VARIATIONAL COUNTERFACTUAL PREDICTION UNDER RUNTIME DOMAIN CORRUPTION

Anonymous authors
Paper under double-blind review

ABSTRACT

To date, various neural methods have been proposed to address the causal effect estimation based on observational data, where the counterfactual prediction commonly assumes the same distribution and availability of variables at both training and inference (i.e., runtime) stages. In reality, covariate shift commonly happens, and the accessibility of variables is usually impaired due mainly to privacy and ethical concerns. We term the co-occurrence of domain shift and inaccessible variables *runtime domain corruption*, which seriously challenges the generalizability of the trained counterfactual predictor on top of the existence of confoundedness and selection bias. To counteract runtime domain corruption, we subsume counterfactual prediction under the notion of domain adaptation. Specifically, we upper-bound the error w.r.t. the target domain (i.e., runtime covariates) by the sum of source domain error and inter-domain distribution distance. In addition, we build an adversarially unified variational causal effect model, named VEGAN, with a novel two-stage adversarial domain adaptation to implicitly reduce the distribution disparity between treated and control groups first, and between training and inference domains afterwards. We demonstrate that VEGAN outperforms other state-of-the-art baselines on individual-level treatment effect estimation in the presence of runtime domain corruption on benchmark datasets.

1 INTRODUCTION

In predictive analytics, causal inference is increasingly important in guiding decision-making in high-stake domains, such as healthcare (Glass et al., 2013), education (Cordero et al., 2018), e-commerce (Phang et al., 2019), etc. Normally, randomized control trial (RCT) is the gold standard for estimating the causal effect. Given that implementing RCTs is costly, time-consuming, and sometimes ethically intractable, various applications alternatively turn to use the passively collected observational data to perform causal inference in a data-driven fashion (Johansson et al., 2016; Yao et al., 2021). Denoting input variables as $\mathbf{x}$, treatment as t, outcome as y, the observational dataset with N samples $\{(\mathbf{x}_i, t_i, y_i)\}_{i=1}^N$ commonly does not satisfy the RCT standard due to unmeasured confounders and selection bias, which correspond to the two prominent challenges below.

Challenge 1: The *untestable unconfoundedness assumption* assumes no unobserved confounders. However, such an assumption cannot be satisfied in many cases, rendering the causal effect estimation erroneous (Rosenbaum, 2002; Caliendo & Kopeinig, 2008).

Challenge 2: The *imbalanced covariate distribution* between the treated group and the control group makes the treatment assignment t not random and subject to selection bias, which further weakens the effectiveness of causal inference.

A healthcare example to explain these two challenges is that, if only the rich can afford drug A while the poor have to use the cheaper alternative drug B, then people's financial status could be a hidden confounder if unmeasured. Consequently, it confounds the treatment assignment, and the effectiveness of drug A and drug B cannot be validly compared based on the rich and the poor groups due to the skewed distribution of variables. To date, by addressing either of the two challenges or both, several neural approaches (Shalit et al., 2017; Louizos et al., 2017; Shi et al., 2019) are made available for causal effect estimation with observational data. Notably, the majority of such counterfactual predictions ignore the domain shift situation, and also bear an assumption that all input variables used for training are also accessible during the inference stage (i.e., runtime). However, real-world applications commonly see that the medical diagnostic models are learned with high-quality open benchmarks, but in the deployment stage, covariate shift could happen, also not all end-users are able to provide the same set of attributes due to accessibility issues (e.g., high-cost medical checks), privacy constraints (e.g., historical treatments), and ethical concerns (e.g., gender and race). We specifically define this challenge as the runtime domain corruption as follows.

Challenge 3: We define each variable vector $\mathbf{x} = [x_1, x_2, ..., x_d] \in \mathbb{R}^d$ as a concatenation of multi-hot-coded categorical features and non-zero numerical features. During training, all d entries x_s, $\forall s \leq d$, are available and assigned corresponding values. Then, during inference, *runtime domain corruption* occurs when the covariate distribution shifts, and vector $\mathbf{x}$ contains an arbitrary number of unavailable variables which are set to $x_{s'} = 0$, $s' \leq d$.

Domain Shift vs. Runtime Domain Corruption. Runtime domain corruption can be interpreted as one step above the well-defined domain/covariate shift, where the runtime not only witnesses changed covariate distribution but also missing values. Note that Challenge 3 specifies the input vector $\mathbf{x}$ only contains non-zero values for known attributes, which can be conveniently achieved via common preprocessing steps like rescaling, exponential, and normalization. In short, in our definition, domain shift happens when covariate distribution shifts, while domain corruption is caused by the co-occurrence of domain shift and missing values. Compared with conventional domain shift, runtime domain corruption more aggressively challenges the generalizability of the trained counterfactual prediction model, because variables deemed important in training might no longer be present during inference, and the domain-invariant patterns are unable to be mapped to those missing variables. Furthermore, a high corruption rate of runtime variables can make the counterfactual predictor learned on full training data incur large generalization errors, and rendering data imputation impractical. Though one can consider discarding the unavailable attributes in the training set, it may lead to an underfitting issue and also it is impractical as the missing attributes can differ among individuals (e.g., users may choose to withhold different personal information).

The Use of Zero-Padding. Specifically, we do not discard inaccessible variables during runtime, which will cause dimensionality reduction. Instead, we pad zeros to entries that correspond to missing variables such that the dimensionality is kept unchanged. Also, zero-padding is a more feasible solution in real applications, as each runtime instance $\mathbf{x}$ may have an arbitrary number and combination of attributes missing, rendering it impractical to train a specific latent feature extractor for each case. In contrast, zero-padding is a more flexible and scalable approach for learning domain-invariant latent representations with a shared feature extractor, where all unknown values of variables are padded with zeros will be filtered out during projection.

This work focuses on causal inference using the Neyman-Rubin potential outcome framework (Neyman, 1923; Rubin, 2005) under the runtime domain corruption circumstance. In this work, we aim to learn a robust, causal, and domain-invariant latent representation $\mathbf{z}$ of variable $\mathbf{x}$, for which the latent distributions across various domains are well-balanced to counter the aforementioned three challenges simultaneously. Our main contributions are:

- We identify an important yet largely understudied setting in causal inference, namely runtime domain corruption where each unit tends to have an incomplete set of covariates during inference besides the potentially shifted covariates. In our paper, we propose an investigation of runtime domain corruption, confoundedness, and selection bias in a unified view.
- We derive the upper bound of the generalization error by extending the in-sample causal inference to the corrupted out-of-sample scenario. To efficiently optimize the multiple Kullback-Leibler (KL) divergence terms, we propose a two-stage domain adaptation scheme, namely the Variational autoEncoder Generative Adversarial Network (VEGAN) for unifying multiple inter-domain distances.
- We compare VEGAN to state-of-the-art baselines for performing predictions on both in-sample covariates and out-of-sample, corrupted runtime covariates. The empirical results demonstrate our model's stronger robustness to runtime domain corruption.

2 Related Work

We relate our work to the representation learning branch in causal inference, which is overlapped with the domain adaptation field due to its unique counterfactual nature. With the strong representation learning ability of deep learning (Bengio et al., 2013; LeCun et al., 2015), new works are proliferating by leveraging the deep learning framework to learn the latent representation on top of the observed covariates. The TARNet (Shalit et al., 2017) builds a shared feature extractor followed by a two-headed neural network to model the outcome for each type of treatment separately. It also incorporates the integral probability metric (IPM), e.g., Wasserstein distance (Vallender, 1974) or maximum mean discrepancy (MMD) (Gretton et al., 2012), to minimize the distance of the learned latent covariate distribution between treated and control groups to mitigate the selection bias. Following that, a variational autoencoder (VAE) framed CEVAE model (Louizos et al., 2017) emphasizes handling the confounding problem by building robust latent representation, and its performance is stated to be more robust than many previous methods, and dragonnet (Shi et al., 2019) leverages the neural net-enhanced propensity estimation and the innovative targeted regularization for causal effect estimation. In addition, other works such as GANITE (Yoon et al., 2018) and DeepMatch (Kallus, 2020) adopt generative adversarial network (GAN) (Goodfellow et al., 2014) and build their own designated GAN learning network. Our work differs from theirs and we relate our work further to the unsupervised domain adaptation (Blitzer et al., 2007) by building a GAN-integrated VAE model, as we additionally consider the domain corruption situation where the trained model's performance could dramatically decline in causal effect estimation.

In addition, it is noted that Jesson et al. (2020) propose an uncertainty estimation plug-in to the state-of-the-arts such as TARNet, CEVAE, and Dragonnet to allow these models to estimate epistemic uncertainty in high-dimensional conditional average treatment effect (CATE) estimation, thus to inform the decision maker to be vigilant when making recommendations if the high uncertainty present. It considers the domain shift during runtime, but it emphasizes making no treatment recommendation if the epistemic uncertainty exceeds a certain threshold. Hence, our work differs from it as we focus on more accurate treatment effect estimation. It should also be noted that some existing works (Qu & Lipkovich, 2009; Mayer et al., 2020; Berrevoets et al., 2022) have been proposed for treatment effect estimation with missing values, where the core is to leverage imputation algorithms to handle the missing values. Since runtime domain corruption also includes domain shift, the imputed target domain data could still deviate heavily from the source domain, rendering those methods inaccurate in such conditions. Furthermore, imputation algorithm falls short in accurate imputing when the number of missing value is large, it even becomes useless when the attributes are completely missing at distribution level during inference stage.

We also relate our work to algorithmic fairness topics, e.g., disparate learning processes (DLPs), in which the ethically concerned, privacy-related features are not available or impermissible to be used during runtime (Lipton et al., 2018; Simons et al., 2021). A similar approach to DLPs is a doubly-robust counterfactual prediction model with additional handling of the confounding problem during training (Coston et al., 2020). However, its causal inference differs from the common causal effect estimation as it assumes that one of the potential outcomes is a known constant for a binary treatment, and is hence inapplicable to the problem studied in this paper.

3 Problem Setup and Analysis

For simplicity, we consider binary treatment t of 1 or 0 to denote the treated group and the control group, respectively. The individual treatment effect (ITE) for a variable vector $\mathbf{x}$ is defined as:

$$\tau(\mathbf{x}) = \mathbb{E}[y_1^* - y_0^* | \mathbf{x}], \tag{1}$$

where y_1^* and y_0^* are potential outcomes with treatment $t = 1$ and $t = 0$ respectively. To validly estimate $\tau(\mathbf{x})$, the following unconfoundedness assumption is required.

Assumption 1 (Unconfoundedness): *Treatment assignment is independent to the potential outcome given the pre-treatment covariate* $\mathbf{x}$, i.e., $t \perp\!\!\!\perp \{y_0^*, y_1^*\} | \mathbf{x}$.

Let $\Psi : \mathcal{X} \times \{0, 1\} \to \mathbb{R}$ parameterizes the hypothesis, our goal is to build the regression model $\Psi_{1-t}(\mathbf{x}, t) = \mathbb{E}[y|X = \mathbf{x}, T = 1 - t]$ with *observed outcome* y which can accurately recover the counterfactual outcome for $\mathbf{x}$ with treatment t, thus the causal effect can be calculated as $\Psi_1 - \Psi_0$. However, Challenges 1 and 2 arise when the observational dataset does not follow the RCT standard, making the trained models Ψ_1 and Ψ_0 unable to reflect the true treatment outcome for each $\mathbf{x}$.

We perceive the observed covariates of treated and control groups from the conventional domain shift perspective, in which covariate $\mathbf{x}$ is a noisy measurement, normally less informative and more confounded (Griliches & Hausman, 1986; Maddala & Nimalendran, 1996), than the domain-invariant latent representation $\mathbf{z}$. Therefore, the unconfoundedness changes from $t \perp\!\!\!\perp \{y_0^*, y_1^*\}|\mathbf{x}$ to $t \perp\!\!\!\perp \{y_0^*, y_1^*\}|\mathbf{z}$. In addition to the treated and control groups from the in-sample set, this paper considers the runtime causal effect estimation where the out-of-sample set is affected by domain corruption (Challenge 3), where the trained model's generalizability is significantly challenged.

3.1 Target Domain Error Upper Bound

Shalit et al. (2017) have shown that the expected precision in estimation of heterogeneous effect (PEHE), or ϵ_{PEHE}, is upper-bounded by both the trained model error ϵ_{F} on factual outcomes and the distance between treated and control distributions, measured by integral probability metric (IPM). Since their derived upper bound for ϵ_{PEHE} does not consider runtime domain corruption on out-of-sample variables, we fill the gap by deriving the bound in Theorem 1.

Theorem 1: *Let $\phi : \mathcal{X} \to \mathcal{Z}$ be the invertible latent representation mapping function (a.k.a. feature extractor) with inverse Φ. Let $\Psi : \mathcal{Z} \times \{0, 1\} \to \mathbb{R}$ be the updated hypothesis. Let $\mathcal{F}$ be a family of functions $f : \mathcal{Z} \to \mathbb{R}, f \in \mathcal{F}$. The source domain is the observational data for treated and control groups, and the target domain is the runtime test/inference set with corrupted variables. Under the conditions of Assumption 8.1, Lemma 8.1, Theorem 8.1, and the upper breakdown of absolute value inequality, we derive the upper bound of target domain error (i.e., generalization error) as*[1]*:*

$$\begin{aligned}\epsilon_{\text{PEHE}}^{tr} &\leq \epsilon_{\text{PEHE}}^{sr} + B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr}) \\ &\leq 2\left[\epsilon_{\text{F}}^{t=1} + \epsilon_{\text{F}}^{t=0} + B_\phi\left(\text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{t=1}, \mathbb{P}_\phi^{t=0}) + \frac{1}{2}\text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr})\right)\right],\end{aligned} \tag{2}$$

where ϵ_F^t denotes the factual training error, $\mathbb{P}_\phi^t$ is the probability measure within treatment group t in the training set, ϵ_{PEHE}^{tr} and ϵ_{PEHE}^{sr} respectively indicate the target and source domain errors, $\mathbb{P}_\phi^{tr}$ and $\mathbb{P}_\phi^{sr}$ are probability measures which denote the covariate distribution in target domain and source domain respectively, and B_ϕ is a bounded constant.

We provide the proof in in Appendix 8.1. The upper bound given in Theorem 1 suggests that, to bring down the runtime/target domain error $\epsilon_{\text{PEHE}}^{tr}$, we are essentially minimizing: (1) the prediction errors on observed outcomes; (2) the imbalance between treated and control groups; (3) the discrepancy between the training and test sets altogether. It guides our algorithm design in general for runtime causal inference. Note that if no domain corruption exists, which means $\mathbb{P}_\phi^{tr} = \mathbb{P}_\phi^{sr}$ and thus $\text{IPM}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr}) = 0$, the runtime error becomes identical to the source domain error $\epsilon_{\text{PEHE}}^{tr} = \epsilon_{\text{PEHE}}^{sr}$.

4 Variational Inference

Our solution is built upon the variational autoencoder (VAE). To start with, in this section we introduce the minimization of the factual error ϵ_{F}, the distribution disparity between treated and control groups first, and between training and runtime domains afterwards.

4.1 Evidence Lower Bound

For modelling the *observed treatment outcome* y, we use the maximum likelihood estimation (MLE) to approximate the parameters. For simplicity, log is commonly used to decompose the joint marginal likelihood $p(\mathbf{y})$ into:

$$\log p(\mathbf{y}) = \sum_{k=1}^{N} \log p(y_k) = \sum_{i=1}^{N_1} \log p(y_i|t_i = 1) + \sum_{j=1}^{N_0} \log p(y_j|t_j = 0), \tag{3}$$

where $\mathbf{y} = [y_1, y_2, \ldots, y_N]$ is a vector hosting all N samples' ground truth, $N = N_1 + N_0$, N_1 and N_0 respectively denote the number of samples in treated and control groups. Thus, to maximize the joint marginal log-likelihood of observing $\mathbf{y}$, we can maximize each individual log-likelihood $\log p(y|t)$, and the expression $\log p(y|t)$ itself serves as a general term to represent a distribution of all the observed y.

[1]We appreciate and follow some notation conventions set by Shalit et al. (2017) and Johansson et al. (2020).

As we assume that there exists a latent representation $\mathbf{z}$ and treatment t that causally determine the observed treatment response y_t, i.e., $y_t \sim p(y|\mathbf{z}, t)$ in a probabilistic way, while the observed proxy $\mathbf{x}$ does not have any causal relations but statistical correlations with y. Due to the potentially high dimensionality of $\mathbf{z}$, the marginal likelihood $p_t(y) = p(y|t)$ of treatment group t is intractable. Here, we apply the variational methodology (Kingma & Welling, 2013) to our scenario to tackle $p(y|t)$ by establishing an encoder network ϕ_t to learn latent representation $\mathbf{z}_t \sim p_{\phi_t}(\mathbf{z}|\mathbf{x})$, and a decoder network Ψ_t to estimate treatment response $y_t \sim p_{\Psi_t}(y|\mathbf{z}, t)$. The detailed probabilistic mechanism to model $p_{\Psi_t}(y|\mathbf{z}, t)$ is described in Appendix 8.2, which leads us to the evidence lower bound (ELBO$_t$) as follows:

$$\begin{aligned} \log p_{\Psi_t}(y|t) &\geq \text{ELBO}_t \\ &= \mathbb{E}_{\mathbb{P}_{\Psi_t}}[\log p_{\Psi_t}(y|\mathbf{z}, t)] - D_{\text{KL}}(\mathbb{P}_{\phi_t}||\mathbb{P}_{\Psi_t}), \end{aligned} \tag{4}$$

where $\mathbb{P}_{\phi_t}$ and $\mathbb{P}_{\Psi_t}$ are posterior and prior distributions respectively. $D_{KL}(\cdot)$ returns the Kullback–Leibler (KL) divergence between two distributions. As such, the task of maximizing the intractable $\log p_{\Psi_t}(y|t)$ can be indirectly solved by pushing up its associated ELBO$_t$, thus minimizing the factual error ϵ_{F}^t. According to the decomposition in Eq. 3, our objective is to maximize the sum of two ELBOs for treated and control groups:

$$\text{ELBO} = \sum_{t \in \{0,1\}} \text{ELBO}_t. \tag{5}$$

It is worth noting that, our derived bound ELBO can be easily extended from our binary treatment setting to scenarios that involve multiple treatments.

4.2 Treated/Control Domain Adaptation

According to the second term in Eq. 4, for $t \in \{0, 1\}$, we have both KL divergence terms that regularize the posterior distribution $\mathbb{P}_{\phi_t}$ and the prior distribution $\mathbb{P}_{\Psi_t}$ as follows :

$$D_{\text{KL}}(\mathbb{P}_{\phi_1}||\mathbb{P}_{\Psi_1}),\ \ D_{\text{KL}}(\mathbb{P}_{\phi_0}||\mathbb{P}_{\Psi_0}). \tag{6}$$

By pushing up the ELBO in Eq. 5, one can notice that both posteriors $\mathbb{P}_{\phi_1}$ and $\mathbb{P}_{\phi_0}$ are regularized to approach the same prior distribution if we set the prior $\mathbb{P}_{\Psi_1}$ and $\mathbb{P}_{\Psi_0}$ to be the same, e.g., standard normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$. Thus, the domain adaptation (DA) for both groups can be naturally achieved to balance their latent distributions and counter Challenge 2 by adjusting the priors using the VAE framework. It is worth noting that KL divergence is an unbounded asymmetric distribution distance measure (Kullback & Leibler, 1951) which does not belong to IPM, so we replace it with a bounded symmetric distribution similarity measurement in Section 5 as a better approximation.

4.3 Training/Runtime Domain Adaptation

In addition to the DA across treated and control groups within the training set, we would also like to do DA between the entire training and runtime sets to minimize the tightness bound $B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr})$ given in Lemma 8.1 and thus alleviate Challenge 3. As such, for a well-trained model, we aim to make the out-of-sample performance as good as the in-sample performance, i.e., the out-of-sample results would not deviate from the in-sample ones drastically while keeping good in-sample performance.

Intuitively, if the VAE prediction framework is applied to the full runtime test set $\{(\mathbf{x}_j^{tr}, t_j^{tr}, y_j^{tr})\}_{j=1}^{N'}$ (i.e., target domain in this context), one can end up with the objective to be maximized similar to the ELBO$_t$ presented in Eq. 4 as follows:

$$\Gamma_{\phi_t^{tr}, \Psi_t^{tr}} = \mathbb{E}_{\mathbb{P}_{\phi_t}^{tr}}[\log p_{\Psi_t}^{tr}(y|\mathbf{z}, t)] - D_{\text{KL}}(\mathbb{P}_{\phi_t}^{tr}||\mathbb{P}_{\Psi_t}^{tr}). \tag{7}$$

However, the label y^{tr} and treatments t^{tr} are apparently unknown in practice, and such an objective cannot be optimized. Since the only available information is the runtime covariates which can be used to extract the domain-invariant representation from DA, the second term in Eq. 7 can be utilised for such purpose with a mild modification.

To be specific, we alternatively walk around to minimize the KL divergence between the runtime posterior $\mathbb{P}_\phi^{tr}$ and the entire training set posterior $\mathbb{P}_\phi^{sr}$, namely $D_{\text{KL}}(\mathbb{P}_\phi^{tr}||\mathbb{P}_\phi^{sr})$, where ϕ is a shared

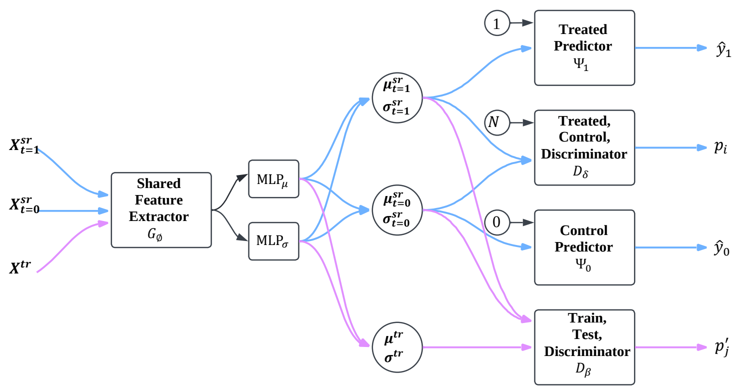
$X^{sr}_{t=1}$
$X^{sr}_{t=0}$
X^{tr}
Shared Feature Extractor G_ϕ
MLP$_\mu$
MLP$_\sigma$
$\mu^{sr}_{t=1}$ $\sigma^{sr}_{t=1}$
$\mu^{sr}_{t=0}$ $\sigma^{sr}_{t=0}$
μ^{tr} σ^{tr}
1
N
0
Treated Predictor Ψ_1
Treated, Control, Discriminator D_δ
Control Predictor Ψ_0
Train, Test, Discriminator D_β
$\hat{y}_1$
p_i
$\hat{y}_0$
p'_j

Figure 1: Unifying the KL-divergences by GAN. The black flows between G_ϕ and MLPs denote the internal link. The pink flows indicate second stage domain adaptation between train and test sets.

feature extractor. Thus, to achieve the second-stage DA, our proposed ultimate evidence lower bound (ELBO$_{ulti}$) for the intractable joint log-likelihood $p(y)$ is:

$$\begin{aligned}\log p(y) &\geq \text{ELBO} \\ &\geq \text{ELBO}_{ulti} \\ &= \text{ELBO} - D_{\text{KL}}(\mathbb{P}^{tr}_\phi || \mathbb{P}^{sr}_\phi).\end{aligned} \tag{8}$$

5 Adversarial Learning

So far, we have multiple $D_{\text{KL}}(\cdot)$ terms: two in Eq. 6 align the posteriors to the same prior $\mathcal{N}(\mathbf{0}, \mathbf{1})$, and the one in Eq. 8 that aligns the posteriors of training set and runtime set. As the direct calculation of KL divergence is computationally inefficient and may even be infeasible with high dimensional data (Nelken & Shieber, 2006; Moral et al., 2021), we propose to implicitly minimize them and unify these terms into a compact generative adversarial network (GAN) (Goodfellow et al., 2014) shown in Figure 1. Apart from that, optimizing the minimax game in GAN is equivalent to minimizing the Jensen-Shannon divergence (Goodfellow et al., 2014), which is a bounded symmetric distribution similarity measurement (Nielsen, 2019). Such technique resonates with adversarial variational Bayes introduced in Mescheder et al. (2017), while our motivation and implementation differ. Here, we present our **V**ariational auto**E**ncoder **G**enerative **A**dversarial **N**etwork runtime counterfactual regression model, coined as VEGAN. In what follows, we unfold the design details of VEGAN.

Firstly, we instantiate ϕ, the shared feature extractor among $\mathbf{x}^{sr}_{t=1}$, $\mathbf{x}^{sr}_{t=0}$ and $\mathbf{x}^{tr}$. It includes G_ϕ and the following two multi-layer perceptrons (MLPs) that map all the data from the original $\mathbb{R}^d$ into latent space $\mathbb{R}^l$. Due to the variational nature of the model, the j-th latent dimension of individual i, denoted by $\mathbf{x}^{sr}_i$, is modelled by a Gaussian distribution with its dedicated mean μ_{ij} and variance σ^2_{ij} as follows:

$$p^{sr}_\phi(\mathbf{z}_i|\mathbf{x}_i) = \prod_{j=1}^{l} \mathcal{N}(\mu_{ij}, \sigma^2_{ij}), \tag{9}$$

where mean μ_{ij} and standard deviation σ_{ij} are respectively the j-th element of latent representations $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$. In VEGAN, $\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i \in \mathbb{R}^l$ are denoted as:

$$\boldsymbol{\mu}_i = \text{MLP}_\mu(G_\phi(\mathbf{x}^{sr}_i)), \boldsymbol{\sigma}_i = \text{MLP}_\sigma(G_\phi(\mathbf{x}^{sr}_i)), \tag{10}$$

which allows us to obtain the latent representation $\mathbf{z}_i$ for the subsequent DAs and inference.

Secondly, for the treated/control group DA, we propose an adversarial way to implicitly reduce inter-domain distribution distance. In this regard, G_ϕ is essentially our generator for notation simplicity, and a discriminator D_δ is thus designed to pair up the generator to facilitate adversarial learning. The minimax game is designed as: the discriminator D_δ tries to differentiate the standard Gaussian sample $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$ from $\mathbf{z}^{sr}$ learned from the training sample; in the meantime, feature extractor G_ϕ tries to update the latent representation $\mathbf{z}^{sr}$ to make it indistinguishable from $\mathbf{n}$. When an equilibrium state is reached, the treated and control domains are well adapted because both latent representations $\mathbf{z}^{sr}$ fall in the same distribution. Thus, the two terms $D_{\text{KL}}(\mathbb{P}_{\phi_1}||\mathbb{P}_{\Psi_1})$ and

$D_{\mathrm{KL}}(\mathbb{P}_{\phi_0}||\mathbb{P}_{\Psi_0})$, are minimized in an adversarial way. As Figure 1 shows, the output $p_i = D_\delta(\mathbf{w}_i)$ is the scalar probability of being a Gaussian sample, where $\mathbf{w}_i = \eta_i \mathbf{n}_i + (1-\eta_i)\mathbf{z}_i^{sr}$ with $\eta_i \in \{1, 0\}$ labeling the i-th sample from two buckets (1 for Gaussian samples, and 0 for training samples). Note that $\mathbf{n}$ is resampled for every training case in every epoch. In our supervised learning setting, we have the cross-entropy loss for the discriminator $D_\delta(\cdot)$:

$$l(\mathbf{w}_i) = \eta_i \log D_\delta(\mathbf{w}_i) + (1-\eta_i)\log(1 - D_\delta(\mathbf{w}_i)). \tag{11}$$

Then, in an adversarial setting, the minimization of two KL divergence in Eq. 6 for treated/control domain adaptation is replaced by the following:

$$\begin{aligned} &\min_\phi \max_\delta \mathbb{E}_{\forall i}[\eta_i \log D_\delta(\mathbf{w}_i) + (1-\eta_i)\log(1 - D_\delta(\mathbf{w}_i))] \\ \iff &\min_\phi \max_\delta \mathbb{E}_{\mathcal{N}(\mathbf{0},\mathbf{1})}[\log D_\delta(\mathbf{n})] + \mathbb{E}_{\forall i}[\log(1 - D_\delta(\mathbf{z}_i^{sr}))]. \end{aligned} \tag{12}$$

Analogously, for the train/runtime domain adaptation, we design another discriminator $D_\beta(\cdot)$ to form the second GAN system between $G_\phi(\cdot)$ and $D_\beta(\cdot)$, where $D_\beta(\cdot)$ predicts the probability p'_j of the sample from the source domain, i.e., training set. The only difference from the first GAN system is that, it takes the training sample as real while the runtime sample is treated as fake. Thus, $D_{\mathrm{KL}}(\mathbb{P}_\phi^{tr}||\mathbb{P}_\phi^{sr})$ is replaced by the followings:

$$\min_\phi \max_\beta \mathbb{E}_{\forall i}[\log D_\beta(\mathbf{z}_i^{sr})] + \mathbb{E}_{\forall j}[\log(1 - D_\beta(\mathbf{z}_j^{tr}))]. \tag{13}$$

Finally, to build the probabilistic model $p(y|\mathbf{z}, t)$, we model each of the treatment classes through two separate MLPs, namely Ψ_1 and Ψ_0 respectively, thus the general representation of modelling the observed outcome y for individual i is given as $p_{\Psi_t}(y_i|\mathbf{z}_i, t_i) = \mathcal{N}(\hat{\mu}_i, \hat{\sigma}_i^2)$, where $\hat{\mu}_{t,i} = \Psi_{t_i}(\mathbf{z}_i^{sr}, t_i)$, and we follow Louizos et al. (2017) to set $\hat{\sigma}_i^2 = 1$ for simplicity.

In a nutshell, to promote a computationally efficient algorithm, we propose to minimize the following loss function $\mathcal{L}$ along with optimizing the minimax game together such that the ELBO_{ulti} in Eq. 8 will be maximized:

$$\min_{\phi,\Psi_1,\Psi_0} \max_{\delta,\beta} \mathbb{E}_{\mathcal{N}}[\log D_\delta(\mathbf{n})] + \mathbb{E}_{\mathcal{Z}^{sr}}[\log(1 - D_\delta(\mathbf{z}))] + \mathbb{E}_{\mathcal{Z}^{sr}}[\log D_\beta(\mathbf{z})] + \mathbb{E}_{\mathcal{Z}^{tr}}[\log(1 - D_\beta(\mathbf{z}))] + \mathcal{L}, \tag{14}$$

where $\mathcal{L} = -\Big(\mathbb{E}_{\mathbb{P}_{\Psi_1}^{sr}}[\log p_{\Psi_1}(y|\mathbf{z}, t=1)] + \mathbb{E}_{\mathbb{P}_{\Psi_0}^{sr}}[\log p_{\Psi_0}(y|\mathbf{z}, t=0)]\Big)$.

We summarize our VEGAN model optimization scheme in Appendix 8.11. Please note that the notation changed accordingly as the reparameterization trick $\mathbf{z} = \omega(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}_\phi^2, \boldsymbol{\epsilon})$ (detailed in Appendix 8.3) is applied as a necessity to get the gradient ∇_ϕ for the feature extractor G_ϕ. Also, the original minimax game in Eq. 14 is adjusted to the double minimization tradition for gradient descent.

6 Experiments

We perform evaluation on two datasets, one is the widely used semi-synthetic dataset IHDP (Infant Health Development Program) (Hill, 2011), and the other is the high-dimensional synthetic dataset from ACIC (Atlantic Causal Inference Conference) (ACIC, 2019). Our models, $\mathrm{VEGAN_I}$ (with only treated/control domain adaptation) and $\mathrm{VEGAN_{II}}$ (with both adaptation steps), are implemented using Pytorch (Paszke et al., 2019), and their hyperparameter settings are left in Appendix 8.9. We compare our models with seven state-of-the-arts: TARNet (Shalit et al., 2017), BTARNet (Jesson et al., 2020), $\mathrm{CFR_{WASS}}$ (Shalit et al., 2017), SITE (Yao et al., 2018), $\mathrm{Dragonnet_{Base}}$ (Shi et al., 2019), $\mathrm{Dragonnet_{TR}}$ (Shi et al., 2019), CEVAE (Louizos et al., 2017) with their provided default settings. We adopt two well-established metrics: (1) ϵ_{CATE} (Hill, 2011), i.e., error of conditional average treatment effect (CATE) for overall performance; and (2) $\sqrt{\epsilon_{\mathrm{PEHE}}}$ (Shalit et al., 2017), i.e., precision in estimation of heterogeneous effect (PEHE) for individual-level performance.

Notes on Additional Results Appended. As we are interested in causal inference under runtime domain corruption, we lay more emphasis on the out-of-sample predictions with corrupted covariates at the individual level. The details for generating out-of-sample test data with domain corruption are in Appendix 8.4 along with the analysis on the varying magnitudes of the Gaussian noise, where we use corruption level (CL) to denote the average ratio of corruption per selected variable distribution.

Metric / Model	ϵ_{CATE}	$\sqrt{\epsilon_{\text{PEHE}}}$
TARNet	0.253	1.033
BTARNet	0.232	1.207
CFR$_{\text{WASS}}$	0.265	**0.992**
SITE	0.256	1.016
Dragonnet$_{\text{Base}}$	0.342	1.591
Dragonnet$_{\text{TR}}$	0.357	1.524
CEVAE	0.329	2.604
VEGAN$_{\text{I}}$	**0.187**	1.450
VEGAN$_{\text{II}}$	0.239	1.394

Table 1: ϵ_{CATE} and $\sqrt{\epsilon_{\text{PEHE}}}$ of in-sample counterfactual prediction on IHDP dataset.

CL / Model	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$	1
TARNet	$1.725 \pm .17$	$2.216 \pm .26$	$2.455 \pm .26$	$3.524 \pm .42$	$5.845 \pm .76$
BTARNet	$1.827 \pm .20$	$2.327 \pm .25$	$2.575 \pm .25$	$3.728 \pm .44$	$5.958 \pm .80$
CFR$_{\text{WASS}}$	$\mathbf{1.610 \pm .15}$	$\mathbf{2.070 \pm .24}$	$2.469 \pm .27$	$3.503 \pm .42$	$5.979 \pm .85$
SITE	$1.794 \pm .19$	$2.136 \pm .25$	$2.488 \pm .27$	$3.427 \pm .43$	$5.531 \pm .70$
Dragonnet$_{\text{Base}}$	$2.111 \pm .24$	$2.272 \pm .26$	$2.507 \pm .28$	$3.190 \pm .39$	$4.521 \pm .60$
Dragonnet$_{\text{TR}}$	$2.023 \pm .23$	$2.219 \pm .25$	$2.516 \pm .28$	$3.221 \pm .40$	$4.760 \pm .63$
CEVAE	$3.074 \pm .37$	$2.910 \pm .32$	$2.987 \pm .35$	$3.672 \pm .46$	$4.164 \pm .53$
VEGAN$_{\text{I}}$	$1.872 \pm .20$	$2.178 \pm .25$	$2.371 \pm .27$	$3.030 \pm .38$	$4.873 \pm .62$
VEGAN$_{\text{II}}$	$1.720 \pm .16$	$2.099 \pm .23$	$\mathbf{2.326 \pm .25}$	$\mathbf{2.954 \pm .35}$	$\mathbf{3.918 \pm .47}$

Table 2: $\sqrt{\epsilon_{\text{PEHE}}}$ of out-of-sample prediction on IHDP dataset with different corruption levels on private features.

We leave CATE prediction error ϵ_{CATE} to Appendix 8.10, where VEGAN demonstrates the superiority of its DA schemes. In the main results, we focus on individual-level causal effect estimation measured by $\sqrt{\epsilon_{\text{PEHE}}}$. To demonstrate the generalizability of our proposed second stage adversarial as a plug-in, we study its compatibility with the most representative baseline TARNet in Appendix 8.5. Additionally, the training stability of VEGAN is analyzed in Appendix 8.6.

6.1 Performance on IHDP

The semi-synthetic dataset contains 25 real-world covariates relating to characteristics of the newborns and their mothers, and we use the non-linear response surface B for simulated outcomes designed by Hill (2011). In IHDP, we selectively corrupt 7 privacy-related features described in Appendix 8.8 with CL $\in \{\frac{1}{20}, \frac{1}{8}, \frac{1}{5}, \frac{1}{3}, 1\}$.

Table 1 shows the in-sample predictions, for which our models perform the best in estimating CATE while staying competitive on individual-level causal effect estimation. For out-of-sample predictions, we conduct the tests on 5 randomly corrupted test sets with ratio CL $\in \{\frac{1}{20}, \frac{1}{8}, \frac{1}{5}, \frac{1}{3}, 1\}$. Notably, CL $= 1$ represents an extreme case where all the sensitive features are completely inaccessible during runtime and only the remaining 18 variables are available for prediction. As Table 2 demonstrates, VEGAN$_{\text{II}}$ yields the second best performance when the domain corruption is relatively restrained, but obtains the highest accuracy after the corruption ratio increases to $\frac{1}{5}$. The best baseline is CFR$_{\text{WASS}}$ when CL is low, but it overfits the training set significantly and thus does not generalize to a higher domain corruption level, while VEGAN$_{\text{II}}$ is less sensitive to the stronger corruption of private variables. VEGAN$_{\text{II}}$ also outperforms its base model VEGAN$_{\text{I}}$ consistently owing to the additional DA between training and runtime domains.

6.2 Performance on ACIC

We use the fully-synthetic high-dimensional dataset with 100 simulations from ACIC (2019), which includes 200 features and 1,000 samples from each simulation. Because there is no clear definition for sensitive features, we allow the corruption to take place for all the features in ACIC dataset with a ratio of CL $\in \{\frac{1}{20}, \frac{1}{8}, \frac{1}{5}, \frac{1}{3}\}$. With this, we can mimic situations where individuals can withhold an arbitrary combination of variables in privacy-sensitive applications. Note that, we omit CL $= 1$ in ACIC dataset as it will set all variables to zero and thus make any predictions infeasible. This setting also allows for additional comparisons with the imputation method, which is a natural choice to handle missing values. Specifically, we implement the imputation algorithm MICE (Van Buuren & Oudshoorn, 2000) widely adopted in treatment effect estimation (Mayer et al., 2020; Berrevoets et al., 2022) to enhance all our baselines. Given the space limit, we only include the the most performant baseline CEVAE after enhancement (denoted as CEVAE-IMP) in Tables 3 and 4. The full experiments and analysis with imputation for all baselines are left to Appendix 8.7.

As a result, VEGAN outperforms all the other models for both in-sample and out-of-sample prediction as shown in Table 3 and Table 4. We notice that when the domain corruption level climbs, the fluctuations of prediction errors are small in magnitude.

Volatility Analysis. To better quantify the advantage of VEGAN under domain corruption on ACIC, in Figure 2 we analyse the deviation (Δ) of each model's performance between in-sample and corrupted prediction tasks, i.e., $\Delta = 100\% \times |\epsilon_{\text{in-sample}} - \epsilon_{\text{corrupted}}| / \epsilon_{\text{in-sample}}$. Δ quantifies the instability of the model, as we commonly rely on models obtained with the training set and prefer lower gener-

Model \ Metric	ϵ_{CATE}	$\sqrt{\epsilon_{\text{PEHE}}}$
TARNet	0.214	0.808
BTARNet	0.176	0.924
CFR_{WASS}	0.216	0.708
SITE	0.293	1.383
$\text{Dragonnet}_{\text{Base}}$	0.186	2.244
$\text{Dragonnet}_{\text{TR}}$	0.183	2.503
CEVAE	0.286	0.702
CEVAE-IMP	0.286	0.702
VEGAN_{I}	**0.143**	0.496
VEGAN_{II}	0.147	**0.476**

Table 3: ϵ_{CATE} and $\sqrt{\epsilon_{\text{PEHE}}}$ of in-sample counterfactual prediction on ACIC datasets. Note that imputation does not alter in-sample results.

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	$0.813 \pm .05$	$0.738 \pm .04$	$0.798 \pm .05$	$0.754 \pm .04$
BTARNet	$0.797 \pm .04$	$0.754 \pm .04$	$0.773 \pm .04$	$0.768 \pm .04$
CFR_{WASS}	$0.730 \pm .04$	$0.655 \pm .03$	$0.738 \pm .04$	$0.644 \pm .04$
SITE	$1.482 \pm .08$	$1.361 \pm .08$	$1.365 \pm .09$	$1.586 \pm .10$
$\text{Dragonnet}_{\text{Base}}$	$2.250 \pm .02$	$2.192 \pm .02$	$2.142 \pm .02$	$2.063 \pm .02$
$\text{Dragonnet}_{\text{TR}}$	$2.517 \pm .02$	$2.456 \pm .02$	$2.406 \pm .02$	$2.342 \pm .02$
CEVAE	$0.646 \pm .02$	$0.629 \pm .02$	$0.594 \pm .02$	$0.581 \pm .03$
CEVAE-IMP	$0.626 \pm .02$	$0.601 \pm .02$	$0.577 \pm .02$	$0.568 \pm .03$
VEGAN_{I}	$\mathbf{0.466 \pm .01}$	$\mathbf{0.489 \pm .01}$	$0.488 \pm .01$	$0.501 \pm .01$
VEGAN_{II}	$0.490 \pm .01$	$0.493 \pm .01$	$\mathbf{0.471 \pm .01}$	$\mathbf{0.455 \pm .00}$

Table 4: $\sqrt{\epsilon_{\text{PEHE}}}$ of out-of-sample prediction on ACIC dataset with different corruption levels on all features.

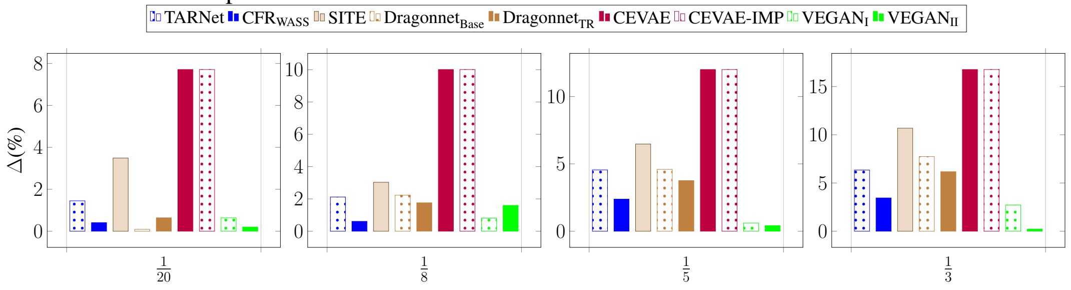

Figure 2: The volatility of the model's performance in corrupted domains with different CL.

alization errors. We include CEVAE-IMP's results while leaving other imputation-enhanced baselines to Appendix 8.7. All models become more volatile as CL increases, while VEGAN_{II} maintains an excellent stability with only 0.22% variation at corruption level $\frac{1}{3}$ and achieves the best accuracy in terms of $\sqrt{\epsilon_{\text{PEHE}}}$. The exact numerical values of Δ are recorded in Appendix 8.10.

Comparison with Data Imputation. When the corruption rate is low, using imputation is generally helpful for slightly increasing the prediction performance, but the improvements are smaller when more attributes are corrupted (e.g., at a rate of 1/5 and 1/3). Also, as 8.7 suggests, imputation methods do not make a difference in volatility, as the imputed target domain is still subject to significant distribution shift. In short, data imputation has very limited benefit under the domain corruption setting. Furthermore, in scenarios where an attribute is completely missing for all instances (e.g., CL= 1 in IHDP), it is infeasible to impute this attribute for prediction.

6.3 Computational Efficiency of VEGAN

One core motivation of utilizing GAN to replace the straightforward KL divergence optimization is to preserve training efficiency with high dimensionality. Hence, we further test VEGAN's efficiency by comparing its training time (in seconds) consumption per 100 epochs with CEVAE, which is the based method VEGAN improves upon. The test is performed on 12th Gen Intel i7-12700K 12-Cores 20-Threads CPU on Ubuntu 22.04 LTS. Figure 3 shows that as a GAN-based alternative to CEVAE, VEGAN_{I}, has significantly faster training speed (over 6× speedup). Furthermore, the introduction of our second-stage adaption in VEGAN_{II} is still able to maintain high computational efficiency, witnessed by over 4× speedup over CEVAE.

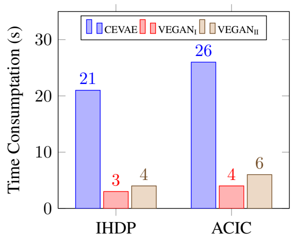

Figure 3: Computation Time.

7 Conclusion

This paper formalizes the runtime causal inference problem under domain corruption, where novel strategies are proposed to counter the imbalance between treated and control groups and the inter-domain discrepancy between training and inference domain. We further adopt adversarial learning to implicitly reduce the distribution disparity for the sake of efficiency. For our proposed approach VEGAN, its performance exceeds other state-of-the-arts under the runtime domain corruption setting in semi-synthetic and full-synthetic benchmark datasets. In addition, the second stage adversarial plug-in is demonstrated applicable to off-the-shelf models to reduce generalization errors.

References

ACIC. Atlantic causal inference conference 2019, Aug 2019. URL https://sites.google.com/view/acic2019datachallenge/data-challenge?authuser=0.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Jeroen Berrevoets, Fergus Imrie, Trent Kyono, James Jordon, and Mihaela van der Schaar. To impute or not to impute?–missing data in treatment effect estimation. *arXiv preprint arXiv:2202.02096*, 2022.

John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. *Advances in neural information processing systems*, 20, 2007.

Marco Caliendo and Sabine Kopeinig. Some practical guidance for the implementation of propensity score matching. *Journal of economic surveys*, 22(1):31–72, 2008.

José M Cordero, Victor Cristobal, and Daniel Santín. Causal inference on education policies: A survey of empirical studies using pisa, timss and pirls. *Journal of Economic Surveys*, 32(3): 878–915, 2018.

Amanda Coston, Edward Kennedy, and Alexandra Chouldechova. Counterfactual predictions under runtime confounding. *Advances in Neural Information Processing Systems*, 33:4150–4162, 2020.

Thomas A Glass, Steven N Goodman, Miguel A Hernán, and Jonathan M Samet. Causal inference in public health. *Annual review of public health*, 34:61, 2013.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Zvi Griliches and Jerry A Hausman. Errors in variables in panel data. *Journal of econometrics*, 31 (1):93–118, 1986.

Jennifer L Hill. Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240, 2011.

Guido W Imbens and Donald B Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.

Andrew Jesson, Sören Mindermann, Uri Shalit, and Yarin Gal. Identifying causal-effect inference failure with uncertainty-aware models. *Advances in Neural Information Processing Systems*, 33: 11637–11649, 2020.

Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In *International conference on machine learning*, pp. 3020–3029. PMLR, 2016.

Fredrik D Johansson, Uri Shalit, Nathan Kallus, and David Sontag. Generalization bounds and representation learning for estimation of potential outcomes and causal effects. *arXiv preprint arXiv:2001.07426*, 2020.

Nathan Kallus. Deepmatch: Balancing deep covariate representations for causal inference using adversarial training. In *International Conference on Machine Learning*, pp. 5067–5077. PMLR, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Zachary Lipton, Julian McAuley, and Alexandra Chouldechova. Does mitigating ml's impact disparity require treatment disparity? *Advances in neural information processing systems*, 31, 2018.

Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal effect inference with deep latent-variable models. *Advances in neural information processing systems*, 30, 2017.

GS Maddala and M Nimalendran. 17 errors-in-variables problems in financial models. *Handbook of statistics*, 14:507–528, 1996.

Imke Mayer, Erik Sverdrup, Tobias Gauss, Jean-Denis Moyer, Stefan Wager, and Julie Josse. Doubly robust treatment effect estimation with missing attributes. *The Annals of Applied Statistics*, 14(3): 1409–1431, 2020.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *International Conference on Machine Learning*, pp. 2391–2400. PMLR, 2017.

Serafín Moral, Andrés Cano, and Manuel Gómez-Olmedo. Computation of kullback–leibler divergence in bayesian networks. *Entropy*, 23(9):1122, 2021.

Rani Nelken and Stuart Merrill Shieber. Computing the kullback-leibler divergence between probabilistic automata using rational kernels. *Harvard Computer Science Group Technical Report TR-07-06*, 2006.

Splawa Neyman. On the application of probability theory to agricultural experiments. essay on principles. section 9. portions translated into english by d. dabrowska and t. speed (1990). *Statistical Science*, pp. 465–472, 1923.

Frank Nielsen. On the jensen–shannon symmetrization of distances relying on abstract means. *Entropy*, 21(5), 2019. ISSN 1099-4300. doi: 10.3390/e21050485.

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. *Advances in neural information processing systems*, 32, 2019.

David CW Phang, Kanliang Wang, Qiuhong Wang, Robert J Kauffman, and Maurizio Naldi. How to derive causal insights for digital commerce in china? a research commentary on computational social science methods. *Electronic Commerce Research and Applications*, 35:100837, 2019.

Yongming Qu and Ilya Lipkovich. Propensity score estimation with missing values using a multiple imputation missingness pattern (mimp) approach. *Statistics in medicine*, 28(9):1402–1414, 2009.

Paul R Rosenbaum. Overt bias in observational studies. In *Observational studies*, pp. 71–104. Springer, 2002.

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

Uri Shalit, Fredrik D Johansson, and David Sontag. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*, pp. 3076–3085. PMLR, 2017.

Claudia Shi, David Blei, and Victor Veitch. Adapting neural networks for the estimation of treatment effects. *Advances in neural information processing systems*, 32, 2019.

Joshua Simons, Sophia Adams Bhatti, and Adrian Weller. Machine learning and the meaning of equal treatment. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 956–966, 2021.

Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

SS Vallender. Calculation of the wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications*, 18(4):784–786, 1974.

Stef Van Buuren and Catharina GM Oudshoorn. Multivariate imputation by chained equations, 2000.

Liuyi Yao, Sheng Li, Yaliang Li, Mengdi Huai, Jing Gao, and Aidong Zhang. Representation learning for treatment effect estimation from observational data. *Advances in Neural Information Processing Systems*, 31, 2018.

Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, and Aidong Zhang. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–46, 2021.

Jinsung Yoon, James Jordon, and Mihaela Van Der Schaar. Ganite: Estimation of individualized treatment effects using generative adversarial nets. In *International Conference on Learning Representations*, 2018.

8 Appendix

8.1 Proof of Theorem 1

To proof **Theorem** 1, we need some preliminary definitions and the proof of **Lemma** 8.1 beforehand.

Definition 8.1 *Let $\mathcal{F}$ be a family of functions, we have $f : \mathcal{Z} \to \mathbb{R}, f \in \mathcal{F}$.* The distribution distance measure - integral probability metric (IPM) between two distribution $\mathbb{P}^{sr}$ and $\mathbb{P}^{tr}$ is defined as:

$$\text{IPM}_{\mathcal{F}}(\mathbb{P}^{tr}, \mathbb{P}^{sr}) = \sup_{f \in \mathcal{F}} \left| \int_{\mathcal{Z}} f(\mathbf{z})(p^{tr}(\mathbf{z}) - p^{sr}(\mathbf{z}))d\mathbf{z} \right|. \tag{15}$$

Definition 8.2 *Let $\Psi : \mathcal{X} \times \{0, 1\} \to \mathbb{R}$ be the hypothesis, the estimated individual treatment effect for unit **x** is:*

$$\hat{\tau}(\mathbf{x}) = \Psi_1(\mathbf{x}, 1) - \Psi_0(\mathbf{x}, 0). \tag{16}$$

Here we slightly extend the expected PEHE defined by Shalit et al. (2017) by replacing the squared loss of causal effect estimation with the expected squared loss defined in **Definition** 8.3 for the proof's sake.

Definition 8.3 *Let $\phi : \mathcal{X} \to \mathcal{Z}$ be the inevitable latent representation mapping function with inverse Φ. Let $\Psi : \mathcal{Z} \times \{0, 1\} \to \mathbb{R}$ be the updated hypothesis. The expected squared loss $l_{\phi,\Psi}(\mathbf{x})$ of inference models is defined as:*

$$\mathbb{E}(L(\hat{\tau}(\mathbf{x}), \tau(\mathbf{x}))) = \int_{\mathcal{X}} L(\hat{\tau}(\mathbf{x}), \tau(\mathbf{x}))p(\mathbf{x})d\mathbf{x}, \tag{17}$$

where $\tau(\boldsymbol{x})$ is the true treatment effect defined in Eq. 1 and $L(\hat{\tau}, \tau)$ is the squared loss.

Definition 8.4 *The expected Precision in Estimation of Heterogeneous Effect (PEHE) of ϕ and Ψ with expected loss function on unit **x** is defined as:*

$$\epsilon_{\text{PEHE}}(\phi, \Psi) = \int_{\mathcal{X}} l_{\phi,\Psi}(\mathbf{x})p(\mathbf{x})d\mathbf{x}, \tag{18}$$

*where $l_{\phi,\Psi}(\mathbf{x})$ is defined in **Definition** 8.3.*

Lemma 8.1: *Let $\phi : \mathcal{X} \to \mathcal{Z}$ be the invertible latent representation mapping function with inverse Φ. Let $\Psi : \mathcal{Z} \times \{0, 1\} \to \mathbb{R}$ be the updated hypothesis. Let $\mathcal{F}$ be a family of functions $f : \mathcal{Z} \to$*

$\mathbb{R}, f \in \mathcal{F}$. Assume we have $B_\phi > 0$ s.t. $\frac{1}{B_\phi} l_{\phi,\Psi}(\Phi(z)) \in \mathcal{F}$. The tightness of target domain error w.r.t. the source domain one is bounded by the distribution distance denoted by IPM:

$$\begin{aligned} \left|\epsilon_{\text{PEHE}}^{tr} - \epsilon_{\text{PEHE}}^{sr}\right| &= \left|B_\phi \int_{\mathcal{Z}} \frac{1}{B_\phi} l_{\phi,\Psi}(\Phi(\mathbf{z}))(p_\phi^{tr}(\mathbf{z}) - p_\phi^{sr}(\mathbf{z}))d\mathbf{z}\right| \\ &\leq B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr}), \end{aligned} \tag{19}$$

where ϵ_{PEHE}^{tr} and ϵ_{PEHE}^{sr} indicate target domain error and source domain error respectively, $\mathbb{P}_\phi^{tr}$ and $\mathbb{P}_\phi^{sr}$ denote covariate distribution in target domain and source domain respectively, and B_ϕ is a bounded constant.

We denote the expected PEHE in Eq. 18 in target domain and source domain as $\epsilon_{\text{PEHE}}^{tr}$ and $\epsilon_{\text{PEHE}}^{sr}$ respectively, also tr and sr indicates the test set and training set in our main text where $p^{tr}(\mathbf{x}) \neq p^{sr}(\mathbf{x})$ if domain corruption exists. We firstly present the proof for **Lemma** 8.1 as follows:

Proof.

$$\begin{aligned} \left|\epsilon_{\text{PEHE}}^{tr} - \epsilon_{\text{PEHE}}^{sr}\right| &= \left|\int_{\mathcal{X}} l_{\phi,\Psi}(\mathbf{x}) p_\phi^{tr}(\mathbf{x}) d\mathbf{x} - \int_{\mathcal{X}} l_{\phi,\Psi}(\mathbf{x}) p_\phi^{sr}(\mathbf{x}) d\mathbf{x}\right| \\ &= \left|\int_{\mathcal{Z}} l_{\phi,\Psi}(\Phi(\mathbf{z})) p_\phi^{tr}(\mathbf{z}) d\mathbf{z} - \int_{\mathcal{Z}} l_{\phi,\Psi}(\Phi(\mathbf{z})) p_\phi^{sr}(\mathbf{z}) d\mathbf{z}\right| \\ &= \left|B_\phi \int_{\mathcal{Z}} \frac{1}{B_\phi} l_{\phi,\Psi}(\Phi(\mathbf{z}))(p_\phi^{tr}(\mathbf{z}) - p_\phi^{sr}(\mathbf{z}))d\mathbf{z}\right| \\ &\leq \left|B_\phi \sup_{f \in \mathcal{F}} \left|\int_{\mathcal{Z}} f(\mathbf{z})(p_\phi^{tr}(\mathbf{z}) - p_\phi^{sr}(\mathbf{z}))d\mathbf{z}\right|\right| \\ &= \left|B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr})\right| = B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr}). \end{aligned} \tag{20}$$

The first equality is by **Definition** 8.4, the second equality is by change of variable, the first inequality is by the premise that $\frac{1}{B_\phi} l_{\phi,\Psi}$ belongs to the function family $\mathcal{F}$, the fourth equality is by **Definition** 8.1, the last equality is by the property that IPM is non-negative. □

Theorem 8.1 *Let $\phi : \mathcal{X} \to \mathcal{Z}$ be the invertible latent representation mapping function with inverse Φ. Let $\Psi : \mathcal{Z} \times \{0,1\} \to \mathbb{R}$ be the hypothesis. Let $\mathcal{F}$ be a family of functions $f : \mathcal{Z} \to \mathbb{R}, f \in \mathcal{F}$. Assume we have $B_\phi > 0$ s.t. $\frac{1}{B_\phi} l_{\phi,\Psi}(\Phi(z), t) \in \mathcal{F}$ (Shalit et al., 2017), we have:*

$$\epsilon_{\text{PEHE}}^{sr} \leq 2(\epsilon_{\text{F}}^{t=1} + \epsilon_{\text{F}}^{t=0} + B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{t=1}, \mathbb{P}_\phi^{t=0}) - 2\sigma_{y^*}^2), \tag{21}$$

where ϵ_F^t denotes the factual training error within treatment group t, $\mathbb{P}_\phi^t$ is probability measure for treatment group t, and $\sigma_{y^*}^2$ is the variance of potential outcome.

Assumption 8.1 Stable Unit Treatment Value Assumption (SUTVA): *a) The potential outcomes for any unit do not vary with the treatment assigned to other units. b) For each unit, there are no different forms or versions of each treatment level, which lead to different potential outcomes (Imbens & Rubin, 2015).*

Proof. Under the conditions of unconfoundedness assumption $t \perp\!\!\!\perp \{y_0^*, y_1^*\}|\phi(\mathbf{x})$ and STUVA, the potential outcome y_t^* is determined by $\mathbf{x}$ via function g, namely $y_t^* = g(\mathbf{x})$, resulting the single unit expected squared loss equals the squared loss, and thus the expected PEHE defined in **Definition** 8.4 equals the one defined by Shalit et al. (2017), it also results in the variance $\sigma_{y^*}^2 = 0$ in Eq. 21.

To conclude, the **Theorem** 1 is thus proofed under the further conditions of **Lemma** 8.1 and **Theorem** 8.1 as follows:

$$\begin{aligned} \epsilon_{\text{PEHE}}^{tr} &\leq \epsilon_{\text{PEHE}}^{sr} + B_\phi \text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr}) \\ &\leq 2\left[\epsilon_{\text{F}}^{t=1} + \epsilon_{\text{F}}^{t=0} + B_\phi \left(\text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{t=1}, \mathbb{P}_\phi^{t=0}) + \frac{1}{2}\text{IPM}_{\mathcal{F}}(\mathbb{P}_\phi^{tr}, \mathbb{P}_\phi^{sr})\right)\right]. \end{aligned} \tag{22}$$

□

We align the function family $\mathcal{F}$ to the one Shalit et al. (2017) use as different choice of function family $\mathcal{F}$ will require different assumption about the joint distribution $p(\mathbf{z}, t, y_1, y_0)$, the representation mapping function ϕ, and the hypothesis Ψ. Thus, we share the same bounded constant B_ϕ.

8.2 Proof of Evidence Lower Bound

We present the reasoning procedures for our evidence lower bound for $\log p(y|t)$ with Ψ_t and ϕ_t implemented for approximation as follows:

Proof.

$$\begin{aligned}
\log p_{\Psi_t}(y|t) &= \log \int_{\mathbb{P}_{\phi_t}} p_{\Psi_t}(y, \mathbf{z}|t) d\mathbf{z} = \log \int_{\mathbb{P}_{\phi_t}} \frac{p_{\Psi_t}(y, \mathbf{z}|t)}{p_{\phi_t}(\mathbf{z}|\mathbf{x})} p_{\phi_t}(\mathbf{z}|\mathbf{x}) d\mathbf{z} \\
&= \log \mathbb{E}_{\mathbb{P}_{\phi_t}} [\frac{p_{\Psi_t}(y, \mathbf{z}|t)}{p_{\phi_t}(\mathbf{z}|\mathbf{x})}] \\
&\geq \mathbb{E}_{\mathbb{P}_{\phi_t}} [\log \frac{p_{\Psi_t}(y, \mathbf{z}|t)}{p_{\phi_t}(\mathbf{z}|\mathbf{x})}] \quad \text{Jensen's Inequality} \\
&= \int_{\mathbb{P}_{\phi_t}} p_{\phi_t}(\mathbf{z}|\mathbf{x}) \log \frac{p_{\Psi_t}(y, \mathbf{z}|t)}{p_{\phi_t}(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\
&= \int_{\mathbb{P}_{\phi_t}} p_{\phi_t}(\mathbf{z}|\mathbf{x}) \log \frac{p_{\Psi_t}(y|\mathbf{z}, t) p_{\Psi_t}(\mathbf{z}|t)}{p_{\phi_t}(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\
&= \mathbb{E}_{\mathbb{P}_{\phi_t}} [\log p_{\Psi_t}(y|\mathbf{z}, t)] - \mathbb{E}_{\mathbb{P}_{\phi_t}} [\log \frac{p_{\phi_t}(\mathbf{z}|\mathbf{x})}{p_{\Psi_t}(\mathbf{z}|t)}] \\
&= \mathbb{E}_{\mathbb{P}_{\phi_t}} [\log p_{\Psi_t}(y|\mathbf{z}, t)] - D_{\text{KL}}(\mathbb{P}_{\phi_t} || \mathbb{P}_{\Psi_t})].
\end{aligned} \tag{23}$$

The conditional prior $p_{\Psi_t}(\mathbf{z}|t)$ is actually a marginal prior $p_{\Psi_t}(\mathbf{z})$. Because for each treatment group, treatment t is just a constant for all units within that group, e.g., $p_{\Psi_1}(\mathbf{z}|t = 1) = p_{\Psi_1}(\mathbf{z})$. □

8.3 Reparameterization

All the neural networks' parameters can be directly differentiated but the shared feature extractor G_ϕ's due to the variational nature of the model. For instance, to deal with the gradient from discriminator D_δ's output:

$$\nabla_\phi [\mathbb{E}_{\mathcal{N}} [\log D_\delta(\mathbf{n})] + \mathbb{E}_{\mathcal{Z}^{sr}} [\log(1 - D_\delta(\mathbf{z}))]] \neq 0 + \mathbb{E}_{\mathcal{Z}^{sr}} [\nabla_\phi \log(1 - D_\delta(\mathbf{z}))]. \tag{24}$$

Because $\mathbf{z}^{sr}$ is drawn from a distribution parameterized by ϕ. Thus, a way out is to use the "reparameterization trick" (Kingma & Welling, 2013). The methodology is to let the generating of $\mathbf{z}$ be dependent of something which is independent from ϕ, preferably in the form $\mathbf{z} = \omega(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}^2_\phi, \boldsymbol{\epsilon})$. According to Eq. 9, $\mathbf{z}$ follows the normal distribution $\mathcal{N}(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}^2_\phi)$ parameterized by the neural nets G_ϕ, thus it can be rephrased as $\mathbf{z} = \boldsymbol{\mu}_\phi + \boldsymbol{\sigma}^2_\phi \odot \boldsymbol{\epsilon}$, where $\odot$ denotes the element-wise product and $\boldsymbol{\epsilon}$ follows an independent standard normal distribution. Thus,

$$\begin{aligned}
\nabla_\phi \mathbb{E}_{\mathcal{Z}^{sr}} [\log(1 - D_\delta(\mathbf{z}))] &= \nabla_\phi \mathbb{E}_\epsilon [\log(1 - D_\delta(\omega(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}^2_\phi, \boldsymbol{\epsilon})))] \\
&= \mathbb{E}_\epsilon [\nabla_\phi \log(1 - D_\delta(\omega(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}^2_\phi, \boldsymbol{\epsilon})))].
\end{aligned} \tag{25}$$

Due to the reparameterization $\mathbf{z} = \omega(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}^2_\phi, \boldsymbol{\epsilon})$, the expectation notation changed accordingly. For instance:

$$\nabla_\phi \mathbb{E}_{\mathcal{Z}^{sr}} [\log p_{\Psi_t}(y|\mathbf{z}, t)] = \mathbb{E}_\epsilon [\nabla_\phi \log p_{\Psi_t}(y|\omega(\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}^2_\phi, \boldsymbol{\epsilon}), t)]. \tag{26}$$

8.4 Domain Corruption Simulation

Our domain corruption is built one step above the conventional domain shift/covariate shift, thus it not only considers the shift in the covariate distribution but the missing value found in the spreadsheet. Another intuitive example for our defined domain corruption scenario is that respondents

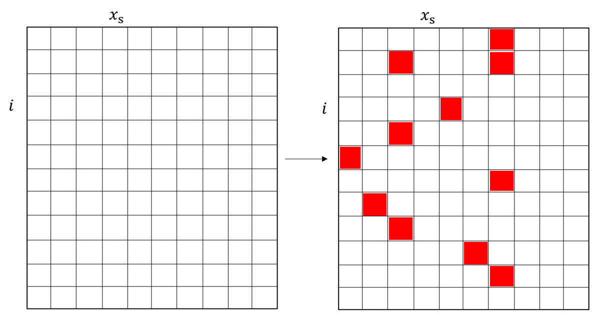

Figure 4: Randomly held-out dataset shifts from left to the corrupted one. The red square suggests the corrupted value, which can be either shifted value or missing value in the test set.

are reluctant to give privacy-related information themselves when responding to a data-collection survey, e.g., falsely report their age or just leave it blank, thus creating domain corruption in the collected dataset compared to the fine-grained one we possess in the database.

8.4.1 Generating Test Set

Each of the dataset is randomly split with 3:1 ratio for train/test set, then we corrupt the dataset to simulate the domain corruption scenario following the rule: For a given domain corruption level (CL), as CL $\in \{\frac{1}{20}, \frac{1}{8}, \frac{1}{5}, \frac{1}{3}, 1\}$, and chosen feature $x_s \in \mathbf{x}_i$, we firstly add a random variation drawn from Gaussian distribution $\mathcal{N}(\bar{\mu}, 0.1)$ to x_s with probability $p =$ CL for each individual i if x_s is continuous, or we flip the binary value if x_s is binary, $\bar{\mu}$ follows an uniform distribution $\mathcal{U}$(0, 0.25). Secondly, we randomly zero the value of x_s to mimic the missing value situation with probability $p =$ CL for each individual i. The first step and second step are performed independently, thus making the randomly held-out dataset noisy. Figure 4 depicts our domain corruption mechanism in a two-dimension spreadsheet with row representing unit i and column representing feature x_s.

8.4.2 Varying the Magnitude of Added Gaussian Noise

To justify the setting for generating drifted data, the Gaussian noise we used follows a normal distribution $\mathcal{N}(\bar{\mu}, \bar{\sigma})$. For each test instance, the random noise has different mean $\bar{\mu}$ as it is drawn from a uniform distribution. On the other hand, we provide additional experiments by varying the upper bound of the uniform distribution for $\bar{\mu}$ (e.g., $\bar{\mu}_1 \sim \mathcal{U}(0, 0.25)$, $\bar{\mu}_2 \sim \mathcal{U}(0, 1)$, $\bar{\mu}_3 \sim \mathcal{U}(0, 1.5)$) and take different variance $\bar{\sigma} \in \{0.1, 0.25\}$ at the domain corruption level 1/3. The test results with different settings are provided in the following:

Model \ Noise	$\bar{\mu} \sim \mathcal{U}(0, 0.25), \bar{\sigma} = 0.1$	$\bar{\mu} \sim \mathcal{U}(0, 1), \bar{\sigma} = 0.25$	$\bar{\mu} \sim \mathcal{U}(0, 1.5), \bar{\sigma} = 0.25$
TARNet	$0.775 \pm .04$	$0.777 \pm .04$	$0.789 \pm .04$
VEGAN$_{\text{II}}$	$0.395 \pm .02$	$0.443 \pm .02$	$0.498 \pm .04$

Table 5: $\sqrt{\epsilon_{\text{PEHE}}}$ of out-of-sample prediction on ACIC dataset with different noise setting.

Model \ Noise	$\bar{\mu} \sim \mathcal{U}(0, 0.25), \bar{\sigma} = 0.1$	$\bar{\mu} \sim \mathcal{U}(0, 1), \bar{\sigma} = 0.25$	$\bar{\mu} \sim \mathcal{U}(0, 1.5), \bar{\sigma} = 0.25$
TARNet	$0.211 \pm .03$	$0.253 \pm .03$	$0.285 \pm .03$
VEGAN$_{\text{II}}$	$0.231 \pm .02$	$0.230 \pm .02$	$0.265 \pm .02$

Table 6: ϵ_{ATE} of out-of-sample prediction on ACIC dataset with different noise settings.

We compare VEGAN$_{\text{II}}$ with TARNet, as all the other baselines built on top of TARNet are equally the same in terms of lacking the functionality to handle noised test set. When the noise magnitude increases, the prediction error of both methods increase, but VEGAN holds a consistent and significant improvement margin compared with TARNet's results.

8.5 Applicability of Second Stage Adversarial Plug-in to Other Baselines

We build the second stage adversarial plug-in to the most representative model - TARNet. The experiments are conducted using the ACIC dataset, and the results are presented in Tables 7 and 8:

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	$0.813 \pm .05$	$0.738 \pm .04$	$0.798 \pm .05$	$0.754 \pm .04$
TARNet-Plugin	$0.540 \pm .01$	$0.545 \pm .02$	$0.526 \pm .02$	$0.554 \pm .02$

Table 7: $\sqrt{\epsilon_{\text{PEHE}}}$ of out-of-sample prediction on ACIC dataset with different corruption levels.

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	1.45	2.12	4.55	6.34
TARNet-Plugin	2.21	0.29	1.90	2.29

Table 8: Variation $\Delta(\%)$ of the out-of-sample $\sqrt{\epsilon_{\text{PEHE}}}$ w.r.t. the in-sample one on ACIC dataset across increasing domain corruption level.

We denote the TARNet with adversarial plug-in as TARNet-Plugin. As the results suggest, when the corruption level becomes higher, the benefit of the stage 2 domain adaptation will be enlarged. When the this adversarial plug-in is in use, it effectively helps TARNet reduce prediction risks under runtime domain corruption as the volatility of the TARNet-Plugin is stabilized at around 2%.

8.6 Stability Analysis of VEGAN Models

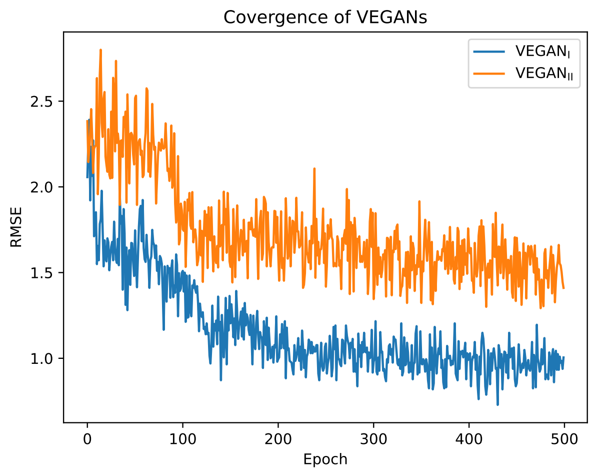

Figure 5: The convergence of VEGAN$_\text{I}$ and VEGAN$_\text{II}$ on ACIC dataset in terms of RMSE.

As GAN is known to be unstable, we provide stability analysis of our proposed VEGAN models in terms of prediction loss convergence and equilibrium status between feature extractor and discriminators. Figure 5 shows the models convergence's on root mean square error (RMSE) during training. It is noted that a higher corruption level brings more challenges in the adversarial training, but we observe an equilibrium state in the majority of the cases. Take the harder 1/3 corruption level in IHDP dataset as an example, both discriminators (for treated/control and training/runtime adaptations) can quickly converge to the equilibrium by returning an average binary cross-entropy loss of 0.69, which means the discriminators are completely decepted by the feature extractors, and always give 0.5 probability for the samples from each of the groups. We occasionally have the situation where one of the discriminators outperforms the feature extractor, predominantly the discriminator for treated/control adaptation. After repeatedly executing 100 runs for each setting, we observe that on average, this happens less frequently than 5% for ACIC dataset and 30% for IHDP dataset. As such, training VEGAN in an adversarial setting is completely attainable.

8.7 Baselines Enhanced with Imputation

To compare imputation-based methods with VEGAN, we using the MICE algorithm (Van Buuren & Oudshoorn, 2000) adopted by Mayer et al. (2020); Berrevoets et al. (2022) to impute the missing values in the test set. Notably, when the corruption level (i.e., missing rate) reaches 1/3, the MICE algorithm melts down, and we then turn to KNN-based imputation (Troyanskaya et al., 2001) from the corruption level of 1/3. We provide the experiment results on the ACIC dataset with imputation-enhanced baselines (noted with IMP) in Table 9 and 10:

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	$0.813 \pm .05$	$0.738 \pm .04$	$0.798 \pm .05$	$0.754 \pm .04$
TARNet-IMP	$0.807 \pm .04$	$0.728 \pm .04$	$0.783 \pm .04$	$0.763 \pm .04$
CFR$_{\text{WASS}}$	$0.730 \pm .04$	$0.655 \pm .03$	$0.738 \pm .04$	$0.644 \pm .04$
CFR$_{\text{WASS}}$-IMP	$0.722 \pm .04$	$0.640 \pm .03$	$0.710 \pm .04$	$0.679 \pm .04$
SITE	$1.482 \pm .08$	$1.361 \pm .08$	$1.365 \pm .09$	$1.586 \pm .10$
SITE-IMP	$1.322 \pm .08$	$1.210 \pm .07$	$1.358 \pm .08$	$1.268 \pm .08$
Dragonnet$_{\text{Base}}$	$2.250 \pm .02$	$2.192 \pm .02$	$2.142 \pm .02$	$2.063 \pm .02$
Dragonnet$_{\text{Base}}$-IMP	$2.232 \pm .02$	$2.162 \pm .02$	$2.097 \pm .02$	$2.117 \pm .02$
Dragonnet$_{\text{TR}}$	$2.517 \pm .02$	$2.456 \pm .02$	$2.406 \pm .02$	$2.342 \pm .02$
Dragonnet$_{\text{TR}}$-IMP	$2.485 \pm .02$	$2.407 \pm .02$	$2.343 \pm .02$	$2.393 \pm .02$
CEVAE	$0.646 \pm .02$	$0.629 \pm .02$	$0.594 \pm .02$	$0.581 \pm .03$
CEVAE-IMP	$0.626 \pm .02$	$0.601 \pm .02$	$0.577 \pm .02$	$0.568 \pm .03$
VEGAN$_{\text{I}}$	$\mathbf{0.466 \pm .01}$	$\mathbf{0.489 \pm .01}$	$0.488 \pm .01$	$0.501 \pm .01$
VEGAN$_{\text{II}}$	$0.490 \pm .01$	$0.493 \pm .01$	$\mathbf{0.471 \pm .01}$	$\mathbf{0.455 \pm .00}$

Table 9: $\sqrt{\epsilon_{\text{PEHE}}}$ of out-of-sample prediction on ACIC dataset with different corruption levels.

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	1.45	2.12	4.55	6.34
TARNet-IMP	2.18	3.45	6.34	7.06
CFR$_{\text{WASS}}$	0.41	0.61	2.38	3.45
CFR$_{\text{WASS}}$-IMP	1.5	2.88	6.08	6.86
SITE	3.49	3.03	6.47	10.68
SITE-IMP	2.44	4.42	5.96	9.37
Dragonnet$_{\text{Base}}$	**0.09**	2.23	4.59	7.74
Dragonnet$_{\text{Base}}$-IMP	0.49	3.65	6.38	5.70
Dragonnet$_{\text{TR}}$	0.64	1.76	3.76	6.17
Dragonnet$_{\text{TR}}$-IMP	0.44	3.41	5.87	4.62
CEVAE	7.71	10.01	12.00	16.76
CEVAE-IMP	8.21	9.24	11.57	14.82
VEGAN$_{\text{I}}$	0.64	**0.81**	0.61	2.72
VEGAN$_{\text{II}}$	0.20	1.60	**0.42**	**0.22**

Table 10: Variation $\Delta(\%)$ of the out-of-sample $\sqrt{\epsilon_{\text{PEHE}}}$ w.r.t. the in-sample one on ACIC dataset across increasing domain corruption level.

As shown in the tables, when the corruption rate is low, using imputation is generally helpful for slightly reduce the $\sqrt{\epsilon_{\text{PEHE}}}$, but may provide negative impact when more attributes are corrupted (e.g., at a rate of 1/5 and 1/3). At the same time, imputation methods do not make a difference in terms of volatility, as the imputed target domain is still subject to significant distribution shift w.r.t. the training domain, and the volatility of the model increases rapidly as corruption level goes up. To conclude, data imputation has very limited practicality under the domain corruption setting. Furthermore, in scenarios where an attribute is completely missing for all instances (e.g., medical checks that are unavailable in some hospitals, which resembles the rightmost column in Table 2), it is infeasible to impute this attribute for prediction.

8.8 Selected Sensitive Features in IHDP

We selectively choose 7 features which can be considered as sensitive and thus highly possible to be corrupted: momage, sex, twin, b.marr, cig, drugs, and work_dur. The explanations are cited from Hill (2011) as follows:

feature	Explanation
momage	The mother's age when she gave birth to the baby
sex	Baby's sex.
twin	Is the baby twin?
b.marr	Is the mother married when giving birth to the baby?
cig	Does the mother smoke during pregnancy?
drugs	Does the mother take drugs during pregnancy?
work_dur	Does the mother have work during pregnancy?

Table 11: Selected sensitive features and their explanations

8.9 VEGAN Architecture

We use the general structure for evaluating VEGAN$_{\text{I}}$ and VEGAN$_{\text{II}}$ on both IHDP and ACIC benchmarks. G_ϕ is used to represent the whole feature extractor for notation simplicity which has two-headed output with each of them modelling the mean and variance of the latent representation respectively by multi-layer perceptrons (MLPs).

Module	Hidden Layers	Hidden Neurons per Hidden Layer	Output
G_ϕ	3	100	Two-headed output
Ψ_1	2	200	Scalar
Ψ_0	2	200	Scalar
D_δ	2	100	Scalar
D_β	2	100	Scalar

Table 12: Modules' details of VEGAN

For other hyperparameters, we set epochs = 100, convergence tolerance = 10, batch size = 100, model complexity L2 regularization = 0.01, learning rate = 0.001.

8.10 Other Results

It is worth noting that 0 CL results are omitted as it resembles the in-sample predictions in Tables 1 and 3. We leave the CATE estimation errors for IHDP and ACIC in Table 13 and Table 14 respectively, and the variation $\Delta(\%)$ on ACIC in Table 15.

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$	1
TARNet	0.359	0.559	0.762	1.383	3.784
CFR$_{\text{WASS}}$	0.349	0.546	0.735	1.342	3.009
SITE	0.362	0.513	0.825	1.440	3.773
Dragonnet$_{\text{Base}}$	0.344	**0.269**	0.417	0.909	2.418
Dragonnet$_{\text{TR}}$	0.308	0.295	0.487	1.017	2.752
CEVAE	0.355	0.478	**0.290**	**0.608**	1.035
VEGAN$_{\text{I}}$	**0.245**	0.313	0.450	0.905	2.734
VEGAN$_{\text{II}}$	0.258	0.290	0.391	0.709	**0.679**

Table 13: ϵ_{CATE} on out-of-sample IHDP dataset with different domain corruption level on privacy-related features.

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	0.201	0.185	0.225	0.184
CFR$_{\text{WASS}}$	0.223	0.207	0.228	0.204
SITE	0.328	0.307	0.315	0.342
Dragonnet$_{\text{Base}}$	0.199	0.198	0.223	0.209
Dragonnet$_{\text{TR}}$	0.196	0.213	0.247	0.218
CEVAE	0.293	0.279	0.280	0.231
VEGAN$_{\text{I}}$	**0.134**	**0.148**	0.147	0.157
VEGAN$_{\text{II}}$	0.149	0.157	**0.145**	**0.139**

Table 14: ϵ_{CATE} on out-of-sample ACIC dataset with different domain corruption level on all features.

Model \ CL	$\frac{1}{20}$	$\frac{1}{8}$	$\frac{1}{5}$	$\frac{1}{3}$
TARNet	1.45	2.12	4.55	6.34
CFR$_{\text{WASS}}$	0.41	0.61	2.38	3.45
SITE	3.49	3.03	6.47	10.68
Dragonnet$_{\text{Base}}$	**0.09**	2.23	4.59	7.74
Dragonnet$_{\text{TR}}$	0.64	1.76	3.76	6.17
CEVAE	7.71	10.01	12.00	16.76
VEGAN$_{\text{I}}$	0.64	**0.81**	0.61	2.72
VEGAN$_{\text{II}}$	0.20	1.60	**0.42**	**0.22**

Table 15: Variation $\Delta(\%)$ of the out-of-sample $\sqrt{\epsilon_{\text{PEHE}}}$ w.r.t. the in-sample one on ACIC dataset across increasing domain corruption level.

8.11 Optimization Scheme of VEGAN

Algorithm 1 Optimization of VEGAN

```
1: Input: Train set $\{(\mathbf{x}_i^{sr}, t_i^{sr}, y_i^{sr})\}_{i=1}^{N}$, runtime variable set $\{(\mathbf{x}_j^{tr})\}_{j=1}^{N'}$ (optional), hyperparameters (e.g., learning rate $\alpha$), initialized neural networks' parameters $\{\phi, \Psi_1, \Psi_0, \delta, \beta\}$;
2: while not converged do
3:     Sample a mini-batch $b_m \in \{b_m\}_{m=1}^{M}$ of size $|b_m| = |b_m^{t=1}| + |b_m^{t=0}|$, with $|b_m^{t=1}| = |b_m^{t=0}|$, from $\{(\mathbf{x}_i^{sr}, t_i^{sr}, y_i^{sr})\}_{i=1}^{N}$. Sample equal size instances from standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{1})$;
4:     $\delta \leftarrow \delta - \alpha \frac{1}{|b_m|} \Big( \sum_{j=1}^{|b_m|} \nabla_\delta \log D_\delta(\mathbf{n}_j) + \sum_{i=1}^{|b_m|} \nabla_\delta \log(1 - D_\delta(\omega_i^{sr})) \Big)$;
5:     if runtime domain corruption exists then
6:         Randomly draw batch $b'_m$ from $\{(\mathbf{x}_j^{tr})\}_{j=1}^{N'}$ with size $|b'_m| = |b_m|$;
7:         $\beta \leftarrow \beta - \alpha \frac{1}{|b_m|} \Big( \sum_{i=1}^{|b_m|} \nabla_\beta \log D_\beta(\omega_i^{sr}) + \sum_{i=1}^{|b_m|} \nabla_\beta \log(1 - D_\beta(\omega_i^{tr})) \Big)$;
8:     end if
9:     $\phi \leftarrow \phi - \alpha \Big( \frac{1}{|b_m|} \sum_{i=1}^{|b_m|} \nabla_\phi \log D_\delta(\omega_i^{sr}) + \frac{1}{|b_m|} \sum_{i=1}^{|b_m|} \nabla_\phi \log(1 - D_\beta(\omega_i^{sr}))$
           $+ \frac{1}{|b_m|} \sum_{i=1}^{|b_m|} \nabla_\phi \log D_\beta(\omega_i^{tr}) + \sum_{t \in \{0,1\}} \frac{1}{|b_m^t|} \sum_{i=1}^{|b_m^t|} \nabla_\phi \log p_{\Psi_t}(y_i|\omega_i, t_i) \Big)$,
       $\Psi_1 \leftarrow \Psi_1 - \frac{\alpha}{|b_m^{t=1}|} \sum_{i=1}^{|b_m^{t=1}|} \nabla_{\Psi_1} \log p_{\Psi_1}(y_i|\omega_i, t_i = 1)$,
       $\Psi_0 \leftarrow \Psi_0 - \frac{\alpha}{|b_m^{t=0}|} \sum_{i=1}^{|b_m^{t=0}|} \nabla_{\Psi_0} \log p_{\Psi_0}(y_i|\omega_i, t_i = 0)$;
10: end while
```