

Article

# An Out-of-Distribution Generalization Framework Based on Variational Backdoor Adjustment

Hang Su \*  and Wei Wang

School of Mathematics, Renmin University of China, Beijing 100872, China; wwei@ruc.edu.cn

\* Correspondence: hangsu@ruc.edu.cn

**Abstract:** In practical applications, learning models that can perform well even when the data distribution is different from the training set are essential and meaningful. Such problems are often referred to as out-of-distribution (OOD) generalization problems. In this paper, we propose a method for OOD generalization based on causal inference. Unlike the prevalent OOD generalization methods, our approach does not require the environment labels associated with the data in the training set. We analyze the causes of distributional shifts in data from a causal modeling perspective and then propose a backdoor adjustment method based on variational inference. Finally, we constructed a unique network structure to simulate the variational inference process. The proposed variational backdoor adjustment (VBA) framework can be combined with any mainstream backbone network. In addition to theoretical derivation, we conduct experiments on different datasets to demonstrate that our method performs well in prediction accuracy and generalization gaps. Furthermore, by comparing the VBA framework with other mainstream OOD methods, we show that VBA performs better than mainstream methods.

**Keywords:** out-of-distribution generalization; causal inference; machine learning; backdoor adjustment; supervised learning

**MSC:** 68T01; 68T07



**Citation:** Su, H.; Wang, W. An Out-of-Distribution Generalization Framework Based on Variational Backdoor Adjustment. *Mathematics* **2024**, *12*, 85. <https://doi.org/10.3390/math12010085>

Academic Editors: Debo Cheng, Junbo Ma and Rongyao Hu

Received: 9 December 2023

Revised: 23 December 2023

Accepted: 25 December 2023

Published: 26 December 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Traditional machine learning algorithms often achieve satisfactory results only when the data adheres to the assumption of being independently and identically distributed (i.i.d.) [1,2]. However, the traditional machine learning algorithms that optimize empirical risk often exhibit poor generalization performance due to distributional shifts in real-world data stemming from potential heterogeneity or selection biases. Ensuring robust out-of-distribution (OOD) generalization capabilities and stability in the face of distribution shifts is of paramount importance [2,3], particularly in domains such as financial forecasting, medical diagnosis, autonomous driving, and others. In the next paragraph, we provide a simple example illustrating the challenges traditional machine learning algorithms face when there is a shift in data distribution.

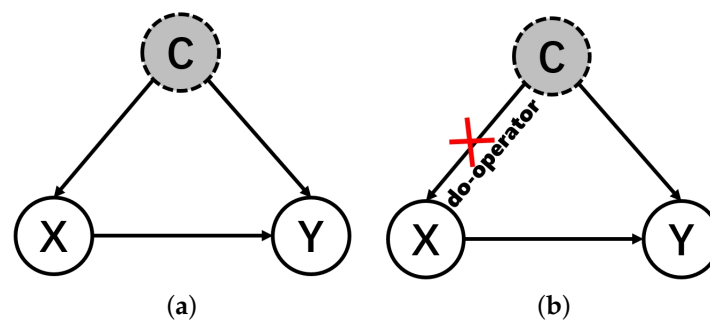
Consider the problem of classifying images of cows and camels [1,4]. As we all know, most cattle are on the grassland, while most camels are in the desert. This phenomenon introduces selection bias, causing the trained model to rely on spurious correlations between the environment and the animals. Therefore, after training on this dataset, the model fails to correctly classify simple examples of cow images when they are taken on sandy beaches. This failure is because the model exploits an easily detectable spurious relationship between animal categories and background colors during training to reduce training error. In summary, traditional methods based on Empirical Risk Minimization can lead to significant errors when confronted with out-of-distribution data.

To address this issue, we need to distinguish between causal features (animal shapes) and spurious correlated (background colors) features related to the category [1,5,6]. The

reason that models often rely on spurious correlated features is the presence of unobservable confounders that simultaneously affect both features and categories. Confounders lead to correlations between certain observed features and categories without a causal relationship. The distribution of the confounder  $P(C)$ , along with the conditional probability distributions of the confounder given the observed features  $P(C|X)$  and categorical variables  $P(C|Y)$ , may undergo changes with variations in the environment from which the data are sampled. Therefore, traditional machine learning methods based on Empirical Risk Minimization may fail when the environment in which the training data are collected differs from the testing environment.

Most of the existing out-of-distribution generalization algorithms utilize datasets with multiple training environments to discover invariant features and subsequently use these invariant features for category inference [1,5,7–9]. These methods are called invariant learning algorithms, which assume that the causal relationship does not vary with the environment and aim to learn invariant causal relationships by training in various sampling environments. The invariant learning algorithms require the source data to include labels indicating the data sampling environment. Therefore, their performance is highly dependent on the quality of the environment. Moreover, the requirements for environment labels can sometimes be overly strict. Typically, raw data comprise randomly selected data from multiple environments without explicit environment labels [2,10].

In this paper, we model the causal relationships between observed features, category labels, and unobservable confounders using Structural Causal Models (SCM) [11–13]. See Figure 1a for a causal graph to illustrate the SCM of real-world data.



**Figure 1.** Structural causal model for machine learning. The dashed circle represents unobservable confounders, X represents observed features, and Y represents category labels. (a) SCM of real-world data. (b) Our interventional model.

From the SCM, it can be seen that the confounders simultaneously affect both features and labels. Since the confounders are unobservable, models trained through empirical risk will learn the spurious correlation between some dimensions of X and Y. The spurious correlation will change when the distribution of the confounder varies. This is why most traditional machine learning models fail to generalize to new datasets.

To address the confounding effect of C and endow the model with robustness to the confounder distributional shift, we propose an interventional model that utilizes the *do*-operator to cut off the causal path between the confounder C and the feature X [12,13], as shown in Figure 1b. By cutting off the path from C to X, we simulated an ideal environment where the generation of feature X is not influenced by the confounder C. Therefore, the correlation between the obtained feature X and the label Y is also a causal relationship in this training environment.

The ideal way to introduce the *do*-operator is to conduct new random experiments and collect data again. However, in most practical applications of the algorithm, we can only obtain data sampled randomly from the natural environment rather than collecting data through random experiments. Additionally, randomized experiments are often impractical and can involve significant costs and ethical concerns in many situations. Fortunately, there are backdoor adjustment methods in statistical causal estimation [13]. This method divides

the data based on different values of the confounders and assigns different weights to each partition. It allows the probability distribution after intervention by the *do*-operator to be calculated solely based on observed data. We introduce the variational inference method to compute the distributions involved in backdoor adjustment, which are difficult to calculate directly.

In summary, based on a causal model different from invariant learning, we construct a variational backdoor adjustment (VBA) framework to solve the OOD generalization problem. This framework does not require multiple training environments. Furthermore, experimental results validate that our algorithm outperforms most existing distributional generalization methods.

The main contributions of this paper can be summarized as follows:

- We present a novel and meaningful perspective on the OOD generalization problem. It involves capturing changes in data sampling environments through variations in the distribution of confounders.
- We propose a method for out-of-distribution generalization without environment labels. Our proposed method employs a variational approach to perform backdoor adjustment on features, thereby eliminating the impact of environmental changes on model training.
- We propose a framework for an OOD generalization algorithm that can be combined with any backbone model.

## 2. Related Work

Our work combines two research domains: OOD generalization and causal inference.

**OOD generalization:** The OOD generalization problem has been widely observed in various domains [1,5,6,9,14–18]. To address this issue, researchers have proposed various algorithms from different perspectives, such as distributional robust optimization [19,20] and causal invariant learning [1,5,6,21–26].

Distributional robust optimization methods [19,20] consider OOD data that are close to the training distribution in terms of probability distance or divergence, such as the Wasserstein distance or  $f$ -divergence [9]. They train the model through robust optimization, making the model robust to the distribution of the training set, thus achieving good generalization performance on what is called an uncertainty set. However, the performance of such methods is highly dependent on the choice of a probability distance metric, and there is no efficient method to select an appropriate probability distance metric [27]. Moreover, they can only be effective for a specific type of OOD data, such as noise-disturbed data [28] or adversarial samples [9], without considering common OOD data caused by distributional shifts. Our VBA framework designs a causal model for the distributional shift and proposes a variational backdoor adjustment method based on the causal model. Therefore, the VBA framework is very effective for distributional shifts that commonly exist in nature. Moreover, there is no need to choose a probability distance.

Causal invariant learning methods rely on the assumption of causal invariance, meaning that the causal relationships between variables do not change with environmental variations [6]. These methods all require training data to include environment labels, and the algorithm's performance heavily depends on the quality of environmental partitioning. The work in [1,5] is representative of the causal invariant learning methods. In these studies, they construct constraints based on the invariance of causal feature distributions and ultimately add regularization terms to the loss function to enable the model to learn causal features. However, extracting causal features through optimization methods is passive, often fails to obtain true causal features, and also has strict requirements for the number of environments in the training set [29]. To address this challenge, our proposed method not only constructs a special loss function but also designs an ingenious model structure. It facilitates the model in converging more easily to the optimal solution and eliminates the strict requirement of the model on the number of training set environments.

Currently, some studies have relaxed the constraints on the environment [2,10,30]. However, these methods all have their limitations. The model proposed in [2] can only handle scenarios where the decomposition of invariant and variant features occurs at the raw feature level. It may break down when the decomposition can only be performed in the representation space [10]. The model in [10] is an improved version in [2]. However, it requires pre-setting the clustering parameter  $K$ , and when  $K$  deviates significantly from the ground truth, the model's performance may deteriorate. The model proposed in [30] employs a two-stage process to infer the environment. This process cannot be jointly optimized, and its performance heavily relies on a pre-defined biased model. The model we propose in this paper does not rely on pre-defined models and parameters and does not require explicit inference of the data's environment. Therefore, our method demonstrates better robustness compared to existing approaches.

**Causal inference:** Causal inference is a classic problem in the field of statistics. The research on it can be divided into three theoretical branches: the framework of potential outcomes and counterfactuals [31,32], the structural equation model (SEM) [13,33,34], and graphical modeling [35,36]. The work in [37] established a connection between these frameworks using single-world intervention graphs. However, these methods are applied to low-dimensional and physically meaningful raw data. They cannot be directly applied to high-dimensional scenarios such as images and natural language. In recent years, some researchers have investigated the combination of causal inference and machine learning methods [38,39]. Our proposed method applies causal estimation techniques to image data. We use variational inference methods to make causal estimation amenable to optimization. Furthermore, the VBA framework enables causal inference methods in high-dimensional data through variational inference methods and deep feature extraction models, providing a broader application prospect for causal inference methods.

### 3. Methodology

This section proposes a novel learning objective based on the backdoor adjustment formula and variational inference methods. Then, we construct a learning framework to estimate the conditional probability distribution between features, labels, and confounders.

#### 3.1. Problem Formulation

Consider a training dataset  $\{(x_i, y_i)\}_{i=1}^N$  sampled randomly from the overall population, where  $x_i \in \mathbb{R}^d$  and  $N$  represent the number of collected samples. Unlike invariant learning methods, there is no requirement to know the specific environment from which each data point is sampled. We aim to build a model  $\Phi$  to fit the function  $f(\cdot) : \mathbf{X} \rightarrow Y$ .

As shown in Figure 1a,  $Y$  and some dimensions in  $\mathbf{X}$  are influenced by the confounders  $\mathbf{C} \in \mathbb{R}^m$ , which are associated with the environment. Therefore, the training data  $\{(x_i, y_i)\}_{i=1}^N$  are generated from the distribution  $(x_i, y_i) \sim P(X, Y | \mathbf{C} = c_i)$  [12], where  $c_i$  represents the confounders corresponding to the environment from which the  $i$ -th sample is collected. Therefore, our objective is to learn a model  $\Phi(x) = P(Y | \mathbf{X} = x, \mathbf{C} = c)$ .

#### 3.2. Limitations of the Empirical Risk Minimization Model

To illustrate the necessity of our proposed model, we will discuss the limitations of traditional models based on Empirical Risk Minimization (ERM) in the problem of OOD generalization [1,12] in this subsection.

Most existing machine learning methods optimize models through the ERM loss function as follows:

$$\min_{\theta} \mathbb{E}_{(x,y) \sim P(X=x, Y=y | C=c)} [l(\hat{f}(x; \theta), y)], \quad (1)$$

where  $\theta$  represents the model parameters,  $\hat{f}(\cdot; \theta)$  represents the prediction model, and  $l(\cdot, \cdot)$  represents a certain metric function, e.g., the cross-entropy function. Through the ERM loss function, what is actually fitted is the conditional distribution  $P(Y | \mathbf{X})$ . It is equivalent to the Maximum Likelihood Estimation (MLE) method. However,  $Y$  is actually

generated by the distribution  $P(Y|X, C = c)$ , so different unobservable confounders,  $C$ , correspond to different  $P_C(Y|X)$ . Therefore, ERM methods attempt to generalize multiple distributions,  $\{P_{c(Y|X)}\}_{c \in e_{tr}}$ , by learning a single distribution,  $P(Y|X)$ , where  $e_{tr}$  includes all of the sampling environments in the training dataset. It is clear that the learned model cannot generalize to  $P_{c_{new}}$  when  $c_{new}$  is not in  $e_{tr}$ .

### 3.3. Backdoor Adjustment Based on Variational Inference

To eliminate the influence of confounders on model training, we intervene on the feature  $X$  to cut off the causal path from  $C$  to  $X$ , as shown in Figure 1b. The intervened conditional probability can be expressed as  $P(Y|do(X))$ , where the *do*-operator represents the intervention operation.

According to the backdoor criterion [13], it is known that  $C$  satisfies the backdoor criterion for  $(X, Y)$ . Therefore, we can calculate  $P(Y|do(X))$  by performing backdoor adjustment on  $C$ :

$$P(Y = y|do(X = x)) = \sum_c P(Y = y|X = x, C = c)P(C = c), \tag{2}$$

where  $do(X = x)$  represents the intervention to make  $X = x$ , and  $c$  represents the value of the confounders  $C$ .

A description of backdoor adjustment is provided in Appendix A. Equation (2) allows the result of the intervention to be calculated from observed data without conducting a randomized experiment. Unfortunately, optimizing Equation (2) on the training dataset is not feasible because the values of  $C$  are unobservable. Therefore, the prior distribution  $P(C)$  and the conditional distribution  $P(Y|X, C)$  are both unobservable and cannot be estimated.

To address the above problem, inspired by [12,40], we use the variational inference method to estimate distributions  $P(Y|X, C)$  and use the reparameterization method to estimate prior  $P(C)$ . We introduce a variational distribution  $Q(C|X)$  to estimate the posterior distribution  $P(C|X)$ . Then, through variational inference, we obtained the following Evidence Lower Bound (ELBO) of  $P(Y|do(X))$ :

$$\log P_{\theta}(Y|do(X)) \geq -\mathbb{D}_{KL}(Q(C|X)||P(C)) + \mathbb{E}_{c \sim Q(C|X)}[\log P_{\theta}(Y|X, C)], \tag{3}$$

where  $\theta$  represents the parameters of the inference model and  $\mathbb{D}_{KL}$  represents the Kullback–Leibler divergence. The derivation for Inequation (3) is presented in Appendix B. The right-hand side of the inequality is referred to as ELBO, which consists of two terms. The first term implies that  $Q(C|X)$  should approximate  $P(C)$ , while the second term represents the similarity between the  $\hat{Y}$  generated by the inference model and the ground truth  $Y$ . From Inequation (3), it can be concluded that by maximizing the ELBO, the maximum of  $P_{\theta}(Y|do(X))$  can be achieved.

Above all, we can formulate the following loss function:

$$\sum_{i=1}^N [l(y_i, \Phi(x_i; \theta))] + \beta \mathbb{D}_{KL}(q_{\phi}||P(C)), \tag{4}$$

where  $\beta$  is a weight coefficient,  $\Phi(\cdot; \theta)$  is the inference model,  $q_{\phi}$  is the learned variational distribution, and  $\phi$  represents the parameters of the variational distribution.

Choosing an appropriate prior  $P(C)$  is an important and challenging problem. Some researchers estimate  $P(C)$  by taking the average of the variational posterior  $q_{\phi}$  [41], i.e.,  $P(C) = \frac{1}{N} \sum_{i=1}^N q_{\phi}(C|X = x_i)$ . However, this method requires a significant computational cost [42], and the effectiveness of the prior estimation depends on the quality of the samples. In addition, [40,43] use a predefined distribution as the prior distribution, such as a Gaussian or uniform distribution. However, such choices often impose over-regularization



on the model, limiting its flexibility [12]. Inspired by [40], we choose a mixture variational posterior as the prior distribution for  $C$  with pseudo-inputs, i.e.,

$$P(C) = \frac{1}{K} \sum_{k=1}^K q_{\phi}(C|X = x_k^{(p)}), \tag{5}$$

where  $x^{(p)}$  represents pseudo-inputs, and  $K$  represents the number of pseudo-inputs. We can treat the pseudo-inputs as learnable model parameters, with  $K$  serving as a hyperparameter of the model. Additionally, we set the variational posterior  $q_{\phi}$  to be a multivariate Gaussian distribution  $\mathcal{N}(\mu, \sigma^2 I)$ . The advantage of this choice is that we can use the reparameterization trick to learn the variational posterior, and the Gaussian distribution aligns with our intuition about the confounders in the natural world.

In summary, we have developed a variational inference-based backdoor adjustment algorithm to learn  $P(Y|do(X))$ , which is entirely data-driven. To enable the optimization of our method, we transformed the backdoor adjustment problem into a variational inference problem through mathematical derivation. In the following section, we construct a unique network structure to simulate the variational inference process and provide a detailed overview of the model’s optimization methods.

### 3.4. Model Structure

From Inequation (3), it is evident that we need to parameterize  $Q(C|X)$  and  $P_{\theta}(Y|X, C)$  and fit them through the model. Therefore, we propose a model framework that encodes features to generate the distribution of the confounders and infer the labels of data.

#### 3.4.1. Variational Posterior Encoder

We introduce a variational posterior encoder to parameterize the variational posterior distribution  $Q(C|X)$ . This encoder takes the original features  $X$  as input and produces the parameters of the variational posterior from which  $C$  is sampled.

We define  $\Phi_e$  as the encoding function. To enable the model to be optimized through gradient backpropagation, we use the reparameterization trick to simulate the learning process of  $Q(C|X)$ . For input  $x_i$ , we define  $q_{\phi}(C|X = x_i) = \mathcal{N}(\mu(x_i; \phi), \sigma^2(x_i; \phi)I)$ . Therefore, the encoder encodes  $X$  into a multivariate Gaussian distribution with a mean vector and variance, i.e.,

$$\begin{aligned} \Phi_e(x_i) &= w_i, \text{ where } w_i \in \mathbb{R}^{m+1}; \\ \mu(x_i; \phi) &= w_i[:m]; \\ \sigma(x_i; \phi) &= w_i[m], \end{aligned} \tag{6}$$

where  $w_i$  represents the vector obtained by  $\Phi_e$  for  $x_i$ , and  $m$  represents the dimension of  $C$ .

Through Equation (6), we obtain the mean and variance of the variational posterior. The encoder can adopt any existing backbone network based on the data type, such as Multilayer Perceptron (MLP), Convolutional Neural Network (CNN) [44], or Transformer [45].

#### 3.4.2. Inference Model

Then, according to Inequation (3), it is necessary to sample from  $Q(C|X)$  as input for the inference model  $P_{\theta}(Y|X, C)$ . For similar situations, some researchers first sample  $\epsilon_i$  from  $\mathcal{N}(\mathbf{0}, I)$  and then set  $c_i = \mu(x_i) + \sigma(x_i) \cdot \epsilon_i$  [40,42]. However, random sampling can introduce significant uncertainty into the results, which is not conducive to precise inference by our inference model. In order to avoid the uncertainty introduced by the sampling process, we introduce a network  $\Phi_s$  to model the sampling process:

$$\begin{aligned}
 \Phi_s^{(1)}(\mu_i) &= c_i^{(1)}; \\
 \Phi_s^{(2)}(\sigma_i) &= c_i^{(2)}; \\
 \Phi_s^{(3)}(c_i^{(1)}, c_i^{(2)}) &= c_i,
 \end{aligned}
 \tag{7}$$

where  $\Phi_s^{(1)}$  represents the mean coding network in the sampling network  $\Phi_s$ ,  $\Phi_s^{(2)}$  represents the variance coding network in  $\Phi_s$ , and  $c^{(1)}, c^{(2)}$  represent the coding results of the mean and variance, respectively.

In Equation (7), for simplicity, we denote  $\mu(x_i)$  and  $\sigma(x_i)$  as  $\mu_i$  and  $\sigma_i$ , respectively. Additionally,  $c_i$  represents the confounders corresponding to  $x_i$ . Equation (7) describes the three branch units in the sampling network  $\Phi_s(\mu_i, \sigma_i) = \Phi_s^{(3)}[\Phi_s^{(1)}(\mu_i), \Phi_s^{(2)}(\sigma_i)] = c_i$ . The model  $\Phi_s^{(3)}$  first splices  $c_i^{(1)}$  and  $c_i^{(2)}$  before entering the backbone network. The three branch units in  $\Phi_s$  can use MLP as the backbone network. The network structure of the sampling process is illustrated in Figure 2.

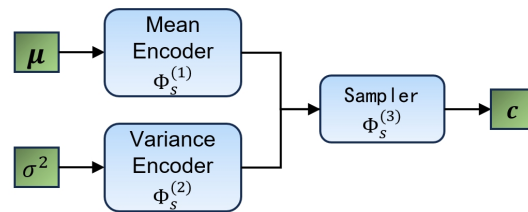


Figure 2. The network structure of the sampling process.

The estimated label  $\hat{y}$  can be obtained through the inference unit  $\Phi_{in}$ :

$$\Phi_{in}(x_i, c_i) = \hat{y}_i,
 \tag{8}$$

where  $\hat{y}_i$  represents the model’s prediction for  $y_i$ .

We parameterize  $P_\theta(Y|X, C)$  through  $\Phi_{in}$ . For more complex data forms, such as images or natural language, extracting features to obtain a low-dimensional representation is necessary. This representation, along with the sampled confounders, then enters the inference unit. In order to visually illustrate the structure of the VBA framework, we present the flowchart of the VBA framework in Figure 3.

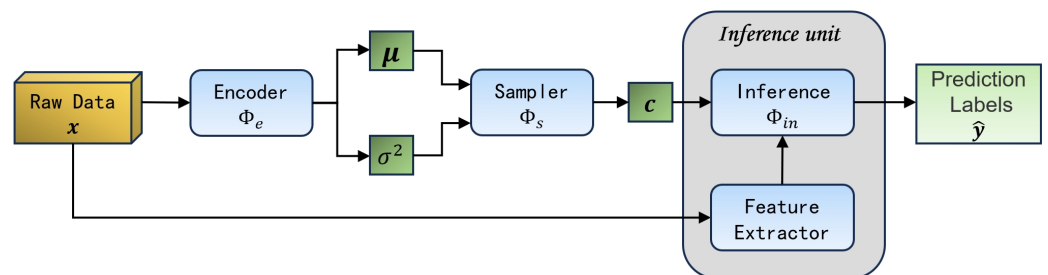


Figure 3. The flowchart of the VBA framework. The blue box represents the network in the framework, and the green box represents the new data representation generated during the processing of the VBA framework.

From the loss function Equation (4), it can be seen that we need to compute the  $\mathbb{D}_{KL}(q_\phi||P(C))$  during optimization. Based on the calculation of the prior in Equation (5), the following method for computing  $\mathbb{D}_{KL}(q_\phi||P(C))$  using samples can be obtained:

$$\mathbb{D}_{KL}[q_\phi(C|X = x_i)||P(C)] = \frac{1}{2} \left[ m \frac{\sigma_i^2}{\sigma_{(p)}^2} + \frac{1}{\sigma_{(p)}^2} (\mu_{(p)} - \mu_i)^T \mathbf{I} (\mu_{(p)} - \mu_i) + \ln \left( \frac{\sigma_{(p)}^{2m}}{\sigma_i^{2m}} \right) - m \right],
 \tag{9}$$

where  $\mu_{(p)}$  and  $\sigma_{(p)}^2$  are the mean and variance of  $P(C)$ . According to the additivity of the Gaussian distribution,  $\mu_{(p)}$  and  $\sigma_{(p)}^2$  can be obtained by summing the mean and variance of each  $q_{\phi}(C|X = x_k^{(p)})$ , respectively. The derivation of Equation (9) is presented in Appendix C.

In this section, we first simulated the variational posterior distribution through an encoder, followed by simulating the sampling process from the variational posterior distribution  $Q(C|X)$  using a sampler. Finally, we simulated  $P_{\theta}(Y|X, C)$  through an inference unit. Algorithm 1 presents the detailed steps of the proposed variational backdoor adjustment (VBA) framework.

---

**Algorithm 1** Variational backdoor adjustment framework

---

**Input:**  $\{(x_i, y_i)\}_{i=1}^N, x_i \in \mathbb{R}^d$ , dimensions of confounders  $m$ , the number of pseudo-inputs  $K$ , and weight parameter  $\beta$ .

Initialize the parameters of  $\Phi_e, \Phi_s, \Phi_{in}$ , and  $\{x_k^{(p)}\}_{k=1}^K$ .

- 1: **while** not converged **do**
- 2:    $\{w_i\}_i^N \leftarrow \{\Phi_e(x_i)\}_i^N$
- 3:    $\mu_i \leftarrow w_i[:m]; \sigma_i^2 \leftarrow w_i[m]$
- 4:   **for**  $k$  in range( $K$ ) **do**
- 5:      $w^{(p)} += \Phi_e(x_k^{(p)})$
- 6:   **end for**
- 7:    $\mu_{(p)} \leftarrow \frac{1}{K} \cdot w^{(p)}[:m]; \sigma_{(p)}^2 \leftarrow \frac{1}{K^2} \cdot w^{(p)}[m]$
- 8:    $KL = \frac{1}{2} \sum_{i=1}^N [m \frac{\sigma_i^2}{\sigma_{(p)}^2} + \frac{1}{\sigma_{(p)}^2} (\mu_{(p)} - \mu_i)^T \mathbf{I} (\mu_{(p)} - \mu_i) + \ln(\frac{\sigma_{(p)}^2}{\sigma_i^2})] + \frac{1}{2} mN$
- 9:    $\{c_i\}_i^N \leftarrow \{\Phi_s(\mu_i, \sigma_i^2)\}_i^N$
- 10:    $\{\hat{y}_i\}_i^N \leftarrow \{\Phi_e(\mu_i, \sigma_i^2)\}_i^N$
- 11:    $\mathcal{L} = \sum_{i=1}^N l(y_i, \hat{y}_i) + \beta KL$
- 12:   Backpropagate the gradients for parameters in  $\Phi_e, \Phi_s, \Phi_{in}$ , and  $\{x_k^{(p)}\}_{k=1}^K$
- 13:   Update the parameters in  $\Phi_e, \Phi_s, \Phi_{in}$ , and  $\{x_k^{(p)}\}_{k=1}^K$
- 14: **end while**

**Output:** The parameters of models  $\Phi_e, \Phi_s$ , and  $\Phi_{in}$ .

---

#### 4. Experimental Results

To test whether our proposed VBA framework can effectively address the impact of distributional shifts caused by confounders on model generalization, we conducted experiments in this section on the simulated, semi-synthetic, and real-world datasets, respectively. We compare our proposed VBA framework with Empirical Risk Minimization (ERM), Distributionally Robust Optimization (DRO) [27], Kernelized Heterogeneous Risk Minimization (KerHRM) [10], Environment Inference for Invariant Learning (EIIL) [30], and Invariant Risk Minimization (IRM) [1]. The IRM method, representing invariant causal learning approaches, requires environment labels. Therefore, we provided environment labels to IRM during training, while other algorithms did not require environment labels. We implement the VBA framework and baselines with PyTorch 2.0.1 on a computer with NVIDIA RTX 3070 Ti. The implementation of the VBA framework is accessed on 19 December 2023. It can be downloaded at <https://github.com/hangsuuuu/VBA>. The parameter settings of the VBA framework in each dataset are shown in Table 1. The settings for other methods are described in Appendix E.



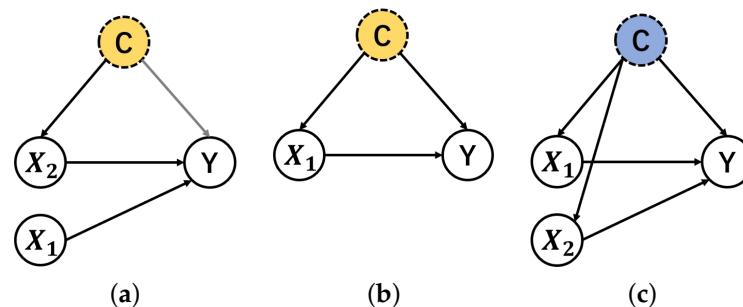
**Table 1.** The parameters and backbone network of the VBA framework used in experiments.

Datasets	$\Phi_e$	$\Phi_s^{(1)}$	$\Phi_s^{(2)}$	$\Phi_s^{(3)}$	$\Phi_{in}$	$m$	$K$	$\beta$
Linear	2-layer MLP	1-layer FNN	1-layer FNN	1-layer FNN	2-layer MLP	1	5	0.5
Colored MNIST	3-layer CNN	2-layer MLP	2-layer MLP	1-layer FNN	2-layer MLP	5	10	0.5
NICO	ResNet [46]	3-layer MLP	3-layer MLP	1-layer FNN	4-layer MLP	20	10	0.2
House prices	2-layer MLP	2-layer MLP	2-layer MLP	2-layer MLP	3-layer MLP	5	5	0.4

#### 4.1. Linear Simulated Data

In this subsection, we assess the performance of the VBA framework on linear simulated data. We employed three different data generation methods. The detailed descriptions of the data generation procedures can be found in Appendix D.

To test the performance of the VBA framework under different conditions, we designed three different OOD cases. The causal graphs for the three cases are illustrated in Figure 4. In each case, the label  $Y$  is generated by a linear combination of  $x$  and  $c$ . From Figure 4 and Table A1, it can be observed that  $C$  is a common unobservable cause for both  $X$  and  $Y$ . Hence,  $C$  serves as a confounder for  $X$  and  $Y$ . By altering the distribution of  $C$ , we can change  $P(Y|X)$ , thereby simulating an OOD dataset.



**Figure 4.** The causal graphs for the linear simulated dataset, where the yellow circle represents  $C$  generated from a Gaussian distribution, and the blue circle indicates  $C$  generated from a uniform distribution. The black arrows indicate a variable directly involved in the generation process of another variable, while the gray arrows signify a variable influencing the generation process of another variable by affecting its distribution parameters. The dashed circles indicate that the variables are unobservable. (a) The causal diagram of Case 1. (b) The causal diagram of Case 2. (c) The causal diagram of Case 3.

We set continuous values for  $y$  to test the model's regression ability. In each case, we generated 800 data points as the training set and 200 data points as the test set. To test the model's performance when there is a distributional shift in the confounder, we use different confounder distributions for the training and test sets. In Case 1 and Case 2, we simulated the distributional shift by changing  $\gamma_1$  and  $\gamma_2$  in Table A1. We chose  $\gamma_1, \gamma_2 \in \{0, 0.5, 1, 1.5\}$  in the training set, and  $\gamma_1 = 2$  in the test set. In Case 3, we chose  $\gamma_3 \in \{1, 1.5, 2, 2.5\}$  in the training set and  $\gamma_3 = 5$  in the test set. To evaluate the performance of the VBA framework when the prior of the confounder is not a Gaussian distribution, as in Case 3, we generated  $C$  using a uniform distribution. We simulated the distributional shift by altering the range of values in the uniform distribution.

The experimental results are reported in Table 2. Each experimental result is the average obtained after running the experiments 10 times.

In Table 2, we evaluate the performance using Mean Squared Error (MSE) and the variance of the MSE. The results indicate that all methods perform suboptimally in Case 3. On the one hand, this is due to using a uniform distribution to generate confounders in Case 3. On the other hand, the distributional shift is most pronounced in Case 3. Overall, ERM performs the worst as it has the highest MSE and variance. DRO, KerHRM, and EIII show varying performances across the three cases, but, overall, KerHRM exhibits the

best performance among these three environment inference algorithms. IRM performs significantly better than DRO, KerHRM, and EIIL, as it utilizes environment labels directly during optimization. In contrast, the performance of the latter three algorithms depends on the effectiveness of environment inference. Our proposed VBA framework performs better than IRM in most cases (Case 1 and Case 2), even without environment labels. However, the MSE of the VBA framework in Case 3 is significantly higher than in Case 1 and Case 2. This is because the distribution of the confounder in the test set differs significantly from a Gaussian distribution, resulting in the model having difficulty accurately inferring the confounder. Although the performance of the VBA framework decreases in Case 3, its MSE remains close to mainstream methods that do not require environment labels (such as KerHRM), and there is no significant increase in variance.

**Table 2.** Results of linear simulation experiments.

Methods	Case 1		Case 2		Case 3		Need Environment Labels?
	MSE	Var	MSE	Var	MSE	Var	
ERM	5.16	0.28	5.04	0.31	5.97	0.35	no
DRO	4.74	0.21	4.51	0.25	5.22	0.23	no
KerHRM	4.52	0.18	4.56	0.21	4.97	0.28	no
EIIL	4.83	0.23	4.74	0.22	5.18	0.25	no
IRM	4.14	<b>0.16</b>	4.17	0.18	<b>4.68</b>	<b>0.20</b>	yes
VBA	<b>4.04</b>	0.18	<b>3.87</b>	<b>0.15</b>	5.07	0.23	no

The bold numbers indicate the best results among all methods.

Table 3 presents the methods' time consumption, including the convergence time during the training phase and the inference time during the prediction phase, where the inference time refers to the time required for the model to predict 200 data points from the test set. Due to the lowest computational and parameter complexity of the ERM model, it exhibits the fastest inference speed. However, ERM has a longer convergence time due to the need for a large number of iterations to converge on OOD datasets, especially in datasets with complex variable relationships, such as in Case 3. Since EIIL performs only environment inference without outputting predicted values for  $y$ , for comparison with other models, we utilize the environment classification output by EIIL as the environment label for IRM. Subsequently, we employ the IRM method to obtain the final prediction results of EIIL. Therefore, to ensure fairness, we recorded the total convergence time for both the environment inference module of EIIL and the inference module of IRM when calculating the convergence time. The recorded inference time reflects the time required for the environment inference module of EIIL to perform environment classification. Thus, EIIL has the longest convergence time among all of the methods. DRO and KerHRM do not separate the training of their environment inference and prediction processes; therefore, their convergence times are shorter than that of EIIL. Although the VBA framework comprises three modules (encoding, sampling, and inference), the simultaneous training of these modules and the backdoor adjustments allow the model to converge after fewer iterations. Consequently, the training time of VBA is comparable to that of IRM, and, in some complex scenarios, it even exhibits the least convergence time. However, the inference time of VBA is slower than that of IRM and ERM. This is because the input features need to pass through multiple modules to obtain the predicted values.

**Table 3.** Convergence time during training and inference time during prediction in linear simulation experiments.

Methods	Case 1		Case 2		Case 3	
	Conv_Time	Infer_Time	Conv_Time	Infer_Time	Conv_Time	Infer_Time
ERM	20.91 s	<b>1.55 s</b>	<b>15.83 s</b>	<b>1.04 s</b>	25.47 s	<b>1.53 s</b>
DRO	29.95 s	2.89 s	23.57 s	1.91 s	37.20 s	3.01 s
KerHRM	25.37 s	2.53 s	23.69 s	1.86 s	29.42 s	2.58 s
EIIL	33.50 s	2.26 s	29.16 s	1.81 s	36.11 s	2.30 s
IRM	<b>17.75 s</b>	2.13 s	16.36 s	1.46 s	19.53 s	2.16 s
VBA	18.56 s	2.50 s	17.20 s	1.83 s	<b>18.76 s</b>	2.39 s

The bold numbers indicate the best results among all methods.

Since the VBA framework uses  $X$  and  $C$  to predict the values of  $Y$ , the ability to generate the correct confounders is a crucial evaluation metric for assessing the interpretability of the VBA framework. To evaluate whether the VBA framework can accurately estimate the confounders, we present in Table 4 the mean values of the estimated confounders obtained by the sampler of the VBA framework. Since different distributions are used to generate the confounders in different environments when generating the dataset, we can evaluate the VBA framework's ability to estimate the confounders by examining the mean values of the estimated confounders in different environments. Table 4 shows that the estimated confounders' mean values by the VBA framework are very close to the true values, demonstrating that the VBA framework provides nearly unbiased estimates for the confounders. The accurate estimation of the confounders is not only a necessary condition for OOD predictions but also indicative of the interpretability of the VBA framework.

**Table 4.** The mean of the confounders generated by the VBA framework. Here,  $E_1 \sim E_4$  represent the training environments, and  $E_5$  is the testing environment. The values in parentheses indicate the true mean of the confounders generated in each environment.

Cases	$E_1$	$E_2$	$E_3$	$E_4$	$E_5$
Case 1	0.02 (0)	0.47 (0.5)	0.97 (1)	1.50 (1.5)	1.91 (2)
Case 2	−0.01 (0)	0.49 (0.5)	1.03 (1)	1.47 (1.5)	1.94 (2)
Case 3	0.06 (0)	0.09 (0)	0.11 (0)	0.09 (0)	0.09 (0)

#### 4.2. Colored MNIST

To further assess the performance of our method on high-dimensional data, we utilized the Colored MNIST [1], a semi-synthetic dataset, in this experiment. The Colored MNIST, is derived from MNIST and is designed for binary classification methods. In Colored MNIST, the hand-written digits 0–4 are labeled as  $y = 0$ , and digits 5–9 are labeled as  $y = 1$ . To simulate distributional shifts in the data, for digits with  $y = 1$ , we colored them green with a probability of  $p^e$ , and, for digits with  $y = 0$ , we colored them green with a probability of  $1 - p^e$ . Then, the remaining uncolored digits were colored red. In the training set, we set  $p \in \{0.1, 0.3\}$ , while in the testing set,  $p = 0.9$ . The data for this example can be generated using the code available at <https://github.com/facebookresearch/InvariantRiskMinimization>. The code is accessed on 29 September 2020. For the IRM method, we divided different environments based on different values of  $p^e$  and enabled IRM to utilize environment labels. For other methods, we mix generated training data from different environments and use them as input for the models.

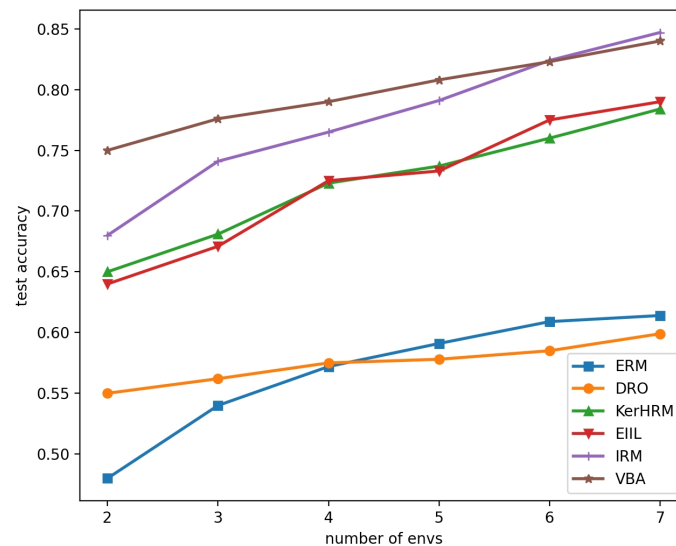
In Colored MNIST, the original pixel values and the arrangement of pixels in the images constitute the model's raw input  $X$ . The true numerical value of the handwritten digit serves as an unobservable confounder  $C$  that influences both  $X$  and  $Y$ . The proposed VBA framework is to eliminate the impact of environmental factors (digit color) on model training through backdoor adjustment.

Table 5 presents the experimental results of VBA and other methods. It includes the accuracy and time consumption of all methods. As in the previous experiment, each experimental result is the average obtained after running the experiments 10 times. Inference time refers to the time required for the model to predict 10,000 images from the test set. We employed three metrics to evaluate the methods' performance: training accuracy, testing accuracy, and generalization gap ( $|Train\_acc - Test\_acc|$ ). To assess the impact of the number of environments in the training set on the methods' performance, we tested each method's performance as the number of environments varied from two to seven. In different training environments,  $p^e$  values range from 0.1 to 0.7. The testing accuracy of each method with respect to the number of training environments is shown in Figure 5.

**Table 5.** Results in Colored MNIST dataset.

Methods	Train_Acc	Test_Acc	Gener_Gap	Conv_Time	Infer_Time	Need Environment Labels?
ERM	<b>0.94</b>	0.48	0.46	33.32 s	<b>5.10 s</b>	no
DRO	0.79	0.55	0.24	87.12 s	8.31 s	no
KerHRM	0.81	0.65	0.16	42.60 s	9.02 s	no
EIIL	0.85	0.64	0.21	79.35 s	6.38 s	no
IRM	0.81	0.68	0.13	35.38 s	5.94 s	yes
VBA	0.84	<b>0.75</b>	<b>0.09</b>	<b>31.61 s</b>	8.40 s	no

The bold numbers indicate the best results among all methods.



**Figure 5.** The testing accuracy of all methods with respect to the number of training environments.

By benefiting from environment labels, IRM achieves decent testing accuracy. However, in scenarios with fewer environments, the performance of IRM significantly lags behind the VBA framework. This result indicates that the performance of IRM is highly dependent on the heterogeneity of environments. KerHRM and EIIL have similar test accuracies in various situations because both methods are based on environment inference, and they obtain similar environment partitions. However, KerHRM and EIIL fall significantly behind the VBA framework regarding test accuracy and generalization gap. ERM and DRO obtain test accuracies close to random selection (50%) in this experiment, and, as the number of environments increases, the performance of DRO even falls behind that of ERM. As the number of environments increases, the distributional shift in the training data is weakened. This benefits ERM in learning the relationship between features and labels, while it does not significantly assist the robust optimization algorithm DRO. The results show that VBA exhibits the best testing performance in terms of testing accuracy and generalization gap. The VBA framework maintains high prediction accuracy when the number of training environments is two. However, due to their high requirements for

environmental heterogeneity, other methods exhibit much lower prediction accuracies than that of the VBA framework. This indicates that the VBA framework has a more significant advantage when the training set exhibits low environmental heterogeneity.

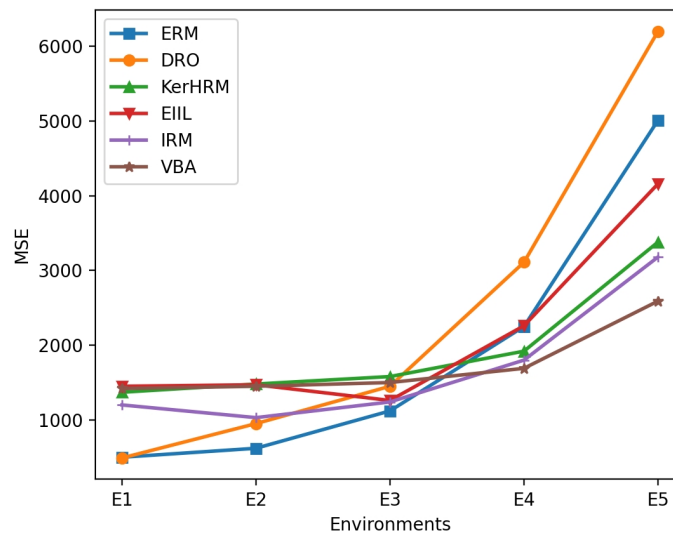
Regarding time consumption, the VBA framework benefits from the rapid convergence facilitated by the variational backdoor adjustment intervention, resulting in the least convergence time. However, the inference speed of the VBA framework ranks only at a moderate level among all of the methods. This is attributed to its complex inference process. DRO struggles to converge on the Colored MNIST dataset, with its training iterations far exceeding those of other methods. Consequently, the convergence time of DRO is significantly longer than those of other methods. This indicates that robust optimization methods struggle to converge stably to solutions with robustness.

#### 4.3. Real-World Data

In this subsection, we utilize two real-world datasets to assess the practical applicability of the VBA framework in real scenarios. The two datasets are Non-I.I.D. Image Dataset with Contexts (NICO) [47] and house sales prices from King County, USA [10], respectively. We evaluate the methods' classification and regression capabilities in real-world scenarios through these two datasets.

NICO contains wildlife and vehicle images captured in different environments. The environments in the NICO dataset are divided based on the collection environment of the images. This dataset is available at <https://nico.thumedia.com>. This dataset is accessed on 18 April 2022. In this experiment, we use NICO to construct a binary classification dataset. The dataset contains images of cows and bears from three environments (forest, river, and snowfield). The different collection environments led to a distributional shift in the data. Since our goal is to classify the species of animals, in this dataset, the label  $Y$  represents the species of the animals, and the confounder  $C$  may include advanced features such as the outline and color of the animals, as well as the background color. We choose data from the forest and river environments to form the training set, while data from the snowy environment serve as the test set.

The house sales prices dataset comprises the sale prices and 17 different house attributes, such as the number of rooms, the built year, etc. This dataset is available at <https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>. This dataset is accessed on 11 January 2020. The dataset contains data for a total of 3000 houses. We use the house sale prices as the target variable  $Y$ , with the other information about the houses serving as predictive variables  $X$ . The relationship between predictor variables and house prices may vary with the built time of the houses. This is because the criteria for assessing the value of houses may change over different periods. Thus, the dataset experiences a distribution shift as the construction year changes. We can partition the data into different environments based on the built year to construct an OOD dataset. The built years of houses in the dataset range from 1872 to 2010. However, due to the scarcity of houses built between 1872 and 1900, we only select data with built years between 1900 and 2010. We divide the dataset into five periods, where the first three decades serve as the training set, and each subsequent period contains a time span of two decades. We test each method in the subsequent four periods and display the MSE for each method in Figure 6. Since training IRM requires environment labels, we divide the training environment into two environments based on the built year, using these as the training environment for IRM.



**Figure 6.** The results of all methods on house sales prices dataset. We trained all methods in E1 and tested them in E2–E5.

The results of the experiment on NICO are provided in Table 6. Inference time refers to the time required for the model to predict 250 images from the test set. The testing accuracy of ERM and DRO is very low (close to 50%), indicating that the dataset exhibits a distributional shift. KerHRM and EIIL still exhibit similar performance, both in terms of testing accuracy and generalization gap. This indicates that invariant learning methods based on the environment have limited capabilities in handling complex data, as their testing accuracy is only slightly higher than that of ERM. The testing accuracy of IRM shows a significant improvement compared to that of ERM, indicating that environment labels provide valuable information for addressing out-of-distribution generalization problems. The proposed VBA framework outperforms all of the compared methods, exhibiting the best testing accuracy and the lowest generalization gap. This indicates that our designed backdoor adjustment method can address complex OOD generalization problems effectively.

The VBA framework exhibits the shortest convergence time among all methods, indicating its ability to rapidly converge to optimal solutions even on complex datasets. This is attributed to the VBA framework having three modules with different functionalities (encoding, sampling, and inference), each of which can rapidly accomplish its specific task during training. Despite incorporating environment labels, IRM experiences a longer convergence time than VBA. This is attributed to the challenge faced by IRM's single model structure in effectively capturing the complex variable relationships present in the OOD dataset. While KerHRM also adopts a training approach with multiple modules, it initially requires clustering to partition data environments, a process that typically converges slowly on complex datasets. Consequently, KerHRM has a longer convergence time than VBA. Due to the complexity of the model parameters, the VBA framework requires a longer inference time. However, even on complex datasets, the inference time of the VBA framework is only slightly longer than those of DRO and EIIL and shorter than that of KerHRM.

Figure 6 shows that all methods perform well on the test set close to the training environment, while the MSE significantly increases in the test set far from the training environment. This aligns with our intuition, as greater time intervals make the distributional shift more pronounced. The MSEs of ERM and DRO increase sharply, and the MSE of DRO was even higher than that of ERM in the test set, indicating that DRO cannot achieve OOD generalization in the real-world dataset. IRM shows slower MSE growth than KerHRM and EIIL, and its MSE is lower than those of KerHRM and EIIL across all test sets. This suggests that environment labels are beneficial for enhancing OOD generalization. The MSE curve for the VBA framework exhibits a very gradual increase, and, in E4 and E5, the



MSE is even lower than that of IRM. This indicates that the VBA framework maintains excellent OOD generalization even in situations with strong dataset distributional shifts.

**Table 6.** Results in real-world dataset.

Methods	Train_Acc	Test_Acc	Gener_Gap	Conv_Time	Infer_Time	Need Environment Labels?
ERM	<b>0.90</b>	0.54	0.36	1648.64 s	<b>10.91 s</b>	no
DRO	0.79	0.55	0.24	4681.59 s	16.89 s	no
KerHRM	0.81	0.62	0.19	3980.41 s	19.90 s	no
EIIL	0.80	0.58	0.18	5023.60 s	16.32 s	no
IRM	0.85	0.69	0.16	2351.21 s	15.64 s	yes
VBA	0.83	<b>0.73</b>	<b>0.10</b>	<b>1631.63 s</b>	17.75 s	no

The bold numbers indicate the best results among all methods.

In this subsection, we evaluate the performance of the VBA framework in two real-world scenarios. The dataset includes both high-dimensional image data and low-dimensional tabular data. Therefore, the experiments in this subsection can validate the practical applicability of the method. The experimental results demonstrate that the VBA framework performs excellently in various scenarios, even when significant distributional shifts occur in the data. This indicates that the VBA framework has excellent applicability in real-world scenarios.

In the experimental section, we combined several mainstream backbone networks, such as MLP, CNN, ResNet, etc., with the VBA framework and achieved good performance. Since the VBA framework does not specify the type of network used. Theoretically, the VBA framework can be combined with any backbone network. Therefore, when applying the VBA framework in practice, we can choose an appropriate backbone network based on the data type. This enables the VBA framework to apply to a wide range of data types.

## 5. Discussion

Currently, most existing OOD generalization methods divide the dataset into different environments [1,2,5,10]. During training, the model needs to calculate losses separately in different environments, significantly compromising computational efficiency. The VBA framework adopts an entirely different perspective, estimating causal effects using intervention methods, and thus does not require partitioning the training set into different environments. Therefore, the VBA framework has better practicality compared to other OOD methods. According to the nature of the intervention operator, calculating  $P(Y|do(X))$  requires using the distribution of the confounder  $C$ . However, due to the unobservability and diversity of the confounder's values, directly calculating  $P(C)$  is impractical. Therefore, we employ variational inference methods to approximate the posterior distribution  $P(C)$  and solve it through model optimization. This significantly enhances the computational efficiency of the model. Although the multi-module structure of the VBA framework slows down its inference speed, the experimental results indicate that this structure enables the model to converge quickly during training. Therefore, compared to other OOD generalization methods, the VBA framework is more suitable for large-scale datasets.

We have demonstrated through experiments that the VBA framework is flexible and can integrate with most mainstream backbone networks. Therefore, if more advanced feature-extraction backbone models emerge in the future, the performance of the VBA framework will be further enhanced. The flexibility of the VBA framework also enables it to handle various types of data, such as natural language and images.

## 6. Conclusions

In this paper, we focus on the issue of OOD generalization. Considering the characteristics of out-of-distribution generalization problems and the limitations of existing methods, we propose a causal learning framework based on variational backdoor adjustment (VBA). We construct a causal graph for the OOD generalization problem by observing features  $X$ ,

labels  $Y$ , and unobservable confounders  $C$ . We then employ variational backdoor adjustment to mitigate the deleterious effects of confounders on the model's generalization ability.

The VBA framework demonstrates excellent predictive performance, even in distributional shifts, without relying on environment labels. We theoretically derive that the VBA framework can simulate the backdoor adjustment method and learn the causal relationship between  $X$  and  $Y$ . We then conducted experiments on datasets of different types. Due to the proposed VBA framework eliminating the interference of confounders through backdoor adjustment without relying on environmental heterogeneity, it exhibits a significant advantage compared to other invariance learning methods in scenarios with low environmental heterogeneity. The experimental results indicate that the performance of the VBA framework surpasses that of most mainstream OOD generalization methods. This suggests that the VBA framework can endow models with robust OOD generalization capabilities even without environment labels. We have also demonstrated through experiments that the VBA framework exhibits more significant superiority than other methods in datasets with stronger distributional shifts.

The VBA framework introduces a novel perspective for OOD generalization methods by combining causal inference with deep learning through variational inference and optimization methods. The incorporation of causal inference enhances the model's OOD generalization capabilities and imparts interpretability to the model. The VBA framework demonstrates excellent OOD generalization performance without the need for training environment labels. This enables OOD generalization methods to extend to domains without training environment labels.

## 7. Limitations and Future Work

The VBA framework utilizes three modules (encoding, sampling, and inference) to achieve variational adjustment. While this imparts excellent predictive capabilities to the VBA framework, the complex model structure requires a longer inference time. This hinders the application of the VBA framework in tasks that require real-time prediction, such as autonomous driving and high-frequency stock trading. Therefore, enhancing the structure of the VBA framework to achieve faster inference speed would be a topic worthy of further investigation.

For ease of optimization, we choose the Gaussian distribution as the posterior distribution for the confounder. However, when the true distribution of the confounder differs significantly from the Gaussian distribution, the performance of the VBA framework may decline. Therefore, finding a better method for simulating the distribution would benefit the VBA framework. This method should not only be objective but also facilitate sampling.

The VBA framework requires sampling from the conditional distribution  $P(C|X)$  obtained from the encoder. However, a completely random sampling process can hinder the convergence of the training of the inference unit model. Therefore, we use a neural network in VBA to simulate the sampling process. This approach may not fully capture the meaning of the mean and variance obtained from the encoder. We believe that investigating sampling methods that better capture the characteristics of probability distributions is meaningful work. These sampling methods allow the inference process of VBA to meet theoretical requirements better, thereby further improving the performance of the VBA framework.

**Author Contributions:** Conceptualization, H.S. and W.W.; methodology, H.S.; software, H.S.; validation, H.S. and W.W.; formal analysis, H.S.; data curation, H.S.; writing—original draft preparation, H.S.; writing—review and editing, W.W. and H.S.; supervision, W.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The linear simulated data can be generated from <https://github.com/hangsuuuu/Linear-simulated-data-in-VBA>. The linear simulated data is accessed on 1 December 2023. The Colored MNIST dataset can be generated from <https://github.com/facebookresearch/InvariantRiskMinimization>. In order to be fair, we deleted the disturbance in our experimental data.

The real-world dataset NICO can be downloaded from <https://www.dropbox.com/sh/8mouawi5guaupyb/AAD4fdySrA6fn3PgSmhKwFgva?dl=0>, and the house sales prices from King County, USA, can be downloaded from <https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>.

**Conflicts of Interest:** The authors declare no conflict of interest.

### Abbreviations

The following abbreviations are used in this manuscript:

OOD Out-of-Distribution  
 VBA Variational Backdoor Adjustment

### Appendix A. The Description of Backdoor Adjustment

Firstly, we provide the definition of the backdoor criterion [11,13]:

**Definition A1.** For the path from  $X$  to  $Y$ , if the nodes in the set  $C$  are not descendants of  $X$ , and conditioning on  $C$  blocks every path from  $X$  to  $Y$  that contains an arrow into  $X$ , then  $C$  satisfies the *backdoor criterion* for  $(X, Y)$ .

**Theorem A1.** If  $C$  satisfies the backdoor criterion, then backdoor adjustment can be performed on  $C$  to calculate  $P(Y|do(X))$  as:

$$P(Y = y|do(X = x)) = \sum_c P(Y = y|X = x, C = c)P(C = c) \tag{A1}$$

**Proof of Theorem A1.** Because  $do(X)$  represents the intervention that cuts off all causal paths leading to  $X$ , for the intervened model (as shown in Figure 1b)  $P(Y|do(X)) = P_{inter}(Y|X)$ , where  $P_{inter}$  represents the probability distribution in the intervened model, the backdoor adjustment formula can be derived as follows:

$$\begin{aligned} P(Y = y|do(X = x)) &= P_{inter}(Y = y|X = x) \\ &= \sum_c P_{inter}(Y = y|X = x, C = c)P_{inter}(C = c|X = x) \\ &= \sum_c P_{inter}(Y = y|X = x, C = c)P_{inter}(C = c) \\ &= \sum_c P(Y = y|X = x, C = c)P(C = c) \end{aligned} \tag{A2}$$

The second equality in the equation is obtained from the law of total probability. The third equality can be derived from the causal graph shown in Figure 1b and the properties of conditional probability. The final equality is derived from the invariance relationship.

According to Definition A1 and the causal model shown in Figure 1a, it can be concluded that the confounder  $C$  satisfies the backdoor criterion. So, according to Theorem A1, the conditional probability distribution of  $Y$  after intervening on  $X$  can be obtained from Equation (2). □

### Appendix B. The Derivation of ELBO

It can be derived from Equation (2) that  $P(Y|do(X)) = \mathbb{E}_{c \sim P(C)}[P(Y|X, C)]$ . Then, we can derive the ELBO through the following steps:

$$\begin{aligned}
 \log P(Y|do(\mathbf{X})) &= \log \mathbb{E}_{c \sim P(\mathbf{C})} [P(Y|\mathbf{X}, \mathbf{C})] \\
 &= \log \mathbb{E}_{c \sim P(\mathbf{C})} [P(Y|\mathbf{X}, \mathbf{C}) \cdot \frac{Q(\mathbf{C}|\mathbf{X})}{Q(\mathbf{C}|\mathbf{X})}] \\
 &= \log \sum_c [P(Y|\mathbf{X}, \mathbf{C}) \cdot \frac{P(\mathbf{C})}{Q(\mathbf{C}|\mathbf{X})} \cdot Q(\mathbf{C}|\mathbf{X})] \\
 &= \log \mathbb{E}_{c \sim Q(\mathbf{C}|\mathbf{X})} [P(Y|\mathbf{X}, \mathbf{C}) \cdot \frac{P(\mathbf{C})}{Q(\mathbf{C}|\mathbf{X})}] \\
 &\geq \mathbb{E}_{c \sim Q(\mathbf{C}|\mathbf{X})} [\log [P(Y|\mathbf{X}, \mathbf{C}) \cdot \frac{P(\mathbf{C})}{Q(\mathbf{C}|\mathbf{X})}]] \\
 &= \mathbb{E}_{c \sim Q(\mathbf{C}|\mathbf{X})} [\log P(Y|\mathbf{X}, \mathbf{C}) + \log \frac{P(\mathbf{C})}{Q(\mathbf{C}|\mathbf{X})}] \\
 &= \sum_{c \sim Q(\mathbf{C}|\mathbf{X})} Q(\mathbf{C}|\mathbf{X}) \cdot [\log P(\mathbf{C}) - \log Q(\mathbf{C}|\mathbf{X})] + \mathbb{E}_{c \sim Q(\mathbf{C}|\mathbf{X})} [\log P(Y|\mathbf{X}, \mathbf{C})] \\
 &= -\mathbb{D}_{KL}[Q(\mathbf{C}|\mathbf{X})||P(\mathbf{C})] + \mathbb{E}_{c \sim Q(\mathbf{C}|\mathbf{X})} [\log P(Y|\mathbf{X}, \mathbf{C})],
 \end{aligned} \tag{A3}$$

where the derivation of the inequality relies on the convexity of the logarithmic function and Jensen’s inequality.

### Appendix C. Calculation of Kullback–Leibler Divergence

The derivation process for Equation (9) is as follows:

$$\begin{aligned}
 \mathbb{D}_{KL}[q_\phi(\mathbf{C}|\mathbf{X} = x_i)||P(\mathbf{C})] &= \mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [\ln q_\phi(\mathbf{C}|\mathbf{X} = x_i) - \ln P(\mathbf{C})] \\
 &= \frac{1}{2} \mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [-\ln \sigma_i^{2m} - (\mathbf{x} - \boldsymbol{\mu}_i)^T \frac{1}{\sigma_i^2} \mathbf{I}(\mathbf{x}_i - \boldsymbol{\mu}_i) + \ln \sigma_{(p)}^{2m} + (\mathbf{x} - \boldsymbol{\mu}_{(p)})^T \frac{1}{\sigma_{(p)}^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_{(p)})] \\
 &= \frac{1}{2} \ln \frac{\sigma_{(p)}^{2m}}{\sigma_i^{2m}} + \frac{1}{2} \mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [-\text{tr}[\frac{1}{\sigma_i^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T] + \text{tr}[\frac{1}{\sigma_{(p)}^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_{(p)})(\mathbf{x} - \boldsymbol{\mu}_{(p)})^T]]
 \end{aligned} \tag{A4}$$

Then, we define that:

$$\begin{aligned}
 A &= \mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [-\text{tr}[\frac{1}{\sigma_i^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T] + \text{tr}[\frac{1}{\sigma_{(p)}^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_{(p)})(\mathbf{x} - \boldsymbol{\mu}_{(p)})^T]] \\
 &= -\text{tr}[\mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [\frac{1}{\sigma_i^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T]] + \text{tr}[\mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [\frac{1}{\sigma_{(p)}^2} \mathbf{I}(\mathbf{x} - \boldsymbol{\mu}_{(p)})(\mathbf{x} - \boldsymbol{\mu}_{(p)})^T]] \\
 &= -\text{tr}[\frac{1}{\sigma_i^2} \mathbf{I} \mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T]] + \text{tr}[\frac{1}{\sigma_{(p)}^2} \mathbf{I} \mathbb{E}_{q_\phi(\mathbf{C}|\mathbf{X}=x_i)} [(\mathbf{x}\mathbf{x}^T - \boldsymbol{\mu}_{(p)}\boldsymbol{\mu}_{(p)}^T - \mathbf{X}\boldsymbol{\mu}_{(p)}^T + \boldsymbol{\mu}_{(p)}\boldsymbol{\mu}_{(p)}^T)] \\
 &= -\text{tr}[(\frac{1}{\sigma_i^2} \mathbf{I})(\sigma_i^2 \mathbf{I})] + \text{tr}[\frac{1}{\sigma_{(p)}^2} \mathbf{I}[(\sigma_i^2 \mathbf{I} + \boldsymbol{\mu}_i\boldsymbol{\mu}_i^T - \boldsymbol{\mu}_{(p)}\boldsymbol{\mu}_i^T - \boldsymbol{\mu}_i\boldsymbol{\mu}_{(p)}^T + \boldsymbol{\mu}_{(p)}\boldsymbol{\mu}_{(p)}^T)] \\
 &= -m + \text{tr}[(\frac{1}{\sigma_{(p)}^2} \mathbf{I})(\sigma_i^2 \mathbf{I})] + \text{tr}[\boldsymbol{\mu}_i(\frac{1}{\sigma_{(p)}^2} \mathbf{I})\boldsymbol{\mu}_i^T - 2\boldsymbol{\mu}_i^T(\frac{1}{\sigma_{(p)}^2} \mathbf{I})\boldsymbol{\mu}_{(p)} + \boldsymbol{\mu}_{(p)}^T(\frac{1}{\sigma_{(p)}^2} \mathbf{I})\boldsymbol{\mu}_{(p)}] \\
 &= -m + m \frac{\sigma_i^2}{\sigma_{(p)}^2} + \frac{1}{\sigma_{(p)}^2} (\boldsymbol{\mu}_{(p)} - \boldsymbol{\mu}_i)^T \mathbf{I}(\boldsymbol{\mu}_{(p)} - \boldsymbol{\mu}_i),
 \end{aligned} \tag{A5}$$

where the fourth line uses the property:  $\mathbb{E}[xx^T] = \Sigma + \mu\mu^T$ . Finally, substituting A into Equation (A4), we obtain that:

$$\begin{aligned} & \mathbb{D}_{KL}[q\phi(C|X = x_i)||P(C) = x_i)||P(C)] \\ &= \frac{1}{2} \left[ m \frac{\sigma_i^2}{\sigma_{(p)}^2} + \frac{1}{\sigma_{(p)}^2} (\mu_{(p)} - \mu_i)^T \mathbf{I} (\mu_{(p)} - \mu_i) + \ln\left(\frac{\sigma_{(p)}^{2m}}{\sigma_i^{2m}}\right) - m \right], \end{aligned} \tag{A6}$$

where  $\mathbf{I}$  represents the unit matrix with diagonal elements of 1 and other elements of 0.

### Appendix D. Generating Linear Simulated Data

We constructed three simulated datasets following the causal structure illustrated in Figure 1a. Table A1 provides detailed descriptions of the data generation formulas.

**Table A1.** The data generation formulas of linear simulated data.

Case 1	Case 2	Case 3
$c \leftarrow \mathcal{N}(\gamma_1^1, 1)$	$c \leftarrow \mathcal{N}(\gamma_2, 2)$	$c \leftarrow \mathcal{U}(-\gamma_3, \gamma_3)$
$c_{test}^2 \leftarrow \mathcal{N}(2, 1)$	$c_{test} \leftarrow \mathcal{N}(1, 1)$	$c_{test} \leftarrow \mathcal{U}(0, 1)$
$x_1 \leftarrow \mathcal{N}(0.5, 1)$	$x_1 \leftarrow \mathcal{N}(0, 1) + c$	$x_1 \leftarrow \mathcal{N}(0, 1) + c$
$x_2 \leftarrow \mathcal{U}(-1, 1) + 2c$	$y \leftarrow x_1 + 0.5c$	$x_2 \leftarrow \mathcal{U}(-1, 0) - 2c$
$y \leftarrow 2x_1 + x_2 + \mathcal{N}(c, 1)$		$y \leftarrow x_1 + 3x_2 + c$

<sup>1</sup> To simulate the distributional shift, we generate data with different distributions by setting multiple values for  $\gamma$ .

<sup>2</sup>  $c_{test}$  represents the distribution of the confounder in the test set.

### Appendix E. Model Settings

Below, we will introduce the parameter settings of each model on different datasets.

#### Appendix E.1. Linear Simulated Data

**ERM** adopts a three-layer MLP as the backbone network and employs the Mean Squared Error (MSE) as the loss function. **DRO** adopts a three-layer MLP as the backbone network and sets  $\rho = 0.5$ . **EIIL** adopts a three-layer MLP as the backbone network. **KerHRM** adopts a two-layer MLP as the  $\mathcal{M}_p$  and  $\mathcal{M}_c$ . We set the cluster number  $K = 2$ . **IRM** adopts a three-layer MLP as the backbone network and sets  $\lambda = 0.5$ .

#### Appendix E.2. Colored MNIST

**ERM** adopts a three-layer CNN as the backbone network and employs cross-entropy as the loss function. **DRO** adopts a three-layer CNN as the backbone network and sets  $\rho = 0.8$ . **EIIL** adopts a three-layer CNN as the backbone network. **KerHRM** adopts a two-layer MLP as the  $\mathcal{M}_p$  and a three-layer CNN as  $\mathcal{M}_c$ . We set the cluster number  $K = 5$ . **IRM** adopts a three-layer CNN as the backbone network and sets  $\lambda = 0.2$ .

#### Appendix E.3. Real-World Data

**NICO: ERM** adopts ResNet18 as the backbone network and employs cross-entropy as the loss function. **DRO** adopts ResNet18 as the backbone network and sets  $\rho = 0.5$ . **EIIL** adopts ResNet18 as the backbone network. **KerHRM** adopts a five-layer MLP as the  $\mathcal{M}_p$  and adopts ResNet18 as  $\mathcal{M}_c$ . We set the cluster number  $K = 8$ . **IRM** adopts ResNet18 as the backbone network and sets  $\lambda = 0.1$ .

**House prices: ERM** adopts a four-layer MLP as the backbone network and employs cross-entropy as the loss function. **DRO** adopts a four-layer MLP as the backbone network and sets  $\rho = 0.5$ . **EIIL** adopts a three-layer MLP as the backbone network. **KerHRM** adopts a three-layer MLP as the  $\mathcal{M}_p$  and a four-layer MLP as  $\mathcal{M}_c$ . We set the cluster number  $K = 5$ . **IRM** adopts a four-layer MLP as the backbone network and sets  $\lambda = 0.2$ .

## References

- Arjovsky, M.; Bottou, L.; Gulrajani, I.; Lopez-Paz, D. Invariant Risk Minimization. *arXiv* **2019**, arXiv:1907.02893.
- Liu, J.; Hu, Z.; Cui, P.; Li, B.; Shen, Z. Heterogeneous risk minimization. In Proceedings of the International Conference on Machine Learning, Virtual Event, 18–24 July 2021; pp. 6804–6814.
- Liu, J.; Shen, Z.; He, Y.; Zhang, X.; Xu, R.; Yu, H.; Cui, P. Towards out-of-distribution generalization: A survey. *arXiv* **2021**, arXiv:2108.13624.
- Beery, S.; Van Horn, G.; Perona, P. Recognition in terra incognita. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 456–473.
- Yin, M.; Wang, Y.; Blei, D.M. Optimization-based causal estimation from heterogenous environments. *arXiv* **2021**, arXiv:2109.11990.
- Schölkopf, B.; Locatello, F.; Bauer, S.; Ke, N.R.; Kalchbrenner, N.; Goyal, A.; Bengio, Y. Toward causal representation learning. *Proc. IEEE* **2021**, *109*, 612–634. [[CrossRef](#)]
- Peters, J.; Bühlmann, P.; Meinshausen, N. Causal inference using invariant prediction: Identification and confidence intervals. *J. R. Stat. Soc. Ser. B* **2016**, *78*, 947–1012. [[CrossRef](#)]
- Koyama, M.; Yamaguchi, S. Out-of-distribution generalization with maximal invariant predictor. In Proceedings of the CoRR, Victoria, BC, Canada, 18 November–16 December 2020.
- Wang, R.; Yi, M.; Chen, Z.; Zhu, S. Out-of-distribution generalization with causal invariant transformations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 375–385.
- Liu, J.; Hu, Z.; Cui, P.; Li, B.; Shen, Z. Kernelized heterogeneous risk minimization. *arXiv* **2021**, arXiv:2110.12425.
- Pearl, J. Causal inference in statistics: An overview. *Stat. Surv.* **2009**, *3*, 96. [[CrossRef](#)]
- Yang, C.; Wu, Q.; Wen, Q.; Zhou, Z.; Sun, L.; Yan, J. Towards out-of-distribution sequential event prediction: A causal treatment. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 22656–22670.
- Pearl, J.; Glymour, M.; Jewell, N.P. *Causal Inference in Statistics: A Primer*; John Wiley & Sons: Hoboken, NJ, USA, 2016; pp. 24–29.
- Muandet, K.; Balduzzi, D.; Schölkopf, B. Domain generalization via invariant feature representation. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 10–18.
- Recht, B.; Roelofs, R.; Schmidt, L.; Shankar, V. Do imagenet classifiers generalize to imagenet? In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 9413–9424.
- Schneider, S.; Rusak, E.; Eck, L.; Bringmann, O.; Brendel, W.; Bethge, M. Improving robustness against common corruptions by covariate shift adaptation. *arXiv* **2020**, arXiv:2006.16971.
- Tu, L.; Lalwani, G.; Gella, S.; He, H. An empirical study on robustness to spurious correlations using pre-trained language models. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 621–633. [[CrossRef](#)]
- Yi, M.; Wang, R.; Sun, J.; Li, Z.; Ma, Z.-M. Improved OOD generalization via conditional invariant regularizer. *arXiv* **2022**, arXiv:2207.06687.
- Sinha, A.; Namkoong, H.; Duchi, J. Certifying some distributional robustness with principled adversarial training. In Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018.
- Cui, P.; Athey, S. Stable learning establishes some common ground between causal inference and machine learning. *Nat. Mach. Intell.* **2022**, *4*, 110–115. [[CrossRef](#)]
- Rojas-Carulla, M.; Schölkopf, B.; Turner, R.; Peters, J. Invariant models for causal transfer learning. *J. Mach. Learn. Res.* **2018**, *19*, 1309–1342.
- Kuang, K.; Xiong, R.; Cui, P.; Athey, S.; Li, B. Stable prediction across unknown environments. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1617–1626.
- Schölkopf, B. Causality for Machine Learning. *arXiv* **2018**, arXiv:1911.10500.
- Chang, S.; Zhang, Y.; Yu, M.; Jaakkola, T.S. Invariant rationalization. In Proceedings of the International Conference on Machine Learning, ICML, Virtual Event, 13–18 July 2020.
- Belcastro, L.; Carbone, D.; Cosentino, C.; Marozzo, F.; Trunfio, P. Enhancing Cryptocurrency Price Forecasting by Integrating Machine Learning with Social Media and Market Data. *Algorithms* **2023**, *16*, 542. [[CrossRef](#)]
- Shen, Z.; Cui, P.; Zhang, T.; Kuang, K. Stable learning via sample reweighting. In Proceedings of the 34th AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 5692–5699.
- Duchi, J.; Namkoong, H. Learning models with uniform performance via distributionally robust optimization. *Ann. Stat.* **2018**, *49*, 1378–1406. [[CrossRef](#)]
- Yi, M.; Hou, L.; Sun, J.; Shang, L.; Jiang, X.; Liu, Q.; Ma, Z.-M. Improved ood generalization via adversarial training and pretraing. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 11987–11997.
- Kamath, P.; Tangella, A.; Sutherland, D.J.; Srebro, N. Does invariant risk minimization capture invariance? In Proceedings of the International Conference on Artificial Intelligence and Statistics, Virtual, 13–15 April 2021; pp. 4069–4077.
- Creager, E.; Jacobsen, J.H.; Zemel, R. Environment inference for invariant learning. In Proceedings of the ICML Workshop on Uncertainty and Robustness, Virtually, 17 July 2020.
- Dawid, A.P. Causal inference without counterfactuals. *J. Am. Stat. Assoc.* **2000**, *95*, 407–424. [[CrossRef](#)]
- Rubin, D.B. Causal inference using potential outcomes: Design, modeling, decisions. *J. Am. Stat. Assoc.* **2005**, *100*, 322–331. [[CrossRef](#)]



33. Robins, J.M.; Hernon, M.A.; Brumback, B. Marginal structural models and causal inference in epidemiology. *Epidemiology* **2000**, *11*, 550–560. [[CrossRef](#)]
34. Pearl, J. *Causality: Models, Reasoning, and Inference*; Cambridge University Press: Cambridge, UK, 2009.
35. Greenl, S.; Pearl, J.; Robins, J.M. Causal diagrams for epidemiologic research. *Epidemiology* **1999**, *10*, 37–48. [[CrossRef](#)]
36. Spirtes, P. *Single World Intervention Graphs (SWIGs): A Unification of the Counterfactual and Graphical Approaches to Causality*; Center for the Statistics and the Social Sciences, University of Washington Series, Working Paper 128; Now Publishers Inc.: Norwell, MA, USA, 2013.
37. Spirtes, P.; Glymour, C.N.; Scheines, R. *Causation, Prediction, and Search*; MIT Press: Cambridge, MA, USA, 2000.
38. Hair, J.F., Jr.; Sarstedt, M. Data, measurement, and causal inferences in machine learning: Opportunities and challenges for marketing. *J. Mark. Theory Pract.* **2021**, *29*, 65–77. [[CrossRef](#)]
39. Br, J.E.; Zhou, X.; Xie, Y. Recent developments in causal inference and machine learning. *Annu. Rev. Sociol.* **2023**, *49*, 81–110.
40. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. In Proceedings of the International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 April 2014.
41. Hoffman, M.D.; Johnson, M.J. Elbo surgery: Yet another way to carve up the variational evidence lower bound. In Proceedings of the Workshop in Advances in Approximate Bayesian Inference, Barcelona, Spain, 9 December 2016.
42. Tomczak, J.; Welling, M. Vae with a vampprior. In Proceedings of the International Conference on Artificial Intelligence and Statistics, PMLR 2018, Playa Blanca, Spain, 9–11 April 2018.
43. Dinh, L.; Krueger, D.; Bengio, Y. Nice: Non-linear independent components estimation. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
44. Kim, Y. Convolutional Neural Networks for Sentence Classification. *arXiv* **2014**, arXiv:1408.5882.
45. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30: In Proceedings of the Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017*; Neural Information Processing Systems Foundation, Inc.: Jolla, CA, USA, 2017; p. 30.
46. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
47. He, Y.; Shen, Z.; Cui, P. Towards non-i.i.d. image classification: A dataset and baselines. *Pattern Recognit.* **2019**, *110*, 107383. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.