left[U_A^2\right]$, and $\rho_{\boldsymbol{U}_B}^2 = \mathbb{E}_{P_{\text{obs}}}\left[U_B^2\right]$ similar to App. B.1 since these values are unaffected by interventions. The expansion of the error term on interventional data was derived in Eq. (9). Thus, the overall training objective Eq. (10) can be written as,

$$J(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}) = (1-\beta)\left(\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)} - w_B w_{AB}\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_A^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_{\boldsymbol{U}_A}^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{\boldsymbol{U}_B}^2\right)$$
$$+ \beta\left(\left(1 - w_A\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_A^2 + \left(w_B\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{B'}^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}\right)^2 \rho_{\boldsymbol{U}_A}^2 + \left(\boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}\right)^2 \rho_{\boldsymbol{U}_B}^2\right)$$

We set $\psi_1 = \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_A^{(A)}$ and $\psi_2 = \boldsymbol{c}^{(A)\top}\boldsymbol{\theta}_B^{(A)}$. Since ERM jointly optimizes the feature extractors and the classifiers, no unique solution minimizes the prediction loss. For example, scaling the feature extractor parameters by an arbitrary constant scalar γ and the classifier parameters by $1/\gamma$ will give the same error. Therefore, we can optimize $J(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)})$ over ψ_1 and ψ_2, similar to (Arjovsky et al., 2019).

$$J(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}) = (1-\beta)\left((1 - w_A\psi_1 - w_B w_{AB}\psi_2)^2 \rho_A^2 + \psi_1^2\rho_{\boldsymbol{U}_A}^2 + \psi_2^2\rho_{\boldsymbol{U}_B}^2\right)$$
$$+ \beta\left((1 - w_A\psi_1)^2 \rho_A^2 + w_B^2\psi_2^2\rho_{B'}^2 + \psi_1^2\rho_{\boldsymbol{U}_A}^2 + \psi_2^2\rho_{\boldsymbol{U}_B}^2\right) \tag{12}$$

The optimal values of ψ_1 and ψ_2 are the stationary points of $J(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)})$ (denoted by J for brevity). Thus the optimal parameter values can be solved for by taking the first-order derivatives of J w.r.t. ψ_1 and ψ_2 and setting them to zero.

$$\frac{\partial J}{\partial \psi_1} = 2(1-\beta)\left(-(1 - w_A\psi_1 - w_B w_{AB}\psi_2)\, w_A\rho_A^2 + \psi_1\rho_{\boldsymbol{U}_A}^2\right) + 2\beta\left(-(1 - w_A\psi_1)\, w_A\rho_A^2 + \psi_1\rho_{\boldsymbol{U}_A}^2\right)$$
$$\frac{\partial J}{\partial \psi_2} = 2(1-\beta)\left(-(1 - w_A\psi_1 - w_B w_{AB}\psi_2)\, w_B w_{AB}\rho_A^2 + \psi_2\rho_{\boldsymbol{U}_B}^2\right) + 2\beta\left(w_B^2\psi_2\rho_{B'}^2 + \psi_2\rho_{\boldsymbol{U}_B}^2\right)$$

Setting $\frac{\partial J}{\partial \psi_1} = \frac{\partial J}{\partial \psi_2} = 0$, we have,

$$\begin{aligned} \left(w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2\right)\psi_1 &+ (1-\beta)w_A w_B w_{AB}\rho_A^2\psi_2 &- w_A\rho_A^2 &= 0 \\ (1-\beta)w_A w_B w_{AB}\rho_A^2\psi_1 &+ \left(\beta w_B^2\rho_{B'}^2 + (1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \rho_{\boldsymbol{U}_B}^2\right)\psi_2 &- (1-\beta)w_B w_{AB}\rho_A^2 &= 0 \end{aligned}$$

The equations are of the form $u_1\psi_1 + v_1\psi_2 + w_1 = 0$ and $u_2\psi_1 + v_2\psi_2 + w_2 = 0$. We can solve for ψ_2 as $\psi_2 = \frac{w_2u_1 - w_1u_2}{v_1u_2 - v_2u_1}$. Since we are only interested in probing the robustness of ERM models, we will check if ψ_2 is zero instead of fully solving the system of linear equations. $E_{\mathsf{A}}^{(2)}$ in Eq. (9) is zero if $\psi_2 = 0$, i.e. if $w_2u_1 - w_1u_2 = 0$.

$$w_2u_1 - w_1u_2 = -(1-\beta)w_B w_{AB}\left(w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2\right)\rho_A^2 + 4(1-\beta)w_A^2 w_B w_{AB}\rho_A^4$$
$$= -(1-\beta)w_B w_{AB}\rho_A^2\rho_{\boldsymbol{U}_A}^2$$

Unless the training data is entirely composed of interventional data (i.e., $\beta = 1$), $w_2u_1 - w_1u_2$ is not zero. Thus, the optimal ERM model is not robust against interventional distribution shifts.

B.3 Minimizing Linear Dependence

In Sec. 3.3, we showed that dependence between interventional features correlated positively with the drop in accuracy on interventional data. We will now verify if minimizing dependence between interventional

features can minimize the drop in accuracy. For ease of exposition, we will minimize the linear dependence between interventional features instead of enforcing statistical independence. The interventional features are given by $\boldsymbol{F}_A = \boldsymbol{\Theta}^{(A)\top}\boldsymbol{X}$ and $\boldsymbol{F}'_B = \boldsymbol{\Theta}^{(B)\top}\boldsymbol{X}$.

$$\begin{aligned}
\boldsymbol{F}_A = \boldsymbol{\Theta}^{(A)\top}\boldsymbol{X} &= \begin{bmatrix}\boldsymbol{\theta}_A^{(A)} & \boldsymbol{\theta}_B^{(A)}\end{bmatrix}\begin{bmatrix}X_A\\ X_B\end{bmatrix} \\
&= X_A\boldsymbol{\theta}_A^{(A)} + X_B\boldsymbol{\theta}_B^{(A)} \\
\boldsymbol{F}'_B = \boldsymbol{\Theta}^{(B)\top}\boldsymbol{X} &= \begin{bmatrix}\boldsymbol{\theta}_A^{(B)} & \boldsymbol{\theta}_B^{(B)}\end{bmatrix}\begin{bmatrix}X_A\\ X_B\end{bmatrix} \\
&= X_A\boldsymbol{\theta}_A^{(B)} + X_B\boldsymbol{\theta}_B^{(B)}
\end{aligned}$$

To define linear independence between interventional features, we use the following definition of cross-covariance from (Gretton et al., 2005):

Definition 1. *The* cross-covariance operator *associated with the joint probability p_{XY} is a linear operator $C_{XY} : \mathcal{G} \to \mathcal{F}$ defined as*

$$C_{XY} = \mathbb{E}_{XY}\left[(\phi(X) - \mu_X) \otimes (\psi(Y) - \mu_Y)\right]$$

where $\mathcal{G}$ and $\mathcal{F}$ are reproducing kernel Hilbert spaces (RKHSs) defined by feature maps ϕ and ψ respectively, and $\otimes$ is the tensor product defined as follows

$$(f \otimes g)h \coloneqq f\langle g, h\rangle_{\mathcal{G}} \text{ for all } h \in \mathcal{G}$$

where $\langle\cdot,\cdot\rangle$ is the inner product defined over $\mathcal{G}$.

In our case, instead of RKHS, we have finite-dimensional feature space $\mathbb{R}^d$. Therefore, we have the cross-covariance matrix as follows,

$$C_{XY} = \mathbb{E}_{XY}\left[\phi(X) \otimes \psi(Y)\right] = \mathbb{E}_{XY}\left[\phi(X)\psi(Y)^\top\right]$$

given that the feature maps have zero mean. Following the definition of HSIC (Gretton et al., 2005), linear dependence in the finite-dimensional case between X and Y is defined as the Frobenius norm of the cross-covariance matrix. Therefore, we define the linear dependence loss between the interventional features as,

$$\mathcal{L}_{\text{dep}} = \text{Dep}\left(\boldsymbol{F}_A, \boldsymbol{F}'_B\right) = \left\|\mathbb{E}_{P_{\text{int}}}\left[\boldsymbol{F}_A\boldsymbol{F}_B'^\top\right]\right\|_F^2 \tag{13}$$

Leveraging the independence relations during interventions, we can expand Eq. (13) as,

$$\begin{aligned}
\mathbb{E}_{P_{\text{int}}}\left[\boldsymbol{F}_A\boldsymbol{F}_B'^\top\right] &= \mathbb{E}_{P_{\text{int}}}\left[\left(X_A\boldsymbol{\theta}_A^{(A)} + X_B\boldsymbol{\theta}_B^{(A)}\right)\left(X_A\boldsymbol{\theta}_A^{(B)} + X_B\boldsymbol{\theta}_B^{(B)}\right)^\top\right] \\
&= \mathbb{E}_{P_{\text{int}}}\left[X_A^2\boldsymbol{\theta}_A^{(A)}\boldsymbol{\theta}_A^{(B)\top} + X_AX_B\boldsymbol{\theta}_A^{(A)}\boldsymbol{\theta}_B^{(B)\top} + X_AX_B\boldsymbol{\theta}_B^{(A)}\boldsymbol{\theta}_A^{(B)\top} + X_B^2\boldsymbol{\theta}_B^{(A)}\boldsymbol{\theta}_B^{(B)\top}\right] \\
&= (w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2)\boldsymbol{\theta}_A^{(A)}\boldsymbol{\theta}_A^{(B)\top} + (w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2)\boldsymbol{\theta}_B^{(A)}\boldsymbol{\theta}_B^{(B)\top} \\
\therefore \mathcal{L}_{\text{dep}} &= \left\|(w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2)\boldsymbol{\theta}_A^{(A)}\boldsymbol{\theta}_A^{(B)\top} + (w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2)\boldsymbol{\theta}_B^{(A)}\boldsymbol{\theta}_B^{(B)\top}\right\|_F^2
\end{aligned}$$

In the last step, all cross-covariance terms are zero due to the independence of the corresponding random variables in the causal graph. The dependence loss is the Frobenius norm of a sum of rank-one matrices $\boldsymbol{\theta}_A^{(A)}\boldsymbol{\theta}_A^{(B)\top}$ and $\boldsymbol{\theta}_B^{(A)}\boldsymbol{\theta}_B^{(B)\top}$. Consider the following general form: $\boldsymbol{Z} = \boldsymbol{a}\boldsymbol{b}^\top + \boldsymbol{c}\boldsymbol{d}^\top$. Then $Z_{ij} = a_ib_j + c_id_j$.

$$\|\boldsymbol{Z}\|_F^2 = \sum_{ij}\left(a_ib_j + c_id_j\right)^2$$

$\|\boldsymbol{Z}\|_F^2$ is a sum of squares and thus is zero iff $a_ib_j + c_id_j = 0$, $\forall i, j$. Therefore, $\mathcal{L}_{\text{dep}}$ is minimized when $\theta_{Ai}^{(A)}\theta_{Aj}^{(B)} + \theta_{Bi}^{(A)}\theta_{Bj}^{(B)} = 0$, $\forall i, j$. The potential solutions that minimize $\mathcal{L}_{\text{dep}}$ are (1) $\boldsymbol{\theta}_A^{(A)} = \boldsymbol{\theta}_B^{(A)} = \boldsymbol{\theta}_A^{(B)} =$

$\boldsymbol{\theta}_B^{(B)} = \mathbf{0}$, (2) $\boldsymbol{\theta}_A^{(A)} = \pm\gamma\boldsymbol{\theta}_B^{(A)}$ and $\gamma\boldsymbol{\theta}_A^{(B)} = \mp\boldsymbol{\theta}_B^{(B)}$ for some $\gamma \neq 0$, and (3) $\boldsymbol{\theta}_A^{(A)} = \mathbf{0}$ or $\boldsymbol{\theta}_A^{(B)} = \mathbf{0}$, and $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}$ or $\boldsymbol{\theta}_B^{(B)} = \mathbf{0}$. The former two solutions result in trivial features and will increase the classification error. The latter solution contains four possible solutions, out of which two solutions result in trivial features. Solutions resulting in trivial features are unlikely to occur during optimization due to a large classification error. Therefore, we need to consider only the remaining two solutions.

The possible solutions are: (1) $\boldsymbol{\theta}_A^{(A)} = \mathbf{0}, \boldsymbol{\theta}_B^{(B)} = \mathbf{0}$, and (2) $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}, \boldsymbol{\theta}_A^{(B)} = \mathbf{0}$. Intuitively, in the former solution, A and B will be predicted using X_B and X_A respectively, and the latter solution corresponds to a robust feature extractor that minimizes the reducible error in Eq. (9). We will compare the predictive error achieved by these solutions to compare their likelihood during training.

Recall the expression for training error in predicting A from Eq. (12).

$$\begin{aligned}
J_A(\boldsymbol{\Theta}^{(A)}, \boldsymbol{c}^{(A)}) &= (1-\beta)\left((1 - w_A\psi_{A1} - w_B w_{AB}\psi_{A2})^2 \rho_A^2 + \psi_{A1}^2\rho_{\boldsymbol{U}_A}^2 + \psi_{A2}^2\rho_{\boldsymbol{U}_B}^2\right) \\
&\quad + \beta\left((1 - w_A\psi_{A1})^2 \rho_A^2 + w_B^2\psi_{A2}^2\rho_{B'}^2 + \psi_{A1}^2\rho_{\boldsymbol{U}_A}^2 + \psi_{A2}^2\rho_{\boldsymbol{U}_B}^2\right) \\
&= (1-\beta)\left((1 - w_A\psi_{A1} - w_B w_{AB}\psi_{A2})^2 \rho_A^2\right) \\
&\quad + \beta\left((1 - w_A\psi_{A1})^2 \rho_A^2 + w_B^2\psi_{A2}^2\rho_{B'}^2\right) + \psi_{A1}^2\rho_{\boldsymbol{U}_A}^2 + \psi_{A2}^2\rho_{\boldsymbol{U}_B}^2
\end{aligned}$$

We use ψ_{A1} and ψ_{A2} instead of ψ_1 and ψ_2 respectively to denote the parameters for predicting A. A similar expression can be written for the error in predicting B with ψ_{B1} and ψ_{B2} denoting the parameters for predicting B.

$$\begin{aligned}
J_B(\boldsymbol{\Theta}^{(B)}, \boldsymbol{c}^{(B)}) &= (1-\beta)\left((1 - w_A\psi_{B1} - w_B w_{AB}\psi_{B2})^2 \rho_A^2 + \psi_{B1}^2\rho_{\boldsymbol{U}_A}^2 + \psi_{B2}^2\rho_{\boldsymbol{U}_B}^2\right) \\
&\quad + \beta\left(w_A^2\psi_{B1}^2\rho_A^2 + (1 - w_B\psi_{B2})^2\rho_{B'}^2 + \psi_{B1}^2\rho_{\boldsymbol{U}_A}^2 + \psi_{B2}^2\rho_{\boldsymbol{U}_B}^2\right) \\
&= (1-\beta)\left((1 - w_A\psi_{B1} - w_B w_{AB}\psi_{B2})^2 \rho_A^2\right) \\
&\quad + \beta\left(w_A^2\psi_{B1}^2\rho_A^2 + (1 - w_B\psi_{B2})^2\rho_{B'}^2\right) + \psi_{B1}^2\rho_{\boldsymbol{U}_A}^2 + \psi_{B2}^2\rho_{\boldsymbol{U}_B}^2
\end{aligned}$$

Case 1: When $\boldsymbol{\theta}_A^{(A)} = \mathbf{0}, \boldsymbol{\theta}_B^{(B)} = \mathbf{0}$: In this case, $\psi_{A1} = 0$ and $\psi_{B2} = 0$. Therefore, the predictive error during training for each latent variable can be written as,

$$\begin{aligned}
J_A &= (1-\beta)\left(w_B w_{AB}\psi_{A2} - 1\right)^2 \rho_A^2 + \beta\rho_A^2 + \beta w_B^2\psi_{A2}^2\rho_{B'}^2 + \psi_{A2}^2\rho_{\boldsymbol{U}_B}^2 \\
J_B &= (1-\beta)\left(w_A\psi_{B1} - w_{AB}\right)^2 \rho_A^2 + \beta w_A^2\psi_{B1}^2\rho_A^2 + \beta\rho_{B'}^2 + \psi_{B1}^2\rho_{\boldsymbol{U}_A}^2
\end{aligned}$$

The optimal values of ψ_{A2} and ψ_{B1} can be obtained by equating the gradients of R_A and R_B to zero.

$$\begin{aligned}
\frac{\partial J_A}{\partial \psi_{A2}} &= 2(1-\beta)w_B w_{AB}\left(w_B w_{AB}\psi_{A2} - 1\right)\rho_A^2 + 2\beta w_B^2\psi_{A2}\rho_{B'}^2 + 2\psi_{A2}\rho_{\boldsymbol{U}_B}^2 = 0 \\
\therefore \psi_{A2}^* &= \frac{(1-\beta)w_B w_{AB}\rho_A^2}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} \\
J_A^* &= \frac{(1-\beta)\rho_A^2\left(\beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2\right)}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} + \beta\rho_A^2
\end{aligned}$$

$$\begin{aligned}
\frac{\partial J_B}{\partial \psi_{B1}} &= 2(1-\beta)w_A\left(w_A\psi_{B1} - w_{AB}\right)\rho_A^2 + 2\beta w_A^2\psi_{B1}\rho_A^2 + 2\psi_{B1}\rho_{\boldsymbol{U}_A}^2 = 0 \\
\therefore \psi_{B1}^* &= \frac{(1-\beta)w_A w_{AB}\rho_A^2}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} \\
J_B^* &= \frac{(1-\beta)w_{AB}^2\rho_A^2(\beta w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2)}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} + \beta\rho_{B'}^2
\end{aligned}$$

The combined training error for this solution is,

$$\begin{aligned} J_1^* &= J_A^* + J_B^* \\ &= \frac{(1-\beta)\rho_A^2\left(\beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2\right)}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} + \beta\rho_A^2 \\ &\quad + \frac{(1-\beta)w_{AB}^2\rho_A^2(\beta w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2)}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} + \beta\rho_{B'}^2 \end{aligned} \tag{14}$$

Case 2: When $\boldsymbol{\theta}_B^{(A)} = \mathbf{0}, \boldsymbol{\theta}_A^{(B)} = \mathbf{0}$: Here, $\psi_{A2} = 0$ and $\psi_{B1} = 0$. The predictive error during training for each latent variable can be written as,

$$\begin{aligned} J_A &= \left(w_A\psi_{A1} - 1\right)^2\rho_A^2 + \psi_{A1}^2\rho_{\boldsymbol{U}_A}^2 \\ J_B &= \left((1-\beta)w_{AB}^2\rho_A^2 + \beta\rho_{B'}^2\right)\left(w_B\psi_{B2} - 1\right)^2 + \psi_{B2}^2\rho_{\boldsymbol{U}_B}^2 \end{aligned}$$

We follow the former procedure to estimate the optimal values of ψ_{A1} and ψ_{B2}.

$$\begin{aligned} \frac{\partial J_A}{\partial \psi_{A1}} &= 2w_A\left(w_A\psi_{A1} - 1\right)\rho_A^2 + 2\psi_{A1}\rho_{\boldsymbol{U}_A}^2 = 0 \\ \therefore \psi_{A1}^* &= \frac{w_A\rho_A^2}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} \\ J_A^* &= \frac{\rho_A^2\rho_{\boldsymbol{U}_A}^2}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} \end{aligned}$$

$$\begin{aligned} \frac{\partial J_B}{\partial \psi_{B2}} &= 2w_B\left((1-\beta)w_{AB}^2\rho_A^2 + \beta\rho_{B'}^2\right)\left(w_B\psi_{B2} - 1\right) + 2\psi_{B2}\rho_{\boldsymbol{U}_B}^2 \\ \therefore \psi_{B2}^* &= \frac{(1-\beta)w_B w_{AB}^2\rho_A^2 + \beta w_B\rho_{B'}^2}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} \\ J_B^* &= \frac{\left((1-\beta)w_{AB}^2\rho_A^2 + \beta\rho_{B'}^2\right)\rho_{\boldsymbol{U}_B}^2}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} \end{aligned}$$

The combined training error for this solution is,

$$\begin{aligned} J_2^* &= J_A^* + J_B^* \\ &= \frac{\rho_A^2\rho_{\boldsymbol{U}_A}^2}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} + \frac{\left((1-\beta)w_{AB}^2\rho_A^2 + \beta\rho_{B'}^2\right)\rho_{\boldsymbol{U}_B}^2}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} \end{aligned} \tag{15}$$

Comparing J_1^* and J_2^*,

$$\begin{aligned} J_1^* - J_2^* = &\frac{(1-\beta)\beta w_B^2\rho_A^2\rho_{B'}^2 + (1-\beta)\rho_A^2\rho_{\boldsymbol{U}_B}^2 - (1-\beta)w_{AB}^2\rho_A^2\rho_{\boldsymbol{U}_B}^2 - \beta\rho_{B'}^2\rho_{\boldsymbol{U}_B}^2}{(1-\beta)w_B^2 w_{AB}^2\rho_A^2 + \beta w_B^2\rho_{B'}^2 + \rho_{\boldsymbol{U}_B}^2} \\ &+ \frac{(1-\beta)\beta w_A^2 w_{AB}^2\rho_A^4 + (1-\beta)w_{AB}^2\rho_A^2\rho_{\boldsymbol{U}_A}^2 - \rho_A^2\rho_{\boldsymbol{U}_A}^2}{w_A^2\rho_A^2 + \rho_{\boldsymbol{U}_A}^2} + \beta(\rho_A^2 + \rho_{B'}^2) \end{aligned}$$

Simplifying the above expression, we get the condition that $J_1^* - J_2^* > 0$ if β satisfies the following conditions: (1) $\beta \geq 1 - \frac{1}{|w_{AB}|}$, (2) $\beta \geq \min\left(\frac{\rho_A^2}{2\rho_{B'}^2 + \rho_A^2}, \frac{\rho_{\boldsymbol{U}_A}^2}{w_A^2 w_{AB}^2\rho_A^2}\right)$. The conditions imply that enforcing linear independence results in robust feature extractors when *enough* interventional data is available during training.

However, this is only a sufficient condition that strictly ensures $J_1^* - J_2^* > 0$. In practice, β could be much lower, especially when the total loss is of the form $\mathcal{L}_{\text{total}} = \lambda_{\text{MSE}}\mathcal{L}_{\text{MSE}} + \lambda_{\text{dep}}\mathcal{L}_{\text{dep}}$, where λ_{MSE} and λ_{dep}

are positive hyperparameters. We verify this empirically by randomly setting the parameters of the data generation process and plotting the predictive errors J_1^* and J_2^* for different values of β. We calculate J_1^* and J_2^* for 5000 runs (shown using thin curves) and plot the average error (shown using thick curves) in Fig. 13. We observe that the average value of J_1^* is always higher than that of J_2^* for all values of β. But, when $\beta \to 0$, their average values get closer to each other.

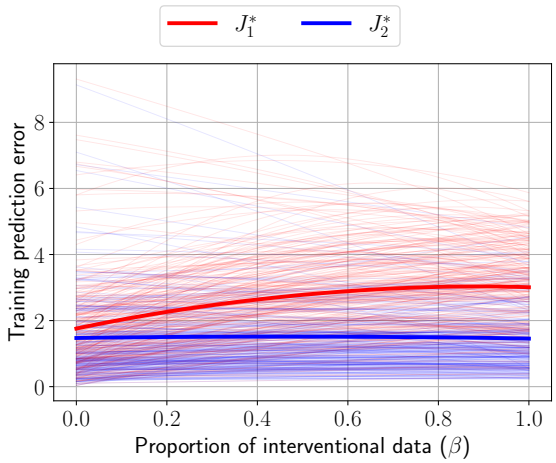


Figure 13: Comparing J_1^* (Eq. (14)) and J_2^* (Eq. (15)) as functions of β for 5000 runs with randomly sampled data generation parameters. We show individual runs using thin curves and the average error values using thick curves. We only show the errors from a few randomly sampled runs for visual clarity. We observe that the average value of J_1^* (shown using thick red curve) is always higher than that of J_2^* (shown using thick blue curve), indicating that enforcing linear independence between interventional features is more likely to obtain robust feature extractors than degenerate solutions.

C Review of identifiable causal representation learning

The primary objective of identifiable causal representation learning (ICRL) is to learn a representation such that it is possible to identify the latent factors (up to permutation and elementwise transformation) from the representation. These methods are commonly built upon autoencoder-based approaches and learn generative representations. The advantage of learning a causal representation is that the decoder then implicitly acts as the true underlying causal model, facilitating counterfactual evaluation and interpretable representations.

Locatello et al. (2019); Khemakhem et al. (2020) showed that disentangled representation learning was impossible without additional assumptions on both the model and the data. Some of the inductive biases that have been proposed since to learn disentangled representations include auxiliary labels (Hyvarinen & Morioka, 2016; Hyvarinen et al., 2019; Sorrenson et al., 2020; Khemakhem et al., 2020; Lu et al., 2021; Ahuja et al., 2022b; Kong et al., 2022), temporal data (Klindt et al., 2021; Yao et al., 2022; Song et al., 2023), and assumptions on the mixing function (Sorrenson et al., 2020; Yang et al., 2021; Lachapelle et al., 2022; Zheng et al., 2022; Moran et al., 2022).

Use of interventional data: Some works also use interventional data as weak supervision for identifiable representation learning (Lippe et al., 2022b; Brehmer et al., 2022; Ahuja et al., 2022a; 2023; Varıcı et al., 2023; von Kügelgen et al., 2023). Lippe et al. (2022b) learns identifiable representations from temporal sequences with possible interventions at any time step. Similar to our setting, they assume the knowledge of the intervention target. They also assume that the intervention on a latent variable at a time step does not affect other latent variables in the same time step. Lippe et al. (2023) relaxes the latter assumption as long as perfect interventions with known targets are available. Von Kügelgen et al. (2021); Zimmermann et al. (2021) showed that self-supervised learning with data augmentations allowed for identifiable representation learning. Brehmer et al. (2022) use pairs of data samples before and after some unknown intervention to learn latent causal models. Ahuja et al. (2022a) learns identifiable representations from sparse perturbations,

with identifiability guarantees depending on the sparsity of these perturbations. Sparse perturbations can be treated as a parent class of interventions where the latent is intervened through an external action such as in reinforcement learning. Ahuja et al. (2022b) use interventional data for causal learning for polynomial mixing functions, under some assumptions on the nature of support for non-intervened variables. Varıcı et al. (2024a) relaxes the polynomial assumption on the mixing function and proves identifiability when two uncoupled hard interventions per node are available along with observational data. Varıcı et al. (2023) learn identifiable representations from data observed under different interventional distributions with the help of the score function during interventions. von Kügelgen et al. (2023) uses interventional data to learn identifiable representations up to nonlinear scaling. In addition to the above uses of interventional data, a few works (Saengkyongam & Silva, 2020; Saengkyongam et al., 2024; Zhang et al., 2023) have also attempted to predict the effect of unseen joint interventions with the help of observational and atomic interventions under various assumptions on the underlying causal model.

Difference from our setting: The general objective in ICRL is to "learn both the true joint distribution over both observed and latent variables" (Khemakhem et al., 2020). In contrast, the objective of our work is to learn representations corresponding to latent variables that are robust against interventional distributional shifts by leveraging *known* interventional independence relations. We pursue this objective in the hope that, as large models (Radford et al., 2021; Brown et al., 2020; Touvron et al., 2023; Dehghani et al., 2023) become more ubiquitous, efficient methods to improve these models with minimal amounts of experimentally collected data will be of interest. Stated more formally, full identifiability of the underlying causal model is not in our interest, as robustness to interventional distribution shift can be achieved without full identifiability. For instance, consider the following setup: Let $A = [A_1, A_2]$ cause B during observation. Here, A_1 is a binary variable (also, the class we are interested in predicting) and A_2 is a continuous variable from a Gaussian mixture with 2 modes. The mode is decided by A_1, and therefore informs the class. The observed data is $X = [X_{A_1} X_{A_2} X_B]$, where X_{A_1} depends only on A_1, X_{A_2} only on A_2, and X_B only on B. Suppose the relations from the latent variables to the corresponding observed variables are such that it is possible to learn A_1 and B from X, but not A_2 (say, due to noise or information loss in the mixing function). The discriminative task here is to predict which class A belongs to. Here, RepLIn can learn robust representations that can fully predict the class of A (through A_1), but is not fully identifiable since it does not have information about A_2.

D Differences w.r.t DG/OOD Setting

Although the problem setting of RepLIn may seem superficially similar to those of domain generalization (DG) or out-of-distribution (OOD) tasks, there are differences between them owing to the assumptions in our setting that also allow us to get more benefits from RepLIn. Expressed in terms of the random variables A, B, and X in our problem setting from Sec. 3, the task in DG is to predict some variable of interest A from the observed data X such that the learned model can generalize to unseen domains (Wang et al., 2022a; Ding et al., 2022b). To ensure robust prediction, we have access to multiple sets of training data that vary from each other in terms of some variable B. DG has a general framework with the goal simply stated as learning a model that transfers well between domains. Tab. 7 shows the differences between the DG framework and ours. The first two rows show the differences in settings, while the last two rows show the differences in learned representations.

E Additional Results from Experiments

As mentioned in the main paper, our objective is to improve the robustness of representations against interventional distribution shifts. However, this robustness might come at the cost of observational accuracy since it removes spurious information that gives better performance on observational data. In this section, we report the results of the baselines and our methods on Windmill, CelebA, and CivilComments datasets.

Differences	DG/OOD	RepLIn
Relation from A to X between domains	May or may not change	Does not change
Is A independent of B in one or more domains?	May or may not be. It is also possible that A is always independent of B.	$A \rightarrow B$ in observational data and A independent of B in interventional data.
Can accommodate more than one B?	No. B is also not of interest.	Yes. Useful in cases where it is of interest to learn representations for B as well.
Is the representation learned for A free of information from B?	Not necessarily. Some DG methods are designed to remove information from B, while others are not (e.g, DARE (Rosenfeld et al., 2022))	Yes, the dependence loss ensures that the features for A are free of information from B in the interventional data.

Table 7: Differences between the problem settings of domain generalization and RepLIn.

Method	$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.05$	$\beta = 0.01$
ERM	93.85 ± 1.84	98.06 ± 1.20	99.70 ± 0.08	99.92 ± 0.02	99.98 ± 0.01
ERM-Resampled	94.53 ± 0.89	94.13 ± 1.19	94.84 ± 0.92	94.56 ± 0.71	94.53 ± 1.14
IRMv1	93.37 ± 0.85	93.59 ± 0.32	93.72 ± 0.73	92.52 ± 0.35	94.04 ± 0.63
Fish	95.54 ± 0.42	95.37 ± 0.36	95.42 ± 0.59	95.83 ± 0.51	96.28 ± 1.12
GroupDRO	82.02 ± 2.00	84.40 ± 2.72	85.35 ± 2.35	84.25 ± 0.91	92.28 ± 1.11
SAGM	94.77 ± 0.62	95.17 ± 0.71	94.13 ± 1.68	95.61 ± 0.69	94.04 ± 1.98
DiWA	94.64 ± 0.96	94.30 ± 0.36	94.57 ± 0.64	94.39 ± 0.99	94.24 ± 0.59
TEP	65.20 ± 14.22	66.94 ± 3.78	61.34 ± 19.35	63.02 ± 15.59	73.77 ± 9.01
RepLIn	95.16 ± 0.53	97.83 ± 0.40	99.24 ± 0.37	98.75 ± 0.43	99.10 ± 0.47
RepLIn-Resampled	95.57 ± 0.62	95.77 ± 0.68	95.59 ± 1.08	95.90 ± 0.35	95.51 ± 1.71

Table 8: Observational accuracy of various methods used in Sec. 5.1.

Method	$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.05$	$\beta = 0.01$
ERM	76.87 ± 1.08	69.86 ± 3.19	62.78 ± 1.77	59.52 ± 1.30	60.15 ± 3.12
ERM-Resampled	73.70 ± 3.19	71.19 ± 3.23	73.62 ± 1.54	71.03 ± 2.83	70.20 ± 3.73
IRMv1	78.24 ± 0.79	74.83 ± 1.74	78.61 ± 2.24	76.28 ± 1.87	71.75 ± 2.03
Fish	77.23 ± 2.24	77.23 ± 1.32	78.24 ± 2.09	76.42 ± 1.95	73.92 ± 2.53
GroupDRO	80.10 ± 1.66	80.96 ± 1.33	80.35 ± 1.01	77.40 ± 1.16	71.86 ± 1.60
SAGM	76.43 ± 2.37	79.05 ± 2.23	76.96 ± 4.36	79.86 ± 1.81	72.81 ± 3.10
DiWA	76.61 ± 2.15	76.71 ± 0.59	76.09 ± 0.69	75.83 ± 1.83	73.39 ± 1.31
TEP	58.68 ± 4.72	60.42 ± 1.30	56.07 ± 3.35	58.52 ± 4.36	59.23 ± 1.13
RepLIn	87.94 ± 1.46	87.76 ± 2.30	83.23 ± 2.67	73.63 ± 2.43	67.52 ± 2.30
RepLIn-Resampled	88.46 ± 0.96	88.05 ± 1.04	87.91 ± 1.36	86.38 ± 0.85	78.41 ± 1.27

Table 9: Interventional accuracy of various methods used in Sec. 5.1.

F More GradCAM Visualization

We show more GradCAM visualizations to illustrate the differences between the representations between ERM and RepLIn. In Sec. 6.1, we compared the GradCAM visualizations for those samples that were correctly classified by RepLIn, but incorrectly by ERM. Here, we visualize the GradCAM from samples that were correctly classified by ERM, but incorrectly by RepLIn. In Fig. 14, the top row shows the GradCAM

Method	$\beta = 0.5$	$\beta = 0.4$	$\beta = 0.3$	$\beta = 0.2$	$\beta = 0.1$	$\beta = 0.05$
ERM-Resampled	91.38 ± 0.09	91.52 ± 0.06	91.39 ± 0.07	90.89 ± 0.10	90.57 ± 0.09	91.82 ± 0.14
RepLIn-Resampled	86.02 ± 0.18	86.35 ± 0.24	86.58 ± 0.11	86.94 ± 0.36	87.67 ± 0.21	89.83 ± 0.11

Table 10: Observational accuracy of various methods used in Sec. 5.2.

Method	$\beta = 0.5$	$\beta = 0.4$	$\beta = 0.3$	$\beta = 0.2$	$\beta = 0.1$	$\beta = 0.05$
ERM-Resampled	81.09 ± 0.17	80.56 ± 0.23	80.06 ± 0.17	79.08 ± 0.16	76.63 ± 0.24	73.42 ± 0.27
RepLIn-Resampled	81.97 ± 0.14	81.94 ± 0.17	81.84 ± 0.18	80.65 ± 0.22	78.56 ± 0.20	75.77 ± 0.05

Table 11: Interventional accuracy of various methods used in Sec. 5.2.

Method	$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.05$	$\beta = 0.01$
ERM-Resampled	81.26 ± 0.12	81.77 ± 0.14	79.78 ± 0.08	79.97 ± 0.12	79.13 ± 0.09
RepLIn-Resampled	79.27 ± 0.09	80.16 ± 0.12	77.65 ± 0.06	77.84 ± 0.12	78.51 ± 0.16

Table 12: Observational accuracy of various methods used in Sec. 5.3.

Method	$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.05$	$\beta = 0.01$
ERM-Resampled	74.51 ± 0.07	75.29 ± 0.22	72.03 ± 0.18	71.78 ± 0.12	69.80 ± 0.45
RepLIn-Resampled	75.30 ± 0.37	75.81 ± 0.31	72.00 ± 0.23	71.70 ± 0.14	69.99 ± 0.80

Table 13: Interventional accuracy of various methods used in Sec. 5.3.

visualizations of ERM, while the bottom row shows the visualizations for RepLIn. These samples are **chosen randomly**. Although these samples were incorrectly classified by RepLIn, the attention maps of RepLIn for most of the shown samples focus on the mouth region. On the other hand, attention maps of ERM do not focus on the mouth region as frequently.

G Visualization of Feature Distribution Learned on Windmill dataset

In this section, we compare the feature distributions learned by RepLIn on Windmill dataset against all the baselines from Sec. 5.1. The feature distributions are shown in Fig. 15.

H Balancing $\mathcal{L}_{\text{dep}}$ and $\mathcal{L}_{\text{self}}$ during training

The goal of RepLIn is to learn robust discriminative representations corresponding to variables of predictive interest such that each representation contains only the information from the latent variable it models, especially when these latent variables are causally related. This goal must be evaluated on two fronts – the absolute utility of the representations for downstream tasks and performance equity between observational and interventional distributions. We quantified these evaluations in our experiments through the performance on interventional data and the relative accuracy drop between observational and interventional distributions. Our proposed loss functions also reflected these objectives: (1) self-dependence loss ($\mathcal{L}_{\text{self}}$) maximizes the information that a representation learns about its corresponding latent variable, and (2) dependence loss ($\mathcal{L}_{\text{dep}}$) minimizes the information shared by the representations of causally related latent variables during interventions, to obtain distributionally robust representations.

However, $\mathcal{L}_{\text{self}}$ and $\mathcal{L}_{\text{dep}}$ have somewhat conflicting objectives. Minimizing $\mathcal{L}_{\text{self}}$ maximizes the statistical information shared between latent variables and their corresponding representations. It does not discriminate the nature of this information and, thus, could include information about the child variables in the

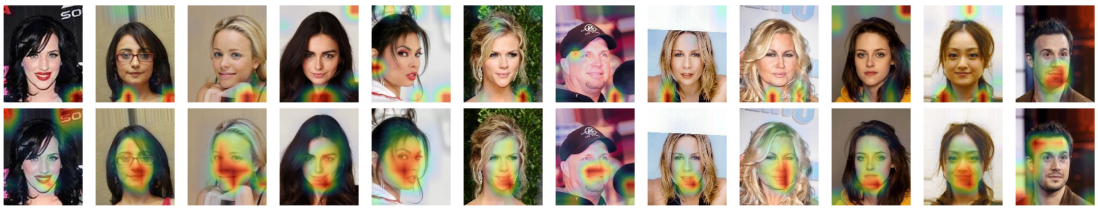


Figure 14

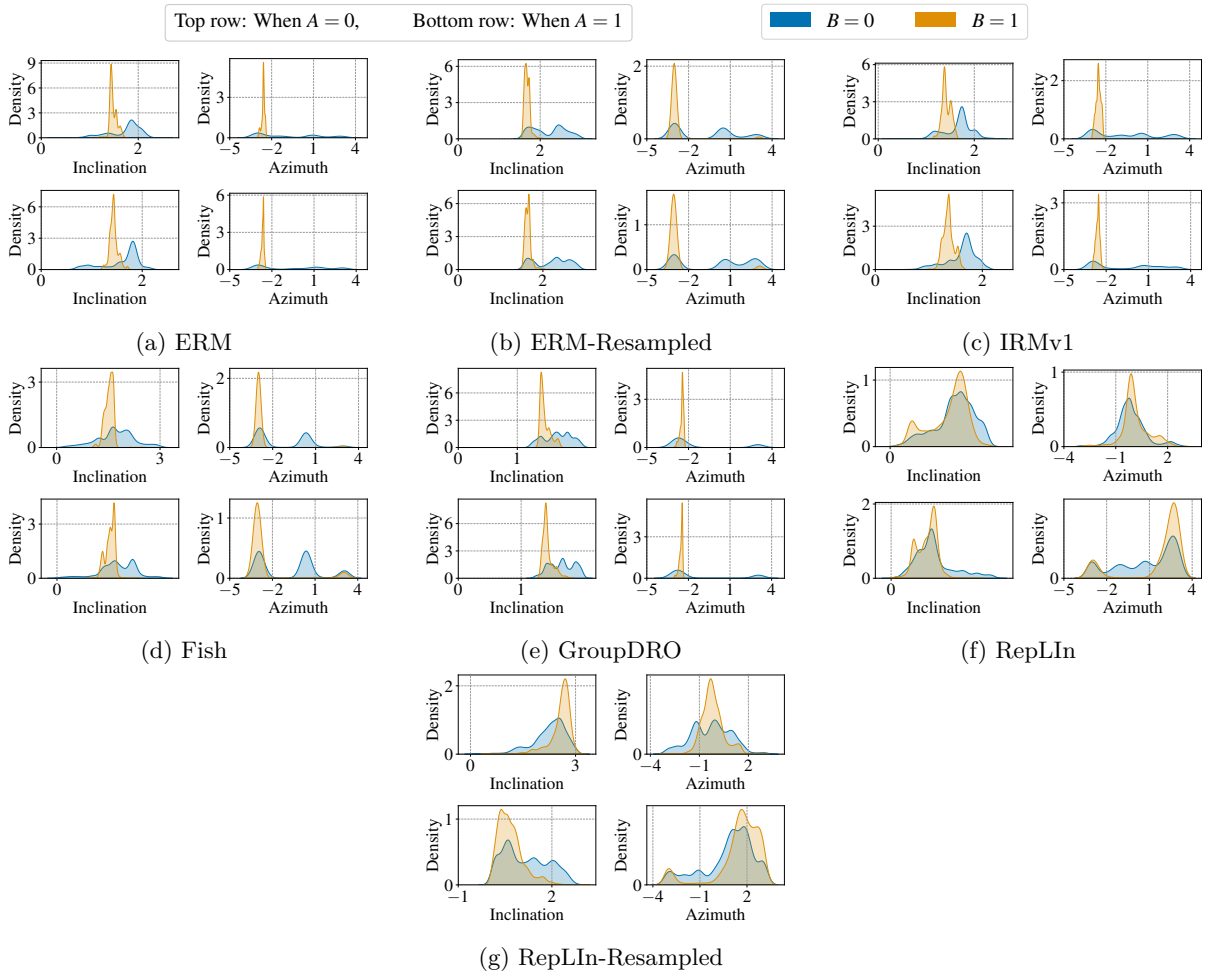


Figure 15: Visualization of interventional features learned by various methods on WINDMILL dataset.

representation when minimized on observational data. Minimizing $\mathcal{L}_{\text{dep}}$ ensures that the interventional representations corresponding to independent variables do not share any information, regardless of whether these representations contain any discriminative information useful for predicting their corresponding latent variable. Thus, fundamentally, $\mathcal{L}_{\text{self}}$ enriches the information in the representations, while $\mathcal{L}_{\text{dep}}$ removes the information from the representations. If these loss functions are not balanced during training using their respective hyperparameters λ_{self} and λ_{dep}, the learned representations may not be robust and discriminative.

We experimentally demonstrate the above statements with the help of a synthetic dataset with linear relations between variables, similar to the one used for theoretical analysis in Sec. 3.4.

Experiment setup: Our dataset consists of the high-dimensional observed signal $X \in \mathbb{R}^{100}$ from which we must predict two latent variables of interest, $A, B \in \mathbb{R}^{10}$. During observation, $A \to B$ in the underlying causal graph with the following linear causal relation between them.

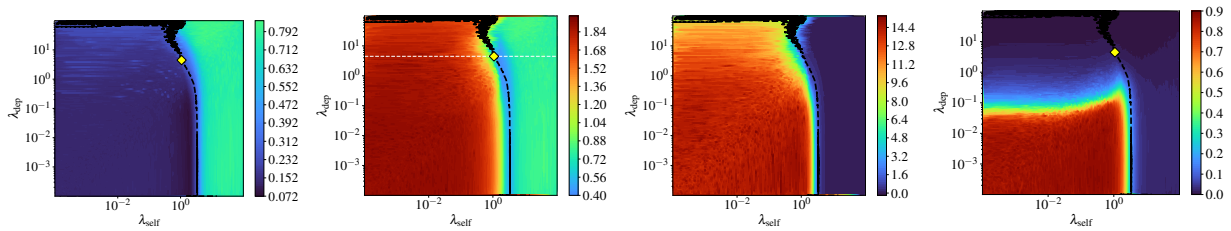
$$
\begin{aligned}
A &\sim \mathcal{N}(0, I_{10}) && (I_p \text{ is } p \times p \text{ identity matrix}) \\
\epsilon &\sim \mathcal{N}(0, I_{10}) && (\text{Noise in observational relation}) \\
B &\coloneqq \sqrt{0.9}A + \sqrt{0.1}\epsilon
\end{aligned}
$$

To collect interventional data, we intervene on B and set it to independently sampled $\tilde{B} \sim \mathcal{N}(0, I_{10})$. During intervention, $A \perp\!\!\!\perp \tilde{B}$. A and B, along with exogenous random variable $U \sim \mathcal{N}(0, I_{80})$, create the observed signal X from which we are tasked with learning representations corresponding to A and B. Formally,

$$
\begin{aligned}
n &\sim \mathcal{N}(0, 0.25 I_{10}) && (\text{Noise in the mixing function}) \\
\hat{A} &= A + n \\
\hat{X} &= \begin{bmatrix} \hat{A} & B & U \end{bmatrix} \\
X &= W\tilde{X} + Z,
\end{aligned}
\tag{16}
$$

where $W \in \mathbb{R}^{100\times100}$ and $Z \in \mathbb{R}^{100}$ are the linear coefficients of the mixing function whose entries are independently sampled from $\mathcal{N}(0, 1)$. During its sampling, we verify that W is a full-rank matrix to ensure that a linear model can predict A and B from X. Note that the noise n added to A has a higher variance than the noise ϵ in the observational causal relation. This would prompt the model to learn *shortcut* (Geirhos et al., 2020) and rely on the information from B to predict A. Since we know the variance of the noise added to A, we can also compute the statistical error of a robust linear model.

Our model consists of a linear layer each to learn the representations corresponding to A and B, and a linear layer each to make the final predictions $\hat{A}$ and $\hat{B}$ from their respective representations. The model does not have any non-linear activation function. The models are trained by minimizing the mean squared error between their predictions and the ground truth, in addition to $\mathcal{L}_{\text{dep}}$ and $\mathcal{L}_{\text{self}}$ weighted by their corresponding hyperparameters λ_{dep} and λ_{self}, respectively. Each batch comprises the entire training dataset. For each run, we first generate a different random seed s_{run} that affects the sampled values for W, Z, A, and B. Random values for s_{run} are generated using a meta random seed s_{meta} obtained from the system timestamp during the experiment run. We also use s_{meta} to randomly sample λ_{dep} and λ_{self} from their uniform distributions in their log space. In total, 27,748 random hyperparameter settings were sampled.



(a) Observational error in predicting A

(b) Interventional error in predicting A

(c) Relative increase in error

(d) Dependence between representations

Figure 16: Results of RepLIn models trained with different values for the hyperparameters λ_{dep} and λ_{self}. The heatmaps show the variations of interventional accuracy (left) and relative drop in accuracy between observational and interventional distributions (right) with the hyperparameters.

The results of our experiments are shown in Fig. 16. In the results, we plot and analyze the prediction accuracy on A since we intervened on B. To obtain continuous-valued plots, we interpolate between the sampled pairs of λ_{dep} and λ_{self} through triangulation. We make the following observations from the results:

(1) Small values for λ_{dep} and λ_{self}: RepLIn behaves similarly to vanilla ERM method as $\lambda_{\text{dep}}, \lambda_{\text{self}} \to 0$. In Fig. 16, this setting corresponds to the lower-left quadrant of each plot. Due to the designed difficulty

in predicting A from X, the model uses information from B to predict A, resulting in a low error in observational data (Fig. 16a) and a high error in interventional data (Fig. 16b). Statistical dependence between representations during interventions measured using NHSIC is also high (Fig. 16d), as expected.

(2) Increasing λ_{dep} alone: When λ_{dep} is increased without changing λ_{self}, dependence between representations of interventional data decreases, as expected. However, increasing λ_{dep} sometimes provides only limited reductions in interventional error, as seen in Fig. 16c. For instance, increasing λ_{dep} from 10^{-3} to 1, while keeping a constant $\lambda_{\text{self}} = 10^{-3}$ slightly decreased the error on interventional data from 1.89 to 1.76, while nominally increasing the error on observational data from 0.127 to 0.129. This shows that while minimizing interventional dependence helps learn robust representations against interventions, the benefits in performance may be marginal.

(3) Increasing λ_{self} alone: Interestingly, increasing only λ_{self} leads to a drop in interventional dependence and reduces the error disparity between observational and interventional data (Fig. 16c), even when λ_{dep} is nearly zero. However, this decrease in performance disparity comes at the cost of higher observational error (left to right in Fig. 16a).

(4) Lowest interventional error: In Fig. 16b, we can observe a valley of relatively lower interventional error. The hyperparameter combination corresponding to the lowest interventional error occurs within this valley, marked with a yellow diamond. The same position is marked on other plots for ease of viewing. The lowest interventional error obtained experimentally was 0.4, considerably higher than the theoretical interventional error of 0.25 that a robust model would have attained. This indicates that the best hyperparameter combination did not result in a fully robust model. However, this is not surprising since our theoretical results in Sec. 3.4 suggested that a linear model cannot learn a fully robust model if the training dataset contains any observational data. Additionally, note that this hyperparameter combination did not result in the lowest performance disparity between the distributions and, instead, it appeared near a *phase change* in the loss values. To observe this phase change more clearly, we plot the loss values along the white dashed line in Fig. 16b, where we vary λ_{self} and fix λ_{dep} to the value it takes in the best hyperparameter combination (yellow diamond).

In Fig. 17, we observe that as λ_{self} increases, interventional error drops rapidly, achieving its minimum at λ_{dep} corresponding to the yellow diamond (denoted by the dashed black line in Fig. 17), and then increases steadily to eventually saturate. Similarly, observational error gradually increases with increasing λ_{self} initially and then displays a more rapid increase, eventually matching the interventional error at higher values of λ_{self}. Throughout these changes, the statistical dependence between the representations of interventional data remains nearly zero.

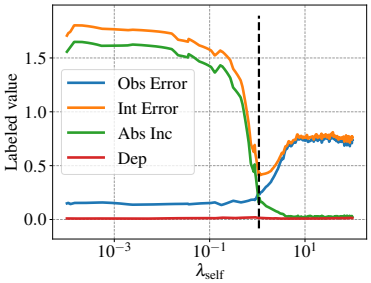


Figure 17: Change in observational and interventional error values for a fixed λ_{dep} corresponding to the yellow diamond in Fig. 16b and varying λ_{self}.

Our results indicate that, while both $\mathcal{L}_{\text{dep}}$ and $\mathcal{L}_{\text{self}}$ are needed to learn discriminative representations that are robust to interventional distribution shifts without losing their utility in downstream applications, hyperparameter tuning is still necessary to balance the effects of these loss functions.

H.1 Why do the hyperparameters change between experiments?

In our main experiments, we chose different hyperparameters for different experiments. In this section, we explore the factors that affect the choice of hyperparameters between experiments. In particular, we focus on λ_{dep}, as $\mathcal{L}_{\text{dep}}$ is the primary loss function responsible for enforcing statistical independence between the interventional representations. We start by noting that robust representations are obtained by (at least partially) inverting the data-generating function from the latent variable to the observed signal. Therefore, we hypothesize that as the complexity of this data-generating function increases, λ_{dep} and λ_{self} generally increase. Here, we use the term "complexity" to roughly mean the minimum degree of a polynomial required to model the data-generating function. Informally, the more complex the data-generating function, the more hesitant the model is to learn robust representations (Geirhos et al., 2020).

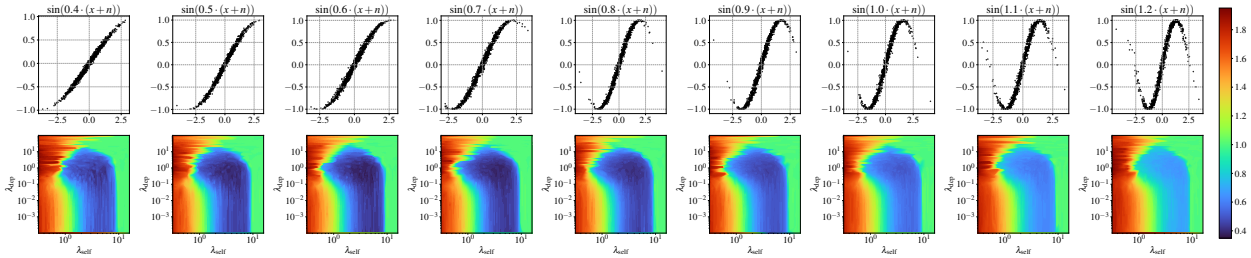


Figure 18: (*Top*) Sinusoidal transformation of a 1D Gaussian random variable x with added noise $n \sim \mathcal{N}(0, 0.01)$, and (*bottom*) variation of interventional error for various values of λ_{dep} and λ_{self}.

We now formally verify our hypothesis by adding a non-linear function in Eq. (16) of the simple dataset that we used to investigate the effect of λ_{dep} and λ_{self}. We modify Eq. (16) as follows:

$$\hat{X} = \left[\sin\left(s \cdot \hat{A}\right) \quad B \quad U\right], \tag{17}$$

where s controls the amount of non-linearity. A very low value for s will result in a nearly linear function, as sin function is approximately linear near the origin. As s increases, the non-linearity also increases. For higher values of s, multiple values of $\hat{A}$ will be mapped to the same value. For the remainder of this section, we will refer to s as the "non-linearity factor." See Fig. 18 (top) on how the value of s affects the sinusoidal transformation of a Gaussian random variable with added noise.

In addition to modifying the data generation process by using a non-linear relation from the latent variable to the observed signal, we also use non-linear models to learn the representations for each variable. Specifically, we use MLPs with 2 hidden layers and the ReLU activation function. We are interested in the variation in λ_{dep} that gives us the minimum value of interventional error as the non-linearity factor s changes. If our hypothesis is correct, then λ_{dep} must increase as the non-linearity factor s in Eq. (17) increases. Following the previous setup, we will sample several values for λ_{dep} and λ_{self}, and train models for each combination of λ_{dep} and λ_{self}. To save compute, we restrict the range of λ_{self} to $[10^{-0.5}, 10^{1.2})$ while sampling λ_{dep} and λ_{self}, as the minimum interventional error usually lies in that range. Additionally, we utilize Bayesian optimization, employing the probability of improvement, to guide hyperparameter selection, thereby sampling more hyperparameters from promising regions where the interventional error is typically low. The interpolated heatmap showing the interventional errors for various chosen hyperparameters is shown in Fig. 18 (bottom).

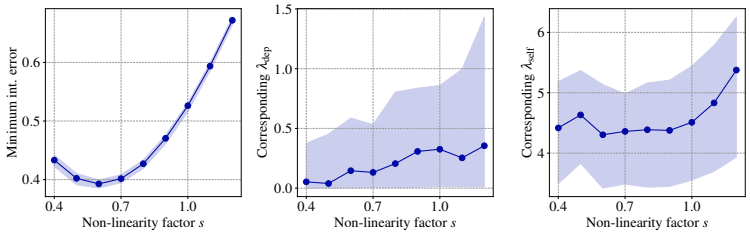


Figure 19: Variation in the minimum interventional error, and the corresponding λ_{dep} and λ_{self} when the non-linearity factor s is increased. The results are for the bottom-5% percentile lowest interventional error values obtained over 4500 runs for each value of s. The median values are shown using dark lines, and the region between the first and third quartiles is shaded.

Fig. 19 shows three plots: bottom-5% percentile of interventional error, and corresponding λ_{dep} and λ_{self} values. For each of these values, we plot the median value using dark blue curves, and the region between the first and third quartiles is shaded in light blue. As expected, the minimum interventional error increases with the non-linearity factor s. A similar trend can be observed for λ_{dep} and λ_{self}. Particularly, for λ_{dep}, the shaded region expands as s increases, indicating that higher values of λ_{dep} can now obtain very low values of interventional error.

I Generating Windmill Dataset

We provide the exact mathematical formulation of WINDMILL dataset described in Sec. 3.1. We define the following constants:

Constants	Description	Default value
n_{arms}	Number of "arms" in WINDMILL dataset	4
$r_{\max}$	Radius of the circular region spanned by the observed data	2
θ_{wid}	Angular width of each arm	$\frac{0.9\pi}{n_{\text{arms}}} = 0.7068$
λ_{off}	Offset wavelength. Determines the complexity of the dataset	6
$\theta_{\text{max-off}}$	Maximum offset for the angle	$\pi/6$

Table 14: Constants used for generating WINDMILL dataset, their meaning, and their values.

$$R_B \sim \mathcal{B}(1, 2.5) \qquad \text{(Sample radius)}$$

$$R = \frac{r_{\max}}{2}\left(BR_B + (1-B)(2-R_B)\right) \qquad \text{(Modify sampled radius based on } B\text{)}$$

$$\Theta_A \sim \mathcal{C}\left(\left\{2\pi\frac{i}{n_{\text{arms}}+1} : i = 0, \ldots, n_{\text{arms}}-1\right\}\right) \qquad \text{(Choose an arm)}$$

$$\Theta_{\text{off}} = \theta_{\text{max-off}} \sin\left(\pi\lambda_{\text{off}}\frac{R}{r_{\max}}\right) \qquad \text{(Calculate radial offset for the angle)}$$

$$U \sim \mathcal{U}(0, 1) \qquad \text{(To choose a random angle)}$$

$$\Theta = \theta_{\text{wid}}\left(U - 0.5\right) + A\left(\Theta_A + \frac{\pi}{n_{\text{arms}}}\right) + (1-A)\Theta_A + \Theta_{\text{off}}$$

$$\text{(Angle is decided by } A \text{ and the radial offset)}$$

$$X_1 = R\cos\Theta, X_2 = R\sin\Theta, X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \qquad \text{(Convert to Cartesian coordinates)}$$

PyTorch code to generate WINDMILL dataset is provided in Listing 1.

Listing 1: Code for Windmill dataset

```
import math
import torch

# Constants
num_arms = 4 # number of blades in the windmill
max_th_offset = 0.5236 # max offset that can be added to the angle for shearing (= pi/6)
r_max = 2 # length of the blade
num_p = 20000 # number of points to be generated
offset_wavelength = 6 # adjusts the complexity of the blade

# Sample latent variables according to the causal graph.
A = torch.bernoulli(torch.ones(num_points) * 0.6)
if observational_data:
    B = A
else:
    B = torch.bernoulli(torch.ones(num_points) * 0.5)

# Convert A, B to X.
th_A0 = torch.linspace(0, 2*math.pi, num_arms+1)[:-1]
th_A1 = torch.linspace(0, 2*math.pi, num_arms+1)[:-1] + math.pi/num_arms
# Choose a random arm for A=0 from possible arms. Likewise for A=1.
th_A0 = th_A0[torch.randint(num_arms, (num_p,))]
th_A1 = th_A1[torch.randint(num_arms, (num_p,))]

# beta distribution with alpha=1, beta=3
beta_dist = torch.distributions.beta.Beta(1, 2.5)

# Sample r according to B. If B=0, sample a small r, else sample a large r.
# r ranges from 0 to r_max
B0_r = beta_dist.sample(torch.Size([num_p])) * r_max/2.
B1_r = r_max - beta_dist.sample(torch.Size([num_p])) * r_max/2.
r = B * B0_r + (1-B) * B1_r

# Sample theta according to A.
# Choose the theta arm according to A and then sample from this arm using a uniform distribution.

# First we will have a cartwheel style.
theta = torch.rand(num_p)*th_wid + th_A0*(1-A) + th_A1*A - th_wid/2.

# Add an offset to theta according to r.
th_offset_mod = torch.sin((r/r_max)*offset_wavelength*math.pi)
th_offset = max_th_offset*th_offset_mod
theta += th_offset

x1 = r*torch.cos(theta)
x2 = r*torch.sin(theta)

data = torch.stack([x1, x2], dim=1)
labels = torch.stack([A, B], dim=1).type(torch.long)
```