A Data Generation Process via SCM

A.1 Assumptions on SCM

We summarize the detailed assumption our proposed SCM in **Assumption 1**.

Assumption 1 *(a) The domain variable U is the root node and does not directly link to Y or $\boldsymbol{X}$. (b) Z_i is generated by either Y or U for any i. (c) The CMB set of Y does not include U, and Y does not directly link to $\boldsymbol{X}$. (d) Z_i is the parent of $\boldsymbol{X}$, and $\boldsymbol{X}$ is the leaf node in the graph. (e) The causal graph over $\{\boldsymbol{X}, Y, \boldsymbol{Z}, U\}$ forms DAG.*

A.2 Generality of the Proposed SCM

We illustrate the distinct SCMs employed by prior causal representation learning methods in Figure 1. Figure 1(a), (b), and (d) share certain characteristics: 1) the input $\boldsymbol{X}$ can be generated through latent high-level factors $\boldsymbol{Z}$; 2) Distribution shifts arise from the domain-specific variable U. U induces changes in the distribution of input $\boldsymbol{X}$ by influencing the distribution of partial high-level latent variables $\boldsymbol{Z}$. In practical scenarios, the high-level latent factors $\boldsymbol{Z}$ are often unobservable, and only their transformation $\boldsymbol{X}$ is observable. Consequently, Invariant Causal Prediction (ICP) [8], which seeks parent variables $\boldsymbol{Z}_p$ from the observed set of $\boldsymbol{Z}$, may falter when applied to $\boldsymbol{X}$. [7] propose a general SCM setting in Figure 1(b). However, they categorize latent factors into two main types: causal factors (parent variables of Y) $\boldsymbol{Z}_p$ and non-causal factors (a collection of various latent variables except for parent variables) $\boldsymbol{Z}_{nc}$. While leveraging causal factors for out-of-distribution (OOD) prediction, they neglect factors that are also invariant and predictive of Y. To enhance prediction performance, we further specify $\boldsymbol{Z}_{nc}$ into different types of factors with respect to Y, as illustrated in Figure 1(c), i.e., $\boldsymbol{Z}_{nc} = \{\boldsymbol{Z}_o, \boldsymbol{Z}_c, \boldsymbol{Z}_s\}$. If we combine $(\boldsymbol{Z}_s, \boldsymbol{Z}_c)$ or $(\boldsymbol{Z}_o, \boldsymbol{Z}_c)$ in our proposed SCM, as shown in Figure 1(d), our proposed SCM is equivalent to the SCM employed by [7]. We hence contend that our SCM is as general as SCMs employed by existing methods. The only additional assumption we make is that

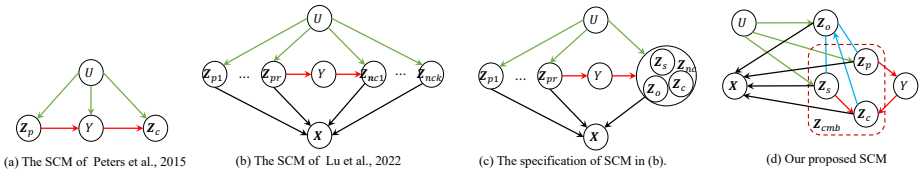

Fig. 1: Comparison between different SCMs.

the domain variable U is not within the CMB of Y, implying no direct causal links between U and child variables $\boldsymbol{Z}_c$. The crucial question is whether there

exists a set of variables solely controlled by target Y and not influenced by domain variable U. We provide examples to demonstrate that such an assumption is reasonable and easily satisfied in real-world scenarios. For instance, in a water bird vs. land bird dataset, the shape of a bird's wings and legs is influenced by the bird type, not the background it is in. In the CMNIST dataset, the shape of digits is influenced by the number they represent, not the image background or color.

B Identifiable Analysis on Variational Autoencoder

B.1 Preliminaries

For a VAE framework, $\boldsymbol{X} \in \mathbb{R}^d$ stands for the observed input data, and $\boldsymbol{Z} \in \mathbb{R}^N$ represents the latent variables. It has been proven that the VAE with an unconditional prior distribution $p(\boldsymbol{Z}) = \prod_i p(Z_i) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ over latent variables is unidentifiable [3]. To tackle the identifiable issue, [2] propose to posit a conditional factorized prior distribution over the latent representations, $p(\boldsymbol{Z}|U) = \prod_i p(Z_i|U), p(Z_i|U) \sim \mathcal{N}(\mu_u, \Sigma_u)$, where U is an auxiliary observed variable. iVAE [4] proposes a factorized prior conditioned on target Y and domain variable U, i.e., $p(\boldsymbol{Z}|Y, U)$, which belongs to a general exponential family, $p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y, U) = \prod_i \frac{\mathcal{Q}_i(Z_i)}{\mathcal{C}_i(Y,U)} \exp[\sum_{j=1}^k T_{i,j}(Z_i)\lambda_{i,j}(Y, U)]$. NF-iVAE [7] further improves the identifiable VAE framework by proposing a conditional, yet unfactorized prior distribution that can capture the dependency within $\boldsymbol{Z}$, i.e., $p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y, U) = \frac{\mathcal{Q}(\boldsymbol{Z})}{\mathcal{C}(Y,U)} \exp[\boldsymbol{T}(\boldsymbol{Z})^T \boldsymbol{\lambda}(Y, U)]$. $p(\boldsymbol{Z}|Y, U)$ is a generic form of the data generation process described by our proposed SCM in **Figure 1** of main paper. We thus adopt the NF-iVAE from [7] to obtain a set of component-wise identifiable $\boldsymbol{Z}$.

B.2 Identifiability of the Employed VAE Framework

In this section, we demonstrate that the latent representation $\boldsymbol{Z}$ learned by the NF-iVAE framework can achieve component-wise identifiability, which is crucial for our proposed CMB discovery. It is important to note that our contributions in this paper **do not** involve proposing a novel identifiable VAE framework or theoretically proving its identifiability. Instead, **we show that we can obtain the desired latent variables using the existing NF-iVAE framework. Our novel SCM assumptions do not affect the proof presented in [7], and the identifiability results still hold.** The identifiability of the latent variables has been thoroughly investigated in the original NF-iVAE paper. We summarize the theoretical analysis from [7] in the following sections.

First, we define the distinct levels of identifiability in **Definition 1**, **Definition 2**, and **Definition 3**. Moreover, we make a similar assumption (**Assumption 2**) for the prior distribution as **Assumption 2** in [7].

Definition 1. *Let Ξ be the domain of the parameters $\boldsymbol{\xi} = \{\boldsymbol{\phi}, \boldsymbol{T}, \boldsymbol{\lambda}\}$. Let $\sim$ be an equivalence relation on Ξ. A generative model is $\sim$-identifiable if*

$$p_{\boldsymbol{\xi}}(\boldsymbol{X}) = p_{\tilde{\boldsymbol{\xi}}}(\boldsymbol{X}) \implies \boldsymbol{\xi} \sim \tilde{\boldsymbol{\xi}} \tag{1}$$

Definition 2. *Let $\sim_A$ be an equivalence relation on Ξ defined by:*

$$(\boldsymbol{\phi}, \boldsymbol{T}, \boldsymbol{\lambda}) \sim_A (\tilde{\boldsymbol{\phi}}, \tilde{\boldsymbol{T}}, \tilde{\boldsymbol{\lambda}}) \iff \exists A, \boldsymbol{c}\ s.t. \boldsymbol{T}(g_{\boldsymbol{\phi}}^{-1}(\boldsymbol{X})) = A\tilde{\boldsymbol{T}}(g_{\tilde{\boldsymbol{\phi}}}^{-1}(\boldsymbol{X})) + \boldsymbol{c} \tag{2}$$

where $A \in \mathbb{R}^{k \times k}$ is an invertible matrix and $\boldsymbol{c} \in \mathbb{R}^k$ is a vector.

Definition 3. *Let $\sim_P$ be an equivalence relation on Ξ defined by:*

$$(\boldsymbol{\phi}, \boldsymbol{T}, \boldsymbol{\lambda}) \sim_P (\tilde{\boldsymbol{\phi}}, \tilde{\boldsymbol{T}}, \tilde{\boldsymbol{\lambda}}) \iff \exists P, \boldsymbol{c}\ s.t. \boldsymbol{T}(g_{\boldsymbol{\phi}}^{-1}(\boldsymbol{X})) = P\tilde{\boldsymbol{T}}(g_{\tilde{\boldsymbol{\phi}}}^{-1}(\boldsymbol{X})) + \boldsymbol{c} \tag{3}$$

where $P \in \mathbb{R}^{k \times k}$ is a block permutation matrix and $\boldsymbol{c} \in \mathbb{R}^k$ is a vector.

Assumption 2 *$p(\boldsymbol{Z}|Y, U)$ belongs to a general exponential family with parameters given by an arbitrary function $\boldsymbol{\lambda}(Y, U)$ and sufficient statistics $\boldsymbol{T}(\boldsymbol{Z}) = [\boldsymbol{T}_f(\boldsymbol{Z})^T, \boldsymbol{T}_{NN}(\boldsymbol{Z})^T]^T$. $\boldsymbol{T}_f(\boldsymbol{Z}) = [\boldsymbol{T}_1(Z_1)^T, \cdots, \boldsymbol{T}_N(Z_N)^T]^T$ are the sufficient statistics of a factorized exponential family and $\boldsymbol{T}_i(Z_i)$ have the dimension k, where $k \geq 2$. $\boldsymbol{T}_{NN}(\boldsymbol{Z})$ capture the dependency between $\boldsymbol{Z}$. The resulting density function is thus given by*

$$p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y, U) = \frac{\mathcal{Q}(\boldsymbol{Z})}{\mathcal{C}(Y, U)} \exp\left[\boldsymbol{T}(\boldsymbol{Z})^T \boldsymbol{\lambda}(Y, U)\right] \tag{4}$$

where $\mathcal{Q}$ is the base measure and $\mathcal{C}$ is the normalizing constant.

We provide a high-level summary of identifiable results in **Theorem 1** in the main paper. In this section, we provide a sketch of the proof and decompose it into

- **Theorem 1**: the model parameters $\boldsymbol{\xi}$ are $\sim_P$ identifiable in using assumptions (i) - (iv) in **Theorem 1** and **Assumption 2**.
- **Theorem 2**: our learning framework can learn the true parameters $\boldsymbol{\xi}^*$ up to a permutation and componentwise transformation if assumption (i) in **Theorem 2** and **Theorem 1** are satisfied.
- **Theorem 3**: the learned latent variables $\boldsymbol{Z}^*$ identifiable up to a permutation and componentwise transformation, subject to **Theorem 1**, **2**, and infinite data assumption.

By employing a general form of prior $p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y, U)$, the data generation that subject to Assumption 1 and SCM in Figure 1 of main paper can be described via following Equations:

$$\begin{aligned} p_{\boldsymbol{\xi}=(\boldsymbol{\phi},\boldsymbol{T},\boldsymbol{\lambda})}(\boldsymbol{X}, \boldsymbol{Z}|Y, U) =& p_{\boldsymbol{\phi}}(\boldsymbol{X}|\boldsymbol{Z}) p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y, U) \\ p_{\boldsymbol{\phi}}(\boldsymbol{X}|\boldsymbol{Z}) =& p_{\boldsymbol{\epsilon}}(\boldsymbol{X} - g_{\boldsymbol{\phi}}(\boldsymbol{Z})) \end{aligned} \tag{5}$$

Theorem 1. *Assume that we observe data sampled from a generative mechanism described by Eq. (5) and Eq. (4), with parameters* $\boldsymbol{\xi} := (\boldsymbol{\phi}, \boldsymbol{T}, \boldsymbol{\lambda})$ *where* $p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y,U)$ *satisfies* ***Assumption 2****. Furthermore, we assume the following holds: (i) The set* $\{\boldsymbol{X}|\varphi_{\boldsymbol{\epsilon}}(\boldsymbol{X}) = 0\}$ *has measure zero, where* $\varphi_{\boldsymbol{\epsilon}}$ *is the characteristic function of the density* $p_{\boldsymbol{\epsilon}}$ *defined in Eq. (5). (ii) Function* g *in Eq. (5) is injective, and has all second-order cross derivatives. (iii) The sufficient statistics in* $\boldsymbol{T}_f$ *are all twice differentiable. (iv) There exist* $k+1$ *distinct points* $(Y^0,U^0),(Y^1,U^1),\cdots,(Y^k,U^k)$ *such that the matrix* $L = \Big(\boldsymbol{\lambda}(Y^1,U^1) - \boldsymbol{\lambda}(Y^0,U^0),\cdots,\boldsymbol{\lambda}(Y^k,U^k) - \boldsymbol{\lambda}(Y^0,U^0)\Big)$ *of size* $k \times k$ *is invertible, where* k *is the dimension of* $\boldsymbol{T}$*. Then the parameter* $\boldsymbol{\xi}$ *are identifiable up to a permutation and componentwise transformation, i.e.,* $\sim_P$ *identifiable.*

To be more precise, we require $\boldsymbol{T}_f$ to be twice differentiable. Appendix H.1 of [7] provides detailed proof. The proof includes two parts:

- The parameters $\boldsymbol{\xi}$ are $\sim_A$ identifiable using assumption (i), assumption (ii) and assumption (iv).
- Based on the $\sim_A$ identifiable results, it further proves that the parameters $\boldsymbol{\xi}$ are $\sim_P$ identifiable by using **Assumption 2** and assumption (ii) and (iii).

Theorem 2. *Assume the following assumptions hold: (i) The family of distributions* $q_{\boldsymbol{\theta}}(\boldsymbol{Z}|\boldsymbol{X},Y,U)$ *contains* $p_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X},Y,U)$*, and* $q_\theta(\boldsymbol{Z}|\boldsymbol{X},Y,U) > 0$ *everywhere. (ii) The NF-iVAE learning framework, which minimizes* $\mathcal{L}_{obj}(\boldsymbol{\xi},\boldsymbol{\theta})$ *in Eq. (6) with respect to both* $\boldsymbol{\xi}$ *and* $\boldsymbol{\theta}$*, can learn the true parameters* $\boldsymbol{\xi}^*$ *up to a permutation and simple transformation of the latent variable* $\boldsymbol{Z}$ *in the limit of infinite data.*

Proof. We recall from the loss function in Phase I is as follows:

$$\mathcal{L}_{\text{obj}}(\boldsymbol{\theta},\boldsymbol{\xi}) \coloneqq \mathcal{L}_{\text{ELBO}}(\boldsymbol{\theta},\boldsymbol{\phi},\hat{\boldsymbol{T}},\hat{\boldsymbol{\lambda}}) + \mathcal{L}_{\text{SM}}(\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\phi}},\boldsymbol{T},\boldsymbol{\lambda}) \tag{6}$$

$$\mathcal{L}_{\text{ELBO}} \coloneqq -\mathbb{E}_{p_{\mathcal{D}}}\big[\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u)}[\log p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z}) + \log p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|y,u) - \log q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u]\big] \tag{7}$$

$$\mathcal{L}_{\text{SM}} \coloneqq \mathbb{E}_{p_{\mathcal{D}}}\big[\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u)}[\|\nabla_{\boldsymbol{z}}\log q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u) - \nabla_{\boldsymbol{z}}\log p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|y,u)\|^2]\big] \tag{8}$$

If the family of $q_{\boldsymbol{\theta}}(\boldsymbol{Z}|\boldsymbol{X},Y,U)$ is flexible enough to contain $p_{\boldsymbol{\xi}}(\boldsymbol{Z}|\boldsymbol{X},Y,U)$, then by optimizing the $\mathcal{L}_{\text{obj}}$ over its parameter $\boldsymbol{\xi}$, the score matching term $\mathcal{L}_{\text{SM}}$ is minimized and eventually reach zero. If we assume that the model is not degenerate and that $q_{\boldsymbol{\phi}} > 0$ everywhere, then we have

$$\begin{aligned} \mathcal{L}_{\text{SM}} = 0 &\implies \nabla_{\boldsymbol{z}}\log q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u) = \nabla_{\boldsymbol{z}}\log p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|y,u) \\ &\implies \log q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u) = \log p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|y,u) + c \end{aligned} \tag{9}$$

for some constant c. c is zero because both $q_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x},y,u)$ and $p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|y,u)$ are pdf's. Therefore, the $\mathcal{L}_{\text{obj}}$ will be equal to the log-likelihood. Under such circumstances, the estimation in Eq. (6) inherits all the properties of maximum

likelihood estimation (MLE). Since our identifiability is guaranteed up to a permutation and componentwise transformation, the consistency of MLE indicates that we converge to the true parameters $\boldsymbol{\xi}^*$ up to a permutation and componentwise transformation in the limit of infinite data.

Theorem 3. *Assume that Theorem 1 and Theorem 2 hold, then in the limit of infinite data, the true latent variables $\boldsymbol{Z}^*$ are identifiable up to a permutation and componentwise transformation.*

Proof. **Theorem 2** and **Theorem 3** guarantee that in the limit of infinite data, NF-iVAE can obtain the true parameters $\boldsymbol{\xi}^* := (\boldsymbol{\phi}^*, \boldsymbol{T}^*, \boldsymbol{\lambda}^*)$ up to a permutation and componentwise transformation of the latent variables. We denote the parameters obtained from NF-iVAE as $\hat{\boldsymbol{\xi}} := (\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{T}}, \hat{\boldsymbol{\lambda}})$, i.e., $(\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{T}}, \hat{\boldsymbol{\lambda}}) \sim_P (\boldsymbol{\phi}^*, \boldsymbol{T}^*, \boldsymbol{\lambda}^*)$. If there were no noise, we have $\hat{\boldsymbol{Z}} = g_{\hat{\boldsymbol{\phi}}}^{-1}(\boldsymbol{X})$ that are equal to $\boldsymbol{Z}^* = g_{\boldsymbol{\phi}^*}^{-1}(\boldsymbol{X})$ up to a permutation and componentwise transformation. If with noise, we can obtain the posterior distribution of the latent variables up to an analogous indeterminacy.

Similar proofs for **Theorem 2** and **Theorem 3** are also provided in [3, 7].

B.3 Derivation of ELBO loss in Eq.(2) of Main Paper

We start with the joint distribution of observed variables $\boldsymbol{X}, Y, U$, i.e., $p(\boldsymbol{X}, Y, U)$.

$$
\begin{aligned}
& -\log p(\boldsymbol{x}, y, u) \\
=& -\log p(\boldsymbol{x}|y,u)p(y,u) \quad y,u \text{ are observed variables. } p(y,u) \text{ is known.} \\
=& -\log p(y,u) - \log p(\boldsymbol{x}|y,u) \\
=& -\log p(\boldsymbol{x}|y,u) + c \quad -\log p(y,u) \text{ is constant.} \\
=& -\log \int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}|y,u)\,\mathrm{d}\boldsymbol{z} + c \\
=& -\log \int_{\boldsymbol{z}} p(\boldsymbol{x}|\boldsymbol{z},y,u)p(\boldsymbol{z}|y,u)\,\mathrm{d}\boldsymbol{z} + c \\
=& -\log \int_{\boldsymbol{z}} p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}|y,u)\,\mathrm{d}\boldsymbol{z} + c \quad \boldsymbol{X} \perp\!\!\!\perp Y, U|\boldsymbol{Z} \\
=& -\log \int_{\boldsymbol{z}} \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}|y,u)}{q(\boldsymbol{z}|\boldsymbol{x},y,u)} q(\boldsymbol{z}|\boldsymbol{x},y,u)\,\mathrm{d}\boldsymbol{z} + c \\
=& -\log \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y,u)}\big[\frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}|y,u)}{q(\boldsymbol{z}|\boldsymbol{x},y,u)}\big] + c \\
\leq& -\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y,u)}\big[\log \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}|y,u)}{q(\boldsymbol{z}|\boldsymbol{x},y,u)}\big] + c \\
=& -\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y,u)}\big[\log p(\boldsymbol{x}|\boldsymbol{z}) + \log p(\boldsymbol{z}|y,u) - \log q(\boldsymbol{z}|\boldsymbol{x},y,u)\big] + c
\end{aligned}
\tag{10}
$$

According to Eq. (10), $-\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y,u)}\big[\log p(\boldsymbol{x}|\boldsymbol{z}) + \log p(\boldsymbol{z}|y,u) - \log q(\boldsymbol{z}|\boldsymbol{x},y,u)\big] + c$ is upper bound of the negative log joint likelihood over observed variables

$\boldsymbol{X}, Y, U$. We ignore the constant term in training since there is no parameter to optimize. Its expected value over data observations from the training distribution $p_{\mathcal{D}}$ is defined as ELBO loss in Eq. (2).

B.4 Decomposition of Prior $p(\boldsymbol{Z}|\boldsymbol{Y}, \boldsymbol{U})$

We aim to show that the generic form prior $p(\boldsymbol{z}|y, u)$ is consistent with our SCM. To do so, we demonstrate that $p(\boldsymbol{z}|y, u)$ can be further decomposed into the product of conditional probabilities on one of the SCMs that satisfy our assumptions. Assume there are four latent variables, one variable for each type. Z_1 is the parent variable of Y ($Z_1 \to Y, U \to Z_1$), Z_2 the child variable ($Y \to Z_2$), Z_3 the spouse variable ($Z_3 \to Z_2$, $U \to Z_3$), Z_4 the spurious variable ($U \to Z_4$). We also allow for the links between spurious variables and CMB variables, e.g., $Z_4 \to Z_1$. The SCM constructed following the above descriptions is illustrated in Fig. 2. It satisfies **Assumption 1**. We employ this SCM as an example and demonstrate that $p(\boldsymbol{z}|y, u)$ can be further decomposed.

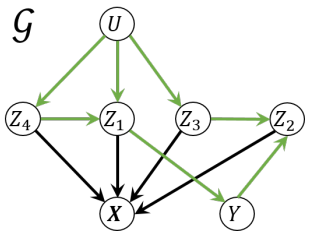

Fig. 2: An example of SCM satisfies **Assumption 1**.

$$
\begin{aligned}
& p(\boldsymbol{x}, y, u) \\
=& \int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}, y, u)\, \mathrm{d}\boldsymbol{z} \\
=& \int_{\boldsymbol{z}} p(u)p(z_3|u)p(z_4|u)p(z_1|z_4, u)p(y|z_1)p(z_2|y, z_3)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
& (Z_2 \perp\!\!\!\perp Z_1|Y, Z_3)_{\mathcal{G}} \text{ and } (Y \perp\!\!\!\perp Z_3|Z_1)_{\mathcal{G}} \\
=& \int_{\boldsymbol{z}} p(u)p(z_3|u)p(z_4|u)p(z_1|z_4, u)p(y|z_1, z_3)p(z_2|y, z_1, z_3)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
=& \int_{\boldsymbol{z}} p(u)p(z_3|u)p(z_4|u)p(z_1|z_4, u)p(z_2, y|z_1, z_3)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
& (Z_3 \perp\!\!\!\perp \{Z_1, Z_4\}|U)_{\mathcal{G}} \text{ and } (\{Z_2, Y\} \perp\!\!\!\perp \{Z_4, U\}|Z_1, Z_3)_{\mathcal{G}} \\
=& \int_{\boldsymbol{z}} p(u)p(z_3|u, z_1, z_4)p(z_1, z_4|u)p(z_2, y|z_1, z_3, z_4, u)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
=& \int_{\boldsymbol{z}} p(u)p(z_1, z_2, z_3, z_4, y|u)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
=& \int_{\boldsymbol{z}} p(u)p(z_1, z_2, z_3, z_4|y, u)p(y|u)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
=& p(y, u) \int_{\boldsymbol{z}} p(\boldsymbol{z}|y, u)p(\boldsymbol{x}|\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}
\end{aligned}
\tag{11}
$$

C Causal Markov Blanket Identification

C.1 Proof of Equivalent Mutual Information

Proof. We aim to prove that if $Z_i \in \boldsymbol{Z}_o$, then $\mathcal{I}(Z_i; Y|\boldsymbol{Z}_{\setminus i}, U) = \mathcal{I}(Z_i; Y|\boldsymbol{Z}_{\setminus i})$. We have

$$\begin{aligned} &\mathcal{I}(Z_i; Y|\boldsymbol{Z}_{\setminus i}, U) \\ =&\mathcal{I}(Y; \{Z_i, U\}|\boldsymbol{Z}_{\setminus i}) - \mathcal{I}(Y; U|\boldsymbol{Z}_{\setminus i}) \quad \text{Chain rule of mutual information} \\ =&\mathcal{I}(Y; Z_i|\boldsymbol{Z}_{\setminus i}) + \mathcal{I}(Y; U|\{\boldsymbol{Z}_{\setminus i}, Z_i\}) - \mathcal{I}(Y; U|\boldsymbol{Z}_{\setminus i}) \\ =&\mathcal{I}(Y; Z_i|\boldsymbol{Z}_{\setminus i}) + \mathcal{I}(Y; U|\boldsymbol{Z}) - \mathcal{I}(Y; U|\boldsymbol{Z}_{\setminus i}) \end{aligned} \tag{12}$$

If $Z_i \in \boldsymbol{Z}_o$, then $\boldsymbol{Z}_{cmb} \subset \boldsymbol{Z}_{\setminus i}$. According to Figure 1 of main paper, we have $Y \perp\!\!\!\perp U|\boldsymbol{Z}$ and $Y \perp\!\!\!\perp U|\boldsymbol{Z}_{\setminus i}$. Hence $\mathcal{I}(Y; U|\boldsymbol{Z}) = 0$ and $\mathcal{I}(Y; U|\boldsymbol{Z}_{\setminus i}) = 0$. Then we have $\mathcal{I}(Z_i; Y|\boldsymbol{Z}_{\setminus i}, U) = \mathcal{I}(Y; Z_i|\boldsymbol{Z}_{\setminus i})$.

C.2 Proof for Proposition 1

Proof. If $Z_i \in \boldsymbol{Z}$ and $Z_i \notin \boldsymbol{Z}_{cmb}$, then $Z_i \in \boldsymbol{Z}_o$. According to Definition 1, $Z_i \perp\!\!\!\perp Y|\boldsymbol{Z}_{cmb}$. Since $Z_i \notin \boldsymbol{Z}_{cmb}$, we know that $\boldsymbol{Z}_{cmb} \subset \boldsymbol{Z}_{\setminus i}$, hence we have $Z_i \perp\!\!\!\perp Y|\boldsymbol{Z}_{\setminus i}$. If $Z_i \in \boldsymbol{Z}_{cmb}$, then the identify of Z_i has three cases: 1) $Z_i \in \boldsymbol{Z}_p$, 2) $Z_i \in \boldsymbol{Z}_c$, and 3) $Z_i \in \boldsymbol{Z}_s$. For cases 1 and 2, since there exist direct causal links between Z_i and Y, Z_i is not independent of Y given any set. For case 3, there must exist a $Z_j, j \neq i$, Z_j is the mutual child of Y and Z_i. Moreover, $Z_j \in \boldsymbol{Z}_{\setminus i}$, condition on $\boldsymbol{Z}_{\setminus i}$, the path $Z_i \leftarrow Z_j \rightarrow Y$ is unblocked, hence $Z_j \not\perp\!\!\!\perp Y|\boldsymbol{Z}_{\setminus i}$.

C.3 Proof of Proposition 2

Proof. Assuming accurate independence test results, variables in $\boldsymbol{Z}_{cmb}$ are included in Step 1 and forward procedure in Step 2 of Algorithm 1. This is because they are dependent on Y in any subset, under faithfulness condition. If a variable is not a member of $\boldsymbol{Z}_{cmb}$, then it will be removed from CMB set in the backward procedure in Step 2 since conditioned on $\boldsymbol{Z}_{cmb}$, it is independent of Y by the minimal property of MB definition. As a result, only ground truth $\boldsymbol{Z}_{cmb}$ will by returned by Algorithm 1. The procedure is the same used by IAMB [11].

C.4 Mutual Information Estimation

Given $\{\boldsymbol{z}(j)\}_{j=1}^{M}, \boldsymbol{z}(j) = \{z_1(j), z_2(j), \cdots, z_i(j), \cdots, z_N(j)\}$ and learned distribution $p(Y|\boldsymbol{Z})$. We aim to calculate (conditional) mutual information $\mathcal{I}(y; z_i|\mathcal{C})$, where $\mathcal{C}$ is the set of variables to condition on. According to the definition of mutual information, we have:

$$\mathcal{I}(y; z_i|\mathcal{C}) = \mathcal{H}[y|\mathcal{C}] - \mathcal{H}[y|\{\mathcal{C}, z_i\}]$$

By the definition of the conditional entropy, the first term can be computed via:

$$\begin{aligned}
&\mathcal{H}[y|\mathcal{C}] \\
=&\mathbb{E}_{p(\mathcal{C})}[\mathcal{H}\big(p(y|\mathcal{C})\big)] \\
=&\mathbb{E}_{p(\mathcal{C})}[\mathcal{H}\big(\mathbb{E}_{p(\boldsymbol{z}\backslash\mathcal{C}|\mathcal{C})}[p(y|\boldsymbol{z})]\big)] \\
\approx&\mathbb{E}_{p(\mathcal{C})}[\mathcal{H}\big(\mathbb{E}_{p(\boldsymbol{z}\backslash\mathcal{C})}[p(y|\boldsymbol{z})]\big)] \text{ we approximate } p(\boldsymbol{z}\backslash\mathcal{C}|\mathcal{C}) \text{ with } p(\boldsymbol{z}\backslash\mathcal{C}) \\
\approx&\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}-\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\mathcal{C}(j),\{\boldsymbol{z}\backslash\mathcal{C}\}(k)\big)}{M}\Big)\log\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\mathcal{C}(j),\{\boldsymbol{z}\backslash\mathcal{C}\}(k)\big)}{M}\Big)
\end{aligned} \tag{13}$$

Similarly, the second term can be computed via:

$$\begin{aligned}
&\mathcal{H}[y|\{\mathcal{C},z_i\}] \\
=&\mathbb{E}_{p(\{\mathcal{C},z_i\})}[\mathcal{H}\big(p(y|\{\mathcal{C},z_i\})\big)] \\
=&\mathbb{E}_{p(\{\mathcal{C},z_i\})}[\mathcal{H}\big(\mathbb{E}_{p(\boldsymbol{z}\backslash\{\mathcal{C},z_i\}|\{\mathcal{C},z_i\})}[p(y|\boldsymbol{z})]\big)] \\
\approx&\mathbb{E}_{p(\{\mathcal{C},z_i\})}[\mathcal{H}\big(\mathbb{E}_{p(\boldsymbol{z}\backslash\{\mathcal{C},z_i\})}[p(y|\boldsymbol{z})]\big)] \\
\approx&\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}-\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\{\mathcal{C},z_i\}(j),\{\boldsymbol{z}\backslash(\mathcal{C},z_i)\}(k)\big)}{M}\Big)\log\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\{\mathcal{C},z_i\}(j),\{\boldsymbol{z}\backslash(\mathcal{C},z_i)\}(k)\big)}{M}\Big)
\end{aligned} \tag{14}$$

Hence, Substituting Eq. (13) and Eq. (14) into Eq. (3) of main paper, we have,

$$\begin{aligned}
&\mathcal{I}(y,z_i|\mathcal{C}) \\
=&-\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\mathcal{C}(j),\{\boldsymbol{z}\backslash\mathcal{C}\}(k)\big)}{M}\Big)\log\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\mathcal{C}(j),\{\boldsymbol{z}\backslash\mathcal{C}\}(k)\big)}{M}\Big)+ \\
&\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\{\mathcal{C},z_i\}(j),\{\boldsymbol{z}\backslash(\mathcal{C},z_i)\}(k)\big)}{M}\Big)\log\Big(\frac{\sum_{k=1}^{M}p\big(y=l|\{\mathcal{C},z_i\}(j),\{\boldsymbol{z}\backslash(\mathcal{C},z_i)\}(k)\big)}{M}\Big)
\end{aligned} \tag{15}$$

For Step 1, if we set $\mathcal{C}=\boldsymbol{z}_{\backslash i}$, for the first term we have:

$$\mathcal{H}[y|\boldsymbol{z}_{\backslash i}]\approx\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}-\Big(\frac{1}{M}\sum_{k=1}^{M}p(y=l|\boldsymbol{z}_{\backslash i}(j),z_i(k))\Big)\log\Big(\frac{1}{M}\sum_{k=1}^{M}p(y=l|\boldsymbol{z}_{\backslash i}(j),z_i(k))\Big) \tag{16}$$

For the second term we have:

$$\mathcal{H}[y|\boldsymbol{z}]\approx\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}-p\big(y=l|\boldsymbol{z}(j)\big)\log p\big(y=l|\boldsymbol{z}(j)\big) \tag{17}$$

Substitute Eq. (16) and Eq. (17) into Eq. (3), we have,

$$\begin{aligned}
\mathcal{I}(y;z_i|\boldsymbol{z}_{\backslash i})=&-\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}\frac{\sum_{k=1}^{M}p\big(y=l|\boldsymbol{z}_{\backslash i}(j),z_i(k)\big)}{M}\log\frac{\sum_{k=1}^{M}p\big(y=l|\boldsymbol{z}_{\backslash i}(j),z_i(k)\big)}{M} \\
&+\frac{1}{M}\sum_{j=1}^{M}\sum_{l=0}^{L-1}p\big(y=l|\boldsymbol{z}(j)\big)\log p\big(y=l|\boldsymbol{z}(j)\big)
\end{aligned} \tag{18}$$

D Complexity Analysis of Inference Procedure

We assume there are M images from the target domain, with K and L add or multiply operations for forward/backward propagation in the decoder $p(\boldsymbol{X}|\boldsymbol{Z})$

and S operations for forward propagation in the predictor. We assume N optimization steps to obtain optimal $\boldsymbol{Z}$ values, setting a maximum of 1000 iterations with early stopping for efficiency. The total operations required are $MN(K+L)+MS$, compared to around $MK+MS$ for regular inference.

E Implementation Details

E.1 Datasets

We experiment on a synthetic dataset, Colored-MNIST (CMNIST), and three real datasets: CelebA, PACS, and VLCS. **CMNIST**: We create a dataset with three domains (2 for training and 1 for testing) following [1]. The task is to predict a binary label assigned to each image that is artificially colored to be strongly but spuriously correlated with the class label, with the test domain data exhibiting an inverse correlation compared to the training domain data. **CelebA** is a widely used computer vision dataset [6]. Following [12], we generate train and test sets with intentionally designed distribution shifts. For each pair of target and spurious attributes, the training set includes samples with high statistical correlations between the target and spurious attributes, while the test set includes samples with opposite or low correlations within the same attribute pair. This process results in 5 sets of training and test data for each attribute pair, each containing $n = 5000$ images. The **PACS** dataset, introduced by [5], contains images from four domains: Photo (P), Art painting (A), Cartoon (C), and Sketch (S), with each domain subdivided into 7 categories. The **VLCS** dataset, presented by [10], contains images categorized into 5 classes from four domains: PASCAL VOC 2007 (P), LabelMe (L), Caltech (C), and Sun (S). Following established DG methods, we employ the standard leave-one-domain-out protocol for both PACS and VLCS datasets. This protocol involves conducting testing on images from one domain while training occurs on the remaining domains.

E.2 Architecture in Our Implementation

For the CMNIST dataset, we adopt the architectural configurations from one of our baselines, as described by [7], utilizing multilayer perceptrons (MLP) as both encoder and decoder components. In the case of the CelebA dataset, we employ a neural network with three convolutional layers as the encoder and a neural network with three deconvolutional layers as the decoder. For PACS and VLCS datasets, we leverage a pre-trained ResNet-50 on ImageNet as the encoder backbone, complemented by a decoder of comparable complexity. In all cases, the classifier consists of a two-layer MLP. For comprehensive insights into training parameters and architectures, please consult Appendix E.

In our experiment, the encoder, decoder, prior model, and the classifier are implemented as follows:

Prior Model: The prior model consists of three neural networks $\boldsymbol{T}_{NN}, \boldsymbol{\lambda}_{NN}, \boldsymbol{\lambda}_f$. We define $\boldsymbol{T}_f$ as $\texttt{concat}(\boldsymbol{Z}, \boldsymbol{Z}^2)$. The density function can be computed via the following equation:

$$p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{Z}|Y,U) = \langle \boldsymbol{T}_{NN}(\boldsymbol{Z}), \boldsymbol{\lambda}_{NN}(Y,U) \rangle + \langle \boldsymbol{T}_f(\boldsymbol{Z}), \boldsymbol{\lambda}_f(Y,U) \rangle$$

- $\boldsymbol{T}_{NN}$**:** Input size: (batch size, dimension of $\boldsymbol{Z}$)
  - Layer 1: Fully connected layer, output size = 50, activation=ReLU
  - Layer 2: Fully connected layer, output size = 45
- $\boldsymbol{\lambda}_{NN}$**:** Input size: (batch size, Number of classes for (Y,U))
  - Layer 1: Fully connected layer, output size = 50, activation=ReLU
  - Layer 2: Fully connected layer, output size = 45
- $\boldsymbol{\lambda}_f$**:** Input size: (batch size, Number of classes for (Y,U))
  - Layer 1: Fully connected layer, output size = 50, activation=ReLU
  - Layer 2: Fully connected layer, output size = $2N$, N is the dimension of $\boldsymbol{Z}$.

$\boldsymbol{X}$-Encoder:

- CMNIST: Input size: (batch size, 2, 28, 28)
  - Layer 1: Convolutional layer, output channels=32, kernel size=3, stride=2, padding=1, activation=ReLU
  - Layer 2: Convolutional layer, output channels=32, kernel size=3, stride=2, padding=1, activation=ReLU
  - Layer 3: Convolutional layer, output channels=32, kernel size=3, stride=2, padding=1, activation=ReLU
  - Layer 4: Flatten layer
- CelebA: Input size: (batch size, 3, 64, 64)
  - Layer 1: Convolutional layer, output channels=64, kernel size=4, stride=2, padding=1, activation=ReLU, batch normalization
  - Layer2: Convolutional layer, output channels=128, kernel size=4, stride=2, padding=1, activation=ReLU, batch normalization
  - Layer3: Convolutional layer, output channels=256, kernel size=4, stride=2, padding=1, activation=ReLU, batch normalization
  - Layer4: Convolutional layer, output channels=512, kernel size=4, stride=2, padding=1, activation=ReLU, batch normalization
  - Layer5: Convolutional layer, output channels=512, kernel size=4, stride=2, padding=1, activation=ReLU, batch normalization
  - Layer 6: Flatten layer
- PACS and VLCS: Input size: (batch size, 3, 224, 224), we employ the ResNet50 without the last fully connected layer as the encoder for $\boldsymbol{X}$.

(Y,U)**-Encoder:** Input size: (batch size, Number of classes for (Y,U))

- Layer 1: Fully connected layer, output size = 100, activation=ReLU

Merge-Encoder: Input size: (batch size, Number of dimensions after Flatten layer + Number of classes for (Y,U))

- Layer 1: Fully connected layer, output size=100, activation=ReLU
- Mean output layer: Fully connected layer, output size = Number of dimension for $\boldsymbol{Z}$
- Log variance output layer: Fully connected layer, output size = Number of dimensions for $\boldsymbol{Z}$

Decoder:

- CMNIST: Input size: (batch size, dimension of $\boldsymbol{Z}$)
  - Layer 1: Fully connected layer, output size=$32 \times 4 \times 4$, activation=ReLU
  - Layer 2: Reshape to (batch size, 32, 4, 4)
  - Layer 3: Deconvolutional layer, output channels=32, kernel size=3, stride=2, padding=1, outpadding=0, activation=ReLU
  - Layer 4: Deconvolutional layer, output channels=32, kernel size=3, stride=2, padding=1, outpadding=1, activation=ReLU
  - Layer 5: Deconvolutional layer, output channels=2, kernel size=3, stride=2, padding=1, outpadding=0, activation=ReLU
  - Output layer: sigmoid activation
- CelebA: Input size: (batch size, dimension of $\boldsymbol{Z}$)
  - Layer 1: Fully connected layer, output size=$64 \times 8 \times 2 \times 2$, activate =leakyReLU(0.2)
  - Component 1: 1) Upsampling layer with scale factor=2, 2) Deconvolutional layer, output channel=512, kernel size=3, stride=1, activate =leakyReLU(0.2), 3) batch normalization
  - Component 2: 1) Upsampling layer with scale factor=2, 2) Deconvolutional layer, output channel=256, kernel size=3, stride=1, activate =leakyReLU(0.2), 3) batch normalization
  - Component 3: 1) Upsampling layer with scale factor=2, 2) Deconvolutional layer, output channel=128, kernel size=3, stride=1, activate =leakyReLU(0.2), 3) batch normalization
  - Component 4: 1) Upsampling layer with scale factor=2, 2) Deconvolutional layer, output channel=64, kernel size=3, stride=1, activate = leakyReLU(0.2), 3) batch normalization
  - Component 5: 1) Upsampling layer with scale factor=2, 2) Deconvolutional layer, output channel=3, kernel size=3, stride=1, activate = leakyReLU(0.2), 3) batch normalization
- PACS and VLCS: We reverse the architecture of `ResNet`50 as a decoder.

Classifier: Input size: (batch size, input dimension)

- Layer 1: Fully connected layer, output size=256, activation=ReLU
- Layer 2: Fully connected layer, output size=Number of classes for Y, activation =sigmoid

E.3 Choices of Hyperparameters

We discuss how to choose the proper hyperparameters in our training and inference procedure:

- λ_1 **and** λ_2**:** We choose the λ_1 and λ_2 in Eq. (7) of main paper on the training and validation data. Empirically, our method yields accurate $\boldsymbol{Z}$ inference by setting $\lambda_1 = 1, \lambda_2 = 1$.
- $N = |\boldsymbol{Z}|$**:** The optimal number of latent variables N varies depending on the dataset. A small N may result in the latent variables $\boldsymbol{Z}$ failing to capture sufficient information from the input $\boldsymbol{X}$. Conversely, a large N can lead to the violation of assumption (iv) in **Theorem 1**, and increase the complexity of CMB discovery, thus compromising the efficiency of our proposed CMBRL algorithm. In practice, we use $N = 32$ for CMNIST and CelebA, and $N = 50$ for PACS and VLCS.
- **Significance Level** α **for MI:** The commonly used significance levels α in the community for CI tests-based MB discovery are $\alpha = 0.01, 0.005, 0.001, \ldots$. The optimal choice of α can be empirically determined by evaluating performance on training and validation data. In our experiments, we found that setting $\alpha = 0.005$ achieves optimal performance for both in-distribution and out-of-distribution predictions across various settings and datasets. A detailed ablation study of the threshold α is presented in the next section, focusing on the CMNIST dataset.

In addition to the previously mentioned hyperparameters, we use the Adam optimizer for training. For CMNIST and CelebA, we set the learning rate to 10^4 and the batch size to 256. For PACS and VLCS, we employ a commonly used hyperparameter tuning method: training domain validation.

E.4 Baseline Implementation

For CMNIST and CelebA datasets:

- *ERM:* We employ an architecture that combines our convolutional X-encoder and the classifier with two fully connected layers. As reported in prior works, the ERM with these architectures can achieve optimal in-distribution accuracy. Moreover, employing the same architecture of encoder and classifier provides a fair comparison between different methods.
- *IRM, F-IRM Game, V-IRM Game:* We employ the implementation of IRM and its two deviant methods from https://github.com/IBM/OoD. We employ the same architectures as ERM for fair comparison.
- *CTrans:* We employ the implementation of CTrans from its original paper (https://github.com/cvlab-columbia/CT4Recognition).
- *Causalrep:* We employ the implementation of Causalrep from its original paper (https://github.com/yixinwang/representation-causal-public).
- *iCaRL:* No implementation of iCaRL is provided, so we implement this method ourselves. Our implementation achieves results similar to those reported in the original paper. To ensure a fair comparison, we also include the reported results for CMNIST, PACS, and VLCS.

For the PACS and VLCS datasets, we run all the baselines using package domainBed(https://github.com/facebookresearch/DomainBed/tree/main).

F Ablation Studies

Table 1: The prediction performance in terms of accuracy (%) on CMNIST dataset with (1) $\boldsymbol{Z}$ with different identifiability; (2) number of latent variables; (3) various significance level α

Methods	# latent variables	α	# selected features	In-dist Acc	OOD Acc
NF-iVAE [7]	10	0.01	7/10	71.5	40.1
		0.015	5/10	64.9	53.6
		0.02	2/10	61.9	52.9
		0.05	1/10	60.8	45.2
NF-iVAE [7]	32	0.01	4/32	64.6	56.4
		0.005	11/32	74.1	71.4
		0.001	20/32	**76.4**	**74.1**
		0.0005	22/32	77.2	72.9
		$\boldsymbol{Z}_p$	12/32	71.8	68.3
VAE	10	0.01	1/10	51.2	46.5
		0.005	1/10	51.2	46.5
		0.001	9/10	84.1	15.6
		0.0005	9/10	84.1	15.6
VAE	32	0.01	2/32	55.2	48.9
		0.005	2/32	55.2	48.9
		0.001	28/32	78.4	22.9
		0.0005	28/32	78.4	22.9

F.1 Significance Level α for Mutual Information

We show the CMB sets identified by our proposed CMB discovery methods under distinct settings in Table 1. In practice, we adopt commonly used significant levels, setting α to be $0.0005, 0.001, 0.002, 0.005, 0.01$. We report the in-distribution and out-of-distribution prediction accuracy using the CMB sets identified by various significance levels. Observing Table 1, we note that a reasonable choice of significance level will result in the selected CMB set achieving optimal performance for both in-distribution and out-of-distribution predictions. A too-small significance level (e.g., 0.0005) may introduce irrelevant variables for in-distribution prediction and potentially spurious variables for out-of-distribution prediction. Conversely, a too-large significance level (e.g., 0.01) may lead to the selection of too few variables, preventing the inclusion of all relevant variables for accurate prediction. Empirically, we opt for CMB features that yield the highest

in-distribution accuracy and employ the selected features along with the trained classifier for out-of-distribution prediction.

F.2 Identifiability and Dimension of VAE Framework

In addition to adjusting significance levels, our investigation extends to exploring the influence of VAE identifiability on the CMB discovery. Especially, we train a vanilla VAE framework to obtain a set of variables with no identifiable guarantee. Then we apply our CMB discovery method on the unidentifiable set of $\boldsymbol{Z}$. As illustrated in Table 1, we cannot effectively separate CMB variables from spurious variables, even if we tune the significance levels exhaustively. As a result, the prediction performance greatly suffers using $\boldsymbol{Z}$ with no identifiable guarantee.

Moreover, we explore the impact on the selected CMB set when only executing Step 1 of the CMB search strategy. We train the iVAE with the prior $p(\boldsymbol{Z}|Y)$ and conduct conditional independence tests by setting the threshold to 0.001. Solely performing Step 1 results in the algorithm producing a CMB with 20 variables. Subsequently, by executing Step 2 with a backward search, we remove two variables from the CMB set. Interestingly, the CMB obtained solely from Step 1 yields a comparable (slightly inferior) accuracy, with an in-distribution accuracy of 78.75% and an out-of-distribution accuracy of 73.5%.When only Step 1 is performed, there is a risk of falsely including variables in the CMB set that do not contribute additional information to the target variable Y. This aligns with our expectation that the CMB set, without the backward search, contains the maximal mutual information of Y, resulting in minimal Bayes errors in predicting Y, yet may not be the minimal set. We also investigate the influence of the dimension of latent variable $\boldsymbol{Z}$, as shown in Table 1.

F.3 Comparision of Runtime

Results are shown in Table 2. While three-phase methods (CMBRL, iCaRL) take longer to train than end-to-end methods, CMBRL, in contrast to iCaRL, enhances invariant feature search in phase II and demands less time.

Table 2: The runtime (seconds) comparison between baselines on CMNIST. We choose $|\boldsymbol{Z}| = 10$. For CMBRL, we set $\alpha = 0.005$.

Methods	Training time				Inference time
	Total	Phase-I	Phase-II	Phase-III	
ERM	125				60
IRM	223				62
iCaRL	235	190	35	10	308
CMBRL	227	195	20	12	303

G Visualization of CMB representation

Fig. 3: The saliency map using fullGrad method on CelebA dataset.

To have a more comprehensive interpretation of why our CMBRL succeeds and the ERM fails, we utilize fullgrad [9], a widely studied saliency map approach, to visualize our learned set of $\boldsymbol{Z}_{cmb}$ and present the results in Figure 3. Importantly, it is essential to note that the visualized features are not used for prediction; their purpose is to showcase the selected features on the original input $\boldsymbol{X}$ from the training data. As depicted in Figure 3, ERM tends to focus on spurious features. For instance, in the data setting of predicting 'arched brows' with spurious attribute 'eye bag', ERM concentrates on the area of eye bags, while in the data setting of predicting 'black hair' with spurious attribute 'open mouth', it shifts attention to the area around the mouth. In contrast, the CMB features prioritize regions associated with muscle movement on the forehead, cheeks, and temples to predict the status of eyebrows. Simultaneously, these features focus on the head and hair area to predict hair color. This underscores the capacity of CMBRL to discern relevant features for prediction, particularly in scenarios where ERM falls short.

H Detailed Empirical Results on CelebA

Table 3: Empirical results on CelebA in terms of OOD prediction accuracy (%).

Target	Spurious	Methods				
		ERM	CTrans	Causalrep	iCaRL	CMBRL(ours)
Arched Brows	Eye Bags	57.4±0.6	62.6±1.8	53.9±4.2	64.3±5.6	**72.9**±6.5
Attractive	Wearing Necklace	58.3±1.0	59.4±0.8	53.7±2.0	65.9±3.7	**68.7**±3.9
Black Hair	Mouth Open	65.4±2.5	69.0±1.3	59.4±3.5	70.6±1.5	**75.1**±1.6
Goatee	Male	61.1±2.3	75.7±2.2	72.8±7.1	78.0±3.5	**81.9**±3.9
Mustache	Male	62.5±3.9	69.8±1.9	78.7±7.2	76.9±3.1	**82.9**±1.3
Arched Brows	Earrings	54.3±1.1	57.2±0.6	52.1±3.4	68.5±4.9	**74.6**±2.8
Mustache	Black Hair	66.1±2.3	70.3±0.4	52.5±3.2	76.6±4.3	**77.6**±2.0
	Avg	60.7	66.3	60.4	71.5	**76.2**

References

1. Ahuja, K., Shanmugam, K., Varshney, K., Dhurandhar, A.: Invariant risk minimization game. In: International Conference on Machine Learning (2020) 9
2. Hyvarinen, A., Sasaki, H., Turner, R.: Nonlinear ica using auxiliary variables and generalized contrastive learning. In: The 22nd International Conference on Artificial Intelligence and Statistics. pp. 859–868. PMLR (2019) 2
3. Khemakhem, I., Kingma, D., Monti, R., Hyvarinen, A.: Variational autoencoders and nonlinear ica: A unifying framework. In: International Conference on Artificial Intelligence and Statistics. pp. 2207–2217. PMLR (2020) 2, 5
4. Khemakhem, I., Monti, R., Kingma, D., Hyvarinen, A.: Ice-beem: Identifiable conditional energy-based deep models based on nonlinear ica. Advances in Neural Information Processing Systems **33**, 12768–12778 (2020) 2
5. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: Proceedings of the IEEE international conference on computer vision. pp. 5542–5550 (2017) 9
6. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE international conference on computer vision. pp. 3730–3738 (2015) 9
7. Lu, C., Wu, Y., Hernández-Lobato, J.M., Schölkopf, B.: Invariant causal representation learning for out-of-distribution generalization. In: International Conference on Learning Representations (2021) 1, 2, 4, 5, 9, 13
8. Peters, J., Buhlmann, P., Meinshausen, N.: Causal inference using invariant prediction: identification and confidence intervals. arxiv. Methodology (2015) 1
9. Srinivas, S., Fleuret, F.: Full-gradient representation for neural network visualization. Advances in neural information processing systems **32** (2019) 15
10. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR 2011. pp. 1521–1528. IEEE (2011) 9
11. Tsamardinos, I., Aliferis, C.F., Statnikov, A.R., Statnikov, E.: Algorithms for large scale markov blanket discovery. In: FLAIRS conference. vol. 2, pp. 376–380. St. Augustine, FL (2003) 7
12. Wang, Y., Jordan, M.I.: Desiderata for representation learning: A causal perspective. arXiv preprint arXiv:2109.03795 (2021) 9