## Appendix A. Appendix for Method

### A.1. Details on the Model Components

We will introduce details on learning components of the model, especially on the pair of posterior $q(\mathbf{z} \mid \mathbf{d})$ and prior $p(\mathbf{z}) \approx p(\mathbf{z} \mid \mathbf{d})$, and of posterior $q(\mathbf{c}, e_{\mathbf{c}} \mid \mathbf{d})$ and prior $p(\mathbf{c}, e_{\mathbf{c}})$. Then, we will introduce the derivation of our ELBO objective. Finally, model components regarding prediction loss and mutual information regularization will be elaborated.

**Modeling regarding Z.** we employ different architectures for each encoder, considering the data generating process of $\mathbf{Z}$ and $\mathbf{C}$ in realistic settings where non-IVs are more prevalent and exhibit a more complex structure. We approximate distribution of latent variable $\mathbf{Z}$ through VAE-based framework (Kingma and Welling, 2013; Sohn et al., 2015) considering association between $\mathbf{Z}$ and $\mathbf{D}_1$. We model a prior distribution of $\mathbf{Z}$ as $p(\mathbf{z} \mid \mathbf{d})$; for implementation, we chose a normal distribution as a prior following practice (Sohn et al., 2015; Lopez-Martin et al., 2017):

$$p(\mathbf{z} \mid \mathbf{d}) = N(\mu_{\mathbf{d}}, \sigma^2_{\mathbf{d}} I).$$

We use $p(\mathbf{z}) = \sum_{\mathbf{d}} p(\mathbf{z}|\mathbf{d})p(\mathbf{d}) \approx \frac{1}{N}\sum_{i=1}^{N} p(\mathbf{z} \mid \mathbf{d}^{(i)})$ where $\mathbf{d}^{(i)} \sim p(\mathbf{d})$. We chose $N = 1$ and intend to learn the prior $p(\mathbf{z}) \approx p(\mathbf{z} \mid \mathbf{d})$.

Then, encoder $q(\mathbf{z} \mid \mathbf{d})$ defined as a variational posterior distribution is trained to be close to $p(\mathbf{z} \mid \mathbf{d})$.

$$q(\mathbf{z} \mid \mathbf{d}) \approx N(\mu_{\mathbf{z}|\mathbf{d}}, \sigma^2_{\mathbf{z}|\mathbf{d}} I).$$

**Modeling regarding C.** we aim to encode $\mathbf{C}$ that can capture likely complex structure of non-IV $\mathbf{D}_2$ which usually have a number more than that of true IVs in observed covariates in real-world scenarios. By accommodating the mixture model prior, we aim to learn $\mathbf{C}$ expressing complex latent structures of the observed data. Thus, we utilize Variational Deep Embedding (VaDE) (Jiang et al., 2017), which assumes prior $p(\mathbf{c})$ following a Gaussian Mixture Model. We derive both $\mathbf{c}$ and its component $e_{\mathbf{c}}$ as

$$p(e_{\mathbf{c}}) \sim \text{Cat}(\boldsymbol{\pi})$$
$$p(\mathbf{c} \mid e_{\mathbf{c}}) \sim N(\mu_{e_{\mathbf{c}}}, \sigma^2_{e_{\mathbf{c}}} I),$$

where $\boldsymbol{\pi} = (\pi_1, ..., \pi_K) \in \mathbb{R}^K$. The Loss term related to ELBO in Eq. 3 implies that $q(\mathbf{c}, e_{\mathbf{c}} \mid \mathbf{d})$ is trained to be close to its prior $p(\mathbf{c}, e_{\mathbf{c}})$. As VaDE originally assumes, we model $q(\mathbf{c}, e_{\mathbf{c}} \mid \mathbf{d})$ to be a meanfield distribution that can be factorized into $q(\mathbf{c}, e_{\mathbf{c}} \mid \mathbf{d}) = q(\mathbf{c} \mid \mathbf{d})q(e_{\mathbf{c}} \mid \mathbf{d})$. Then, we will model $q(\mathbf{c} \mid \mathbf{d})$ as,

$$q(\mathbf{c} \mid \mathbf{d}) = N(\mu_{\mathbf{c}|\mathbf{d}}, \sigma^2_{\mathbf{c}|\mathbf{d}} I).$$

For $q(e_\mathbf{c} \mid \mathbf{d})$, we compute it as $p(e_\mathbf{c} \mid \mathbf{c})$ following the same logic in Jiang et al. (2017). This is because, as shown in the following rewritten forms,

$$
\begin{aligned}
&\mathcal{L}_{\text{ELBO}} \\
&= -\mathbb{E}_{q(\mathbf{z},\mathbf{c},e_\mathbf{c}|\mathbf{d})}\left[\log \frac{p(\mathbf{d}_1,\mathbf{d}_2,\mathbf{z},\mathbf{c},e_\mathbf{c})}{q(\mathbf{z},\mathbf{c},e_\mathbf{c} \mid \mathbf{d})}\right] \\
&= -\int_{\mathbf{z},\mathbf{c}} \sum_{e_\mathbf{c}} q(\mathbf{z} \mid \mathbf{d}) q(\mathbf{c} \mid \mathbf{d}) q(e_\mathbf{c} \mid \mathbf{d}) \\
&\qquad \cdot \left[\log \frac{p(\mathbf{d}_1,\mathbf{z}) p(\mathbf{d}_2 \mid \mathbf{c}) p(\mathbf{c})}{q(\mathbf{z} \mid \mathbf{d}) q(\mathbf{c} \mid \mathbf{d})} + \log \frac{p(e_\mathbf{c} \mid \mathbf{c})}{q(e_\mathbf{c} \mid \mathbf{d})}\right] \mathrm{d}\mathbf{z}\, \mathrm{d}\mathbf{c} \\
&= -\int_{\mathbf{z},\mathbf{c}} q(\mathbf{z} \mid \mathbf{d}) q(\mathbf{c} \mid \mathbf{d}) \log \frac{p(\mathbf{d}_1,\mathbf{z}) p(\mathbf{d}_2 \mid \mathbf{c}) p(\mathbf{c})}{q(\mathbf{z} \mid \mathbf{d}) q(\mathbf{c} \mid \mathbf{d})} \mathrm{d}\mathbf{z}\mathrm{d}\mathbf{c} \\
&\qquad + \int_{\mathbf{z},\mathbf{c}} q(\mathbf{z} \mid \mathbf{d}) q(\mathbf{c} \mid \mathbf{d}) D_{\text{KL}}(q(e_\mathbf{c} \mid \mathbf{d}) \parallel p(e_\mathbf{c} \mid \mathbf{c})) \mathrm{d}\mathbf{z}\, \mathrm{d}\mathbf{c}
\end{aligned}
$$

to optimize $\mathcal{L}_{\text{ELBO}}$, $D_{\text{KL}}(q(e_\mathbf{c} \mid \mathbf{d}) \parallel p(e_\mathbf{c} \mid \mathbf{c}))$ should be 0.

**Modeling of the decoder $p(\mathbf{d}|\mathbf{z},\mathbf{c})$.** A shared decoder $p(\mathbf{d}|\mathbf{z},\mathbf{c})$ is trained to have $\widehat{\mathbf{Z}} \sim q(\mathbf{z}|\mathbf{d})$ and $\widehat{\mathbf{C}} \sim q(\mathbf{c}|\mathbf{d})$ obtaining enough dependence with $\mathbf{D}$. As we have two distinct architectures for the encoders, the reconstruction term $\mathbb{E}_q[p(\mathbf{d}|\mathbf{z},\mathbf{c})]$ can be collapsed in the early stage of the training. Thus, we pretrain the shared decoder and encoders $q(\mathbf{z}|\mathbf{d})$ and $q(\mathbf{c}|\mathbf{d})$ with 30 epochs.

**Details on ELBO of $\mathbf{D}$.** We can derive the negative ELBO of $\log p(\mathbf{d})$ with the components above.

$$
\begin{aligned}
-\log p(\mathbf{d}) &\leq -\mathbb{E}_{q(\mathbf{z},\mathbf{c},e_\mathbf{c}|\mathbf{d})}\left[\log \frac{p(\mathbf{d},\mathbf{z},\mathbf{c},e_\mathbf{c})}{q(\mathbf{z},\mathbf{c},e_\mathbf{c} \mid \mathbf{d})}\right] \\
&= -\mathbb{E}_{q(\mathbf{z},\mathbf{c},e_\mathbf{c}|\mathbf{d})}\left[\log \frac{p(\mathbf{d} \mid \mathbf{z},\mathbf{c}) p(\mathbf{z}) p(\mathbf{c} \mid e_\mathbf{c}) p(e_\mathbf{c})}{q(\mathbf{z},\mathbf{c},e_\mathbf{c} \mid \mathbf{d})}\right] \\
&= -\mathbb{E}_q\left[\log p(\mathbf{d} \mid \mathbf{z},\mathbf{c},e_\mathbf{c})\right] \\
&\qquad + \int_\mathbf{c} \sum_{e_\mathbf{c}} q(\mathbf{c},e_\mathbf{c} \mid \mathbf{d}) D_{\text{KL}}(q(\mathbf{z} \mid \mathbf{d}) \parallel p(\mathbf{z})) \mathrm{d}\mathbf{c} \\
&\qquad + \int_\mathbf{z} q(\mathbf{z} \mid \mathbf{d}) D_{\text{KL}}(q(\mathbf{c},e_\mathbf{c} \mid \mathbf{d}) \parallel p(\mathbf{c},e_\mathbf{c})) \mathrm{d}\mathbf{z} \\
&\approx -\mathbb{E}_q\left[\log p(\mathbf{d} \mid \mathbf{z},\mathbf{c})\right] \\
&\qquad + \int_\mathbf{c} \sum_{e_\mathbf{c}} q(\mathbf{c},e_\mathbf{c} \mid \mathbf{d}) D_{\text{KL}}(q(\mathbf{z} \mid \mathbf{d}) \parallel p(\mathbf{z} \mid \mathbf{d})) \mathrm{d}\mathbf{c} \\
&\qquad + \int_\mathbf{z} q(\mathbf{z} \mid \mathbf{d}) D_{\text{KL}}(q(\mathbf{c},e_\mathbf{c} \mid \mathbf{d}) \parallel p(\mathbf{c},e_\mathbf{c})) \mathrm{d}\mathbf{z} \\
&= \widehat{\mathcal{L}_{\text{ELBO}}}.
\end{aligned}
$$

As we approximate $p(\mathbf{z})$ with $p(\mathbf{z} \mid \mathbf{d})$, the derived term incorporates the components of the model $p(\mathbf{d} \mid \mathbf{z},\mathbf{c}), p(\mathbf{z} \mid \mathbf{d}), p(\mathbf{c},e_c), q(\mathbf{z} \mid \mathbf{d}), q(\mathbf{c},e_c \mid \mathbf{d})$.

**Modeling $p(x \mid \mathbf{z})$.** We parameterize it to approximate $p(x \mid \mathbf{z})$ assumed to follow distribution in regular exponential family.

$$p_\theta(x \mid \mathbf{z}) = \begin{cases} \text{Bern}(\text{sigmoid}(\theta_1(\mathbf{z}))) & \text{for binary } x \\ N(\mu_{\theta_2(\mathbf{z})}, \sigma^2_{\theta_2(\mathbf{z})}). & \text{for continuous } x \end{cases}$$

One can further model it with more complex exponential family, but for simplicity we select the two distributions each of which covers either binary or continuous variable.

**Tractable MI regularization.** Since we model both $q(\mathbf{z} \mid \mathbf{d})$ and $q(\mathbf{c} \mid \mathbf{d})$ as normal distributions, we can obtain tractable MI as follows (Gelfand and Yaglom, 1959):

$$I(Z_j; C_k|\mathbf{D}) = -\frac{1}{2}\log(1 - Q^2_{z_j,c_k}),$$

for all $z_j \in \mathbf{Z}$ and $c_k \in \mathbf{C}$, where $Q_{z_j,c_k} = \text{corr}(z_j, c_k)$ can be replaced with the sample correlation $\hat{Q}_{z_j,c_k}$ from mini- batch. Thus, we can formulate the training objective in terms of MI as follows:

$$\mathcal{L}_3 = -\frac{1}{2}\sum_{j,k}\log(1 - \hat{Q}^2_{z_j,c_k}). \tag{7}$$

Instead, we use a simplified version of the formula, $\mathcal{L}_{3,\text{simple}} = \sum_{j,k}\hat{Q}^2_{z_j,c_k}$, which has the same global optimum when $\hat{Q}_{z_j,c_k} = 0$ for all $z_j$ and $c_k$, but does not explode.

## A.2. Discussion on Model Identification

Here, we provide conditions for the model identifiability and how we considered them in our architectural design:

- When the prior distribution of latent variable $\mathbf{Z}, \mathbf{C}$ follows a Gaussian Mixture Model (possibly, a number of components can be one or more);
- When $\mathbf{D}_1, \mathbf{D}_2$ are fed into respectively learned encoders; and
- When the function that maps latent variables to observed variables is injective;

It is known that the model is identifiable up to affine transformations if the above conditions are satisfied (Kivva et al., 2022). As such conditions are often too strict and infeasible, prior works (Zhang et al., 2021; Cheng et al., 2023b,a) also lack the guarantee of the model identifiability. In our work, we employed ReLU activations (Stock and Gribonval, 2023) and MI constraints to impose them in practice where its effectiveness is demonstrated in Sec 5.1.3.

## A.3. Causal Effect Identification

Here, we clarify how the causal effect of $X$ on $Y$ is identified under the relaxed assumption allowing dependence between covariates $\mathbf{D}$ and unobserved confounders $U$.

Prior works (i.e., UAS, WAS, Ivy) established the identifiability of the causal effects (e.g., ATE or APCE) under restrictive assumptions (Table 1). Therefore, it is crucial to validate its soundness under the weaker assumptions that we consider. To prove it, we suppose the conditions mentioned in Remark 2 as following:

1. (*Separability*)

$$Y = g_Y(X, \mathbf{C}, \mathbf{D}_2) = g_{Y_1}(X) + g_{Y_2}(\mathbf{C}, \mathbf{D}_2) + U$$

2. (*Completeness*) A set of distributions $X|\mathbf{Z}$ is complete

**Proposition 4** *Under the conditions 1,2, if* $\mathbf{V} = \{X, Y, \mathbf{D}, \mathbf{Z}, \mathbf{C}, \mathbf{U}\}$ *is generated based on dependency graph in Fig. 2(a), then ATE/APCE is identifiable.*

**Proof** Our proof mostly follows a proof from Newey and Powell (2003). From condition 1, the following equation can be formulated.

$$\mathbb{E}[y \mid \mathbf{z}] = \int g_{Y_1}(x)p(x \mid \mathbf{z})\mathrm{d}x + \mathbb{E}[g_{Y_2}(\mathbf{c}, \mathbf{d}_2)|\mathbf{z}] + \mathbb{E}[U|\mathbf{z}] \tag{8}$$

Identification of ATE/APCE relies on identification of $g_{Y_1}(x)$. Identification of $g_{Y_1}(x)$ depends on the existence of a unique solution to be Eq. 8. To prove it, let $g'_{Y_1}$ be one of the solutions of Eq. 8 Then, $g_{Y_1}$ and $g'_{Y_1}$ both solve Eq. 8 and by subtracting Eq. 8 from the same equation replaced with another solution $g'_{Y_1}$ we can elicit the following;

$$\begin{aligned}
0 &= \mathbb{E}[y \mid \mathbf{z}] - \mathbb{E}[y \mid \mathbf{z}] \\
&= \left(\int g'_{Y_1}(x)p(x \mid \mathbf{z})\mathrm{d}x + \mathbb{E}[g_{Y_2}(\mathbf{c}, \mathbf{d}_2)|\mathbf{z}] + \mathbb{E}[U|\mathbf{z}]\right) \\
&- \left(\int g_{Y_1}(x)p(x \mid \mathbf{z})\mathrm{d}x + \mathbb{E}[g_{Y_2}(\mathbf{c}, \mathbf{d}_2)|\mathbf{z}] + \mathbb{E}[U|\mathbf{z}]\right) \\
&= \int [g'_{Y_1}(x) - g_{Y_1}(x)]p(x \mid \mathbf{z})\mathrm{d}x.
\end{aligned}$$

The uniqueness of the solution is equivalent to the non-existence of any function $u(x) = g_{Y_1}(x) - g'_{Y_1}(x) \neq 0$ such that $\mathbb{E}[u(x) \mid \mathbf{z}] = 0$. It is equivalent to the completeness of a set of $X \mid \mathbf{Z}$ derived by Condition 2. Thus, Eq. 8 has a unique solution of $g_{Y_1}(x)$ and ATE/APCE can be identified.

■

For reference, the proposition (for continuous treatment) can also be proved following the work of Wong Wong (2022) with additional technical assumptions. The mentioned conditions here align with the standard assumptions used in IV analysis. The articulation of these conditions provides the following insights: first, to guide researchers on the application of our model in practical IV analysis scenarios; second, to emphasize that these are general conditions often assumed or accepted in typical settings, thus underscoring the broad applicability of our model. This framework not only enhances the utility of the model but also supports its generalization across different research contexts.

## Appendix B. Appendix for Experiment

### B.1. Dataset Details

**Synthetic Datasets** We create synthetic datasets with sample sizes of 5000. In order to make the setting more realistic, in both low-dimensional and high-dimensional cases, approximately the half of the variables in $\mathbf{D}_2$ are correlated with an unobserved confounder denoted as $U$. Additionally, the number of instrumental variables (IVs) in $\mathbf{D}$ is less than the half of all the observed covariates.

Datasets are generated based on the following process. The dimensions of $\mathbf{D}_1, \mathbf{Z}, \mathbf{D}_2, \mathbf{C}$ are denoted as $m, k, l, o$, respectively. Distributions of exogenous variables and variables without parents is $\epsilon_{\mathbf{D}_1} \sim N(0, 10I_m)$, $\epsilon_{\mathbf{Z}} \sim N(0, 0.01I_k)$, $\epsilon_{\mathbf{D}_2} \sim N(0, 0.25I_l)$, $U \sim N(0, 1)$, and $\mathbf{C} \sim N(0, I_o)$. The rest are determined as

$$\begin{aligned}
&\mathbf{D}_1 = f_{\mathbf{D}_1}(\epsilon_{\mathbf{D}_1}), \mathbf{Z} = f_{\mathbf{Z}}(\mathbf{D}_1) + \epsilon_{\mathbf{Z}}, \\
&\mathbf{D}_2 = A^\top \mathbf{C} + \epsilon_{\mathbf{D}_2} + U \cdot 1(\mathbf{p} > 0.5), \\
&\qquad\quad \text{where, } p_i \sim \text{Bern}(0.5),\ i = 1, 2, \ldots, l, \\
&\quad X = \begin{cases} 1\left[\dfrac{1}{1 + \exp(f_X(\mathbf{Z}) + B^\top[\mathbf{C} : \mathbf{D}_2] + U)} > 0.5\right] & \text{(binary)} \\ f_X(\mathbf{Z}) + B^\top[\mathbf{C} : \mathbf{D}_2] + U & \text{(continuous)} \end{cases} \\
&\quad Y = g(X) + f_Y(\mathbf{C}, \mathbf{D}_2) + U,
\end{aligned}$$

where $[\mathbf{C} : \mathbf{D}_2]$ is the concatenation of $\mathbf{C}$ and $\mathbf{D}_2$.

We apply an affine transformation of $\mathbf{C}$ and $\mathbf{D}_2$ by using coefficient matrices $A$ and $B$ sampled from a normal distribution, respectively. Other arbitrary functional relationships between variables are introduced by sampling $f_{(.)} \sim \mathcal{GP}(0, k_{\text{RBF}})$, i.e., a Gaussian Process prior with zero mean function and RBF kernel with its length scale set to 1. The function sampler is intended to incorporate the complexity of real-world scenarios.

We generate $X$ and $Y$ using the same process in the low-dimensional and high-dimensional case. For the response function relating $X$ to $Y$, denoted as $g(X)$, we investigate two specific cases: a linear relationship, where $g(X) = 3X$, and a non-linear relationship, where $g(X) = \exp(0.5X)$. In other words, in the linear case, the true ATE is 3, while in the non-linear case, the true APCE is given by $\partial_x \mathbb{E}[Y(X = x)] = 0.5\exp(0.5x)$.

In low-dimensional scenarios, each dimension of the variable, i.e., $m$, $k$, $l$, $o$ is set to 2, 2, 4, 3, and in high-dimensional scenarios, we utilize the MNIST dataset (LeCun et al., 1998). Images in the dataset are represented by $\mathbf{D} \in \mathbb{R}^{28\times28=784}$, i.e. $m + l = 784$. Due to the lack of specific information about which variables (pixels) are $\mathbf{D}_1$ or $\mathbf{D}_2$, we implicitly assume that $\mathbf{D}_1$ is a set of pixels corresponding to digit. However, in this case, $\mathbf{D}_1$ can be changed by each image, which is more challenging. In terms of assumption about distinguishable $\mathbf{D}$, we can consider it as an even more general setting that should learn the representation of $\mathbf{Z}$ from indistinguishable $\mathbf{D}$. Additionally, we set $\mathbf{Z}$ as the label of each image, thus establishing an intuitive relationship between $\mathbf{D}_1$ and $\mathbf{Z}$. To create distinctive values for $\mathbf{D}_2$, the first 100 pixels are selected and noise $U$ is injected, which can serve as a confounding variable for digit recognition.

For each dataset, we conduct 20 independent replications with the training and test datasets, divided in a 7:3 ratio.

**Relationships between Participation in Tax-deferred Programs (Abadie, 2003).** The dataset is composed of 11 variables with 9,275 units. Except for the treatment (participation in 401(k)) and the outcome (participation in IRA), income, net family financial assets, family size, age, gender etc. were collected. Among 11 variables, we use 6 numerical variables that are suitable for averaging methods such as UAS and WAS. As the outcome variable is binary, we extend our model to binary outcome by introducing binary cross entropy to model $f(y \mid \hat{x}, \mathbf{c})$. The dataset is available in R package.[2]

**Police Force Size and Civilization Race (Chalfin et al., 2022).** The dataset is composed of 195 columns with 1037 units. Among 195 columns, regardless of methods, we use city id (with one-hot encoding), interacted state by year, power of population, detailed data for city's race, gender and age composition, median household income, poverty rate, and unemployment rate except for the treatment (number of sworn police officers) and outcome (Quality of life arrests for white). We also included IV (variation in the timing of federal block grants provided by the US Department of Justice's Community Oriented Policing Services (COPS) office) in those covariates. The dataset is available on replication package.[3]

## B.2. Baselines

**UAS.** Unweighted Allele Score (UAS) (Burgess and Thompson, 2013) was devised for Mendelian randomization (Katan, 1986). The model calculates the average of each covariate per unit and uses the value as an IV. The model assumes that all averaged covariates satisfy the conditions of IVs, but they are weak. We implement UAS with basic functions of average provided by *python* packages *pytorch* and *numpy* by following the instructions in the original papers.

**WAS.** Weighted Allele Score (WAS) (Burgess et al., 2016) takes a similar approach with UAS, but it calculates a weighted average of values of each covariate. The weights are calculated from the correlation between the covariates and treatment. WAS also assumes that all the covariates satisfy the conditions of IVs. We implement WAS in a similar way to UAS.

**DVAE.CIV** DVAE.CIV (Cheng et al., 2023a) learns a representation of conditional IV and its conditioning set. Conditional IV requires conditioning set to function as IV. DVAE.CIV learns the two representations by leveraging disentangling Variational Autoencoder. For implementation, we use the code provided by the authors.[4]

## B.3. Estimators

For IV-based estimators, 2SLS and IVGMM (Hansen, 1982) are implemented with python package *linearmodels*. We use the implementation of DML (Chernozhukov et al., 2018), Ortho (Syrgkanis et al., 2019) in python package *econml*. As Ortho and DML do not provide a prediction method, we use the value of response function when the treatment value is 1 and 0. Default parameters are used for each class. For Poly2SLS and KernelIV (Singh et al., 2019), we used open sourced implementation.

## B.4. Implementation

Our method have VaDE architecture to learn complex distributions. We used open-sourced implementation of VaDE.

2. https://cran.r-project.org/web/packages/wooldridge/wooldridge.pdf
3. https://www.aeaweb.org/articles?id=10.1257/aeri.20200792 and https://www.openicpsr.org/openicpsr/project/135761/version/V1/view.
4. As DVAE.CIV is an advanced version of CIV.VAE, we omitted experiments on CIV.VAE.

| | | Linear response function | | | | | | Non-linear response function | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Method | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV |
| Binary | Ours | 0.26 (0.09) | 0.31 (0.07) | **1.59** (1.17) | **1.3** (1.25) | **1.66** (3.24) | **0.38** (0.31) | **1.12** (0.17) | **1.06** (0.2) | **0.4** (0.56) | **0.37** (0.47) | **0.47** (0.18) | **0.33** (0.02) |
| | w/o pred | 0.26 (0.08) | 0.32 (0.08) | 3.33 (1.8) | 3.42 (1.47) | 2.82 (2.22) | 0.42 (0.44) | 1.46 (0.11) | 1.37 (0.1) | 3.78 (1.87) | 3.71 (1.52) | 34.41 (143.74) | 0.5 (0.23) |
| | w/o aux | **0.23** (0.06) | **0.29** (0.06) | 5.67 (6.8) | 3.13 (1.71) | 2.17 (1.2) | **0.38** (0.4) | 1.49 (0.1) | 1.41 (0.1) | 4.97 (5.25) | 3.13 (1.71) | 2.12 (1.24) | 0.68 (0.3) |
| Continuous | Ours | **0.07** (0.09) | **0.08** (0.09) | **0.03** (0.03) | **0.02** (0.01) | **0.35** (0.07) | **3.51** (0.55) | **1.38** (0.1) | **0.3** (0.15) | **1.35** (0.12) | **1.37** (0.1) | **4.13** (0.32) | **3.82** (1.16) |
| | w/o pred | 1.42 (1.69) | 1.48 (1.69) | 0.69 (0.72) | 0.35 (0.48) | 2.37 (2.75) | 4.78 (1.63) | 1.42 (0.55) | 1.31 (1.99) | 1.75 (0.75) | 1.54 (0.6) | 4.95 (0.77) | 6.6 (1.29) |
| | w/o aux | 1.4 (1.17) | 1.47 (1.14) | 0.92 (1.21) | 0.38 (0.24) | 1.96 (2.02) | 4.96 (1.51) | 1.72 (1.67) | 2.12 (2.79) | 2.52 (2.68) | 1.43 (0.77) | 5.19 (1.01) | 6.4 (1.59) |

Table 6: Experiment results on ablation study.

| Parameters | Low-dimensional | | High-dimensional | |
|---|---|---|---|---|
| | binary | continuous | binary | continuous |
| Dim of $\mathbf{Z}$ | 3 | 3 | 25 | 25 |
| Dim of $\mathbf{C}$ | 2 | 2 | 25 | 25 |
| Number of components | 4 | 4 | 4 | 4 |
| Hidden dim | 200 | 200 | 150 or 200 | 150 or 200 |
| Hidden layers | 1 or 3 | 1 or 3 | 1 or 3 | 1 or 3 |
| Activation Function | (ReLU, Sigmoid) | (ReLU, Sigmoid) | (ReLU, Sigmoid) | (ReLU, Sigmoid) |

Table 7: Hyperparameters of the model.

As VaDE method did, we also pretrain our encoder and decoder to avoid the problem that the reconstruction term would be weak (Jiang et al., 2017). Additionally, we select Adam optimizer and apply early stopping for efficient learning. All layers are fully connected, including encoders, decoder, and auxiliary nets. Hyperparameter details are on the Table  7 and Table 8 according to the experiment settings with dimensions and type of treatment. In the Table 7, ”or” in ”Number of components” statement means that we applied both values with linear or non-linear case respectively. ”or” in ”Hidden layers” means that we chose either one or three layers when constructing networks. On the other hand, in the Table 8, early stopping epochs are applied with 3 when training most of low-dimensional cases. High dimensional case is trained with early stopping epochs 10. We tuned the hyperparameters with grid search within a specific range. The dimensions of latent variable $\mathbf{Z}$ are selected from $[1, 2, 3]$ for low-dimensional setting and $[10, 20, 25]$ for high-dimensional setting. The dimensions of $\mathbf{C}$ are selected from the same range, and the number of its component is selected from $[2, 3, 4]$. Experiments were executed on NVIDIA RTX 6000 with 48GB memory.

## Appendix C. Additional Experimental Results

Full results of the ablation study on various estimators are reported in Table 6. The results indicate the importance of each component in our model. (1) The lack of auxiliary regression networks leads to a significant decline in performance. (2) The dual prediction nets take an important role for the inference of unconfounded representation from the perspective of performance decline. We also provide the results of experiments on various latent model architectures for representation $\mathbf{Z}$ and $\mathbf{C}$, which is elaborated in the last part of the section.

**Experiments on assumption violation.** We assess how robust our model is to dependence among observed covariates $\mathbf{D}$. This is a violation of assumption $\mathbf{D}_1 \perp\!\!\!\perp \mathbf{D}_2$ in Sec. 3. In this case, as

| | | | |
|---|---|---|---|
| Prediction loss weight $\alpha$ | 5 | Weight decay | 0.0001 |
| MI loss weight $\beta$ | 5 | Optimizer | Adam |
| Early stopping epochs | 3 or 10 | Batch size | 512 |
| Training epochs | 128 | Learning Rate | 0.001 |
| LR Decay | 0.01 | LR Scheduler | StepLR |

Table 8: Hyperparameters for training.

| | | Linear response function | | | | | | Non-linear response function | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Method | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV |
| Binary | UAS | 1.23 | 1.23 | 1.78 | 1.35 | 0.69 | **0.3** | 1.23 | 1.23 | 1.57 | 1.35 | 0.64 | 0.48 |
| | WAS | 0.91 | 0.91 | 1.4 | **1.25** | 0.65 | 0.39 | 1.15 | 1.15 | 1.34 | 1.25 | 0.6 | 0.51 |
| | DVAE.CIV | 1.5 (0.09) | 1.5 (0.09) | **1.31** (1.18) | 1.37 (1.18) | 0.98 (0.15) | 1.11 (0.05) | **1.12** (1.65) | **1.12** (1.65) | **0.6** (0.47) | 1.32 (0.78) | 0.88 (0.05) | 0.88 (0.01) |
| | CoCoIV (Ours) | **0.15** (0.14) | **0.16** (0.15) | 1.46 (0.48) | 1.31 (0.5) | **0.59** (0.23) | 0.35 (0.17) | 1.46 (0.35) | 1.4 (0.34) | 0.8 (0.95) | **0.81** (0.9) | **0.42** (0.39) | **0.32** (0.19) |
| Continuous | UAS | 0.47 | 0.47 | 0.12 | 0.14 | 0.43 | 1.7 | 4.39 | 4.39 | 3.79 | **3.78** | **9.69** | **6.23** |
| | WAS | 0.44 | 0.44 | 0.11 | 0.13 | 0.41 | **1.11** | 4.18 | 4.18 | 3.79 | 3.79 | 9.86 | 7.62 |
| | AutoIV | 0.51 (0.65) | 0.51 (0.65) | 0.14 (0.27) | 0.14 (0.27) | 0.66 (0.81) | 2.3 (0.86) | 18.41 (34.71) | 18.41 (34.71) | N/A (N/A) | 5.25 (5.61) | 24.23 (55.97) | 15.71 (14.5) |
| | CoCoIV (Ours) | **0.13** (0.04) | **0.13** (0.03) | **0.03** (0.01) | **0.04** (0.01) | **0.24** (0.01) | 2.1 (0.33) | **3.85** (0.39) | **1.22** (0.59) | **3.77** (0.41) | 3.85 (0.43) | 10.66 (0.3) | 19.49 (12.01) |

Table 9: Experiment results on low-dimensional synthetic datasets Interdependency.

| | | Linear response function | | | | | | Non-linear response function | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Method | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV |
| Binary | UAS | 3.73 | 3.73 | 5.53 | 3.64 | 1.75 | 0.58 | 2.37 | 2.37 | 5.0 | 3.64 | 1.77 | **0.45** |
| | WAS | 0.52 | 0.52 | 2.29 | **1.39** | **0.65** | **0.43** | 1.35 | 1.35 | 1.8 | 1.39 | **0.67** | **0.45** |
| | DVAE.CIV | 1.52 (0.23) | 1.52 (0.23) | 2.78 (2.27) | 3.17 (6.92) | 1.14 (0.2) | 1.27 (0.03) | **0.76** (0.5) | **0.76** (0.5) | 1.49 (1.3) | 2.48 (3.32) | 0.93 (0.1) | 0.94 (0.02) |
| | CoCoIV(Ours) | **0.33** (0.1) | **0.34** (0.12) | **2.02** (1.07) | 2.01 (1.05) | 1.0 (0.52) | 0.57 (0.32) | 1.19 (0.25) | 1.17 (0.29) | **1.37** (1.17) | **1.37** (1.2) | 1.02 (0.7) | 0.48 (0.1) |
| Continuous | UAS | 1.52 | 1.52 | 0.46 | 0.42 | 1.24 | 6.79 | 2.76 | 2.76 | **0.83** | **0.88** | **5.97** | 9.19 |
| | WAS | 0.38 | 0.38 | 0.17 | 0.15 | 0.7 | 3.18 | **1.64** | 1.64 | 1.81 | 1.84 | 8.11 | **8.74** |
| | AutoIV | 2.25 (2.68) | 2.25 (2.68) | 0.27 (0.24) | 0.24 (0.23) | 0.89 (0.68) | 3.06 (1.58) | 32.79 (52.83) | 32.79 (52.83) | 2.64 (1.85) | 2.65 (1.61) | 6.96 (0.44) | 10.52 (2.06) |
| | CoCoIV(Ours) | **0.27** (0.16) | **0.29** (0.16) | **0.11** (0.09) | **0.1** (0.05) | **0.48** (0.16) | **1.87** (0.83) | 2.43 (0.53) | **0.64** (0.53) | 2.32 (0.65) | 2.28 (0.53) | 7.49 (1.13) | 10.77 (2.33) |

Table 10: Experiment results on low-dimensional synthetic datasets within dependency.

mentioned in Sec. 5.2, learned representation $Z$ may not be a valid IV because the variables in $\mathbf{D}_1$ are not IVs anymore. However, in real-world settings, IVs are seldom perfectly unconfounded but are more often in a corrupted state, influenced by other variables. The main goal of our experiment is thus to determine whether our model could robustly estimate the causal effect even within such compromised circumstances.

We execute experiments on the datasets assuming the existence of unobserved common causes between $\mathbf{D}_1$ and $\mathbf{D}_2$. The generating process is similar with that in Sec. B.1 except for the presence of $U_{\mathbf{D}_1,\mathbf{D}_2}$:

$$\begin{aligned}
&U_{\mathbf{D}_1,\mathbf{D}_2} \sim N(0,1),\\
&\mathbf{D}_1 = f_{\mathbf{D}_1}(\epsilon_{\mathbf{D}_1}) + U_{\mathbf{D}_1,\mathbf{D}_2},\\
&\mathbf{D}_2 = A^\top \mathbf{C} + \epsilon_{\mathbf{D}_2} + U \cdot 1(\mathbf{p} > 0.5) + U_{\mathbf{D}_1,\mathbf{D}_2}.
\end{aligned}$$

Table 9[5] depicts the results on the modified datasets. Despite the violation of assumption $\mathbf{D}_1 \perp\!\!\!\perp \mathbf{D}_2$, our model records the lowest MAEs on half of the estimators with binary treatment. In addition, the decrease in performance with our model is quite mild, as shown in Table 9 compared to Table 2. All the estimators with binary treatment and linear response function even yield lower MAEs than

5. Here the result of DML with AutoIV on continuous treatment and non-linear response function shows significantly high MAE over 100,000, likely due to the sensitivity of DML mentioned in Sec. 5.1

| | Estimation Method | | | | | |
|---|---|---|---|---|---|---|
| | 2SLS | IVGMM | DML | Ortho | Poly 2SLS | KernelIV |
| DVAE.CIV | 0.92(0.35) | 0.92(0.35) | 2.11(2.74) | 1.53(2.36) | 1.19(0.62) | 1.06(0.01) |
| CoCoIV (Ours) | **0.79**(0.03) | **0.79**(0.03) | **0.83** (0.14) | **0.82**(0.13) | **0.98**(0.07) | **1.04**(0.01) |

Table 11: Experiment results on DVAE.CIV baseline

| | | Linear response function | | | | | | Non-linear response function | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Method | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV | 2SLS | IVGMM | DML | Ortho | Poly2SLS | KernelIV |
| Binary | Ours | **0.26** (0.09) | **0.31** (0.07) | **1.59** (1.17) | **1.3** (1.25) | 1.66 (3.24) | 0.38 (0.31) | **1.12** (0.17) | **1.06** (0.2) | **0.4** (0.56) | **0.37** (0.47) | **0.47** (0.18) | **0.33** (0.02) |
| | only VAE | 0.43 (0.36) | 0.45 (0.36) | 1.96 (1.68) | 1.75 (1.48) | **0.99** (0.58) | 0.51 (0.36) | 1.23 (0.84) | 1.14 (0.85) | 1.15 (0.95) | 1.28 (1.54) | 0.75 (0.36) | 0.4 (0.12) |
| | only VaDE | 0.28 (0.06) | **0.31** (0.05) | 2.32 (2.0) | 2.0 (1.37) | 1.06 (0.86) | **0.34** (0.2) | 1.28 (0.11) | 1.26 (0.09) | 1.26 (1.08) | 1.06 (0.88) | 0.6 (0.43) | 0.38 (0.14) |
| Continuous | Ours | **0.07** (0.09) | **0.08** (0.09) | **0.03** (0.03) | **0.02** (0.01) | **0.35** (0.07) | **3.51** (0.55) | 1.38 (0.1) | 0.3 (0.15) | 1.35 (0.12) | 1.37 (0.1) | **4.13** (0.32) | 3.82 (1.16) |
| | only VAE | 0.09 (0.1) | 0.11 (0.12) | **0.03** (0.04) | 0.03 (0.02) | 0.37 (0.11) | 3.61 (0.38) | **1.33** (0.1) | 0.35 (0.3) | **1.32** (0.07) | **1.35** (0.07) | 4.24 (0.4) | **3.59** (0.87) |
| | only VaDE | 0.16 (0.17) | 0.17 (0.19) | 0.05 (0.04) | 0.06 (0.05) | 0.44 (0.27) | 3.61 (0.54) | 1.47 (0.13) | **0.26** (0.16) | 1.43 (0.11) | 1.44 (0.08) | 4.3 (0.47) | 4.09 (1.56) |

Table 12: Experiment results on various latent variable models.

MAEs shown in Table 2 in the paper. When MAEs are higher than in the original setting, the difference is lower than 0.1 for 2SLS, IVGMM, and Ortho with continuous treatment and linear response function. However, in the most challenging case with continuous treatment and non-linear response function, as mentioned in Sec. 5.1, all the models, including our own, yield biased estimates, leading to higher MAEs than the original setting.

**Extended experiment on dependence within $\mathbf{D}_1$ and $\mathbf{D}_2$.** We extend our experiments on dependence within $\mathbf{D}_1$, $\mathbf{D}_2$. As we do not assume independence within variables in $\mathbf{D}_1$ or within variables in $\mathbf{D}_2$, we assess how our model works in such setting.

Synthetic datasets are generated in steps similar to Sec. B.1, but we imposed dependence within $\mathbf{D}_1$ and within $\mathbf{D}_2$ as sampling

$$\epsilon_{\mathbf{D}_1} \sim N\left(0, 10\begin{bmatrix}1 & 0.5\\0.5 & 1\end{bmatrix}\right), \qquad \epsilon_{\mathbf{D}_2} \sim N\left(0, 0.25\begin{bmatrix}1 & 0.5\\0.5 & 1\end{bmatrix}\right).$$

As illustrated in Table 10, our model demonstrates better or similar MAEs for IV-based estimators 2SLS and IVGMM. In particular, our model showed relatively better performance on continuous treatment with a linear response function. Aligned with the conclusions in the previous part, performance does not decline abruptly compared to MAEs in Table 2 except for some estimators in non-linear response function.

**Experiment with DGP of DVAE.CIV (Cheng et al., 2023a).** As our works tackled generalized setting, our method indeed can be applied under more restrictive settings of prior works. As shown in Table 11, it outperforms DVAE.CIV on their setup based on stronger assumption (i.e., independence between the covariates and unobserved confounders).

**Experiment on architecture choice** As we propose a framework of generating representation of IVs, one can select a range of architectures with the same framework. For reference, we provide results of different combinations of latent variable models: 1) VAEs for encoders of $\mathbf{Z}$, $\mathbf{C}$, 2) VaDEs for encoders of $\mathbf{Z}$, $\mathbf{C}$. Table 12 demonstrates that, regardless of the choice of architecture, the models derive a similar range of estimates for each estimator. In real-world applications, the choice of architecture can be varied according to the property of underlying data.

In our case, we adopt a VAE-based framework with flexible prior (Sohn et al., 2015) to encode $\mathbf{Z}$ that are associated with $\mathbf{D}_1$. For the prior, we use $p(\mathbf{z}) \approx \frac{1}{N}\sum_{i=1}^{N} p(\mathbf{z} \mid \mathbf{d}^{(i)})$ where $\mathbf{d}^{(i)} \sim p(\mathbf{d})$. We chose $N = 1$ and intend to learn the prior $p(\mathbf{z}) \approx p(\mathbf{z} \mid \mathbf{d})$. Also, we aim to encode representation $\mathbf{C}$ that can capture likely a complex structure of non-IV $\mathbf{D}_2$, which usually have a number more than that of true IVs in observed covariates in real-world scenarios. Thus, we model prior $p(\mathbf{c})$ as a Gaussian Mixture Model, adopting Jiang et al. (2017). By accommodating the mixture model prior, we aim to learn $\mathbf{C}$ expressing complex latent structures of the observed data.