ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240855

Revisiting the Dataset Bias Problem from a Statistical Perspective

Kien Do, Dung Nguyen, Hung Le, Thao Le, Dang Nguyen, Haripriya Harikumar, Truyen Tran, Santu Rana and Svetha Venkatesh

Applied Artificial Intelligence Institute (A2I2), Deakin University, Australia
{k.do, dung.nguyen, thai.le, thao.le, d.nguyen, h.harikumar, truyen.tran, santu.rana, svetha.venkatesh}@deakin.edu.au
ORCID (Kien Do): https://orcid.org/0000-0002-0119-122X

Abstract. In this paper, we study the "dataset bias" problem from a statistical standpoint, and identify the main cause of the problem as the strong correlation between a class attribute u and a non-class attribute b in the input x, represented by $p(u|b)$ differing significantly from $p(u)$. Since $p(u|b)$ appears as part of the sampling distributions in the standard maximum log-likelihood (MLL) objective, a model trained on a biased dataset via MLL inherently incorporates such correlation into its parameters, leading to poor generalization to unbiased test data. From this observation, we propose to mitigate dataset bias via either weighting the objective of each sample n by $\frac{1}{p(u_n|b_n)}$ or sampling that sample with a weight proportional to $\frac{1}{p(u_n|b_n)}$. While both methods are statistically equivalent, the former proves more stable and effective in practice. Additionally, we establish a connection between our debiasing approach and causal reasoning, reinforcing our method's theoretical foundation. However, when the bias label is unavailable, computing $p(u|b)$ exactly is difficult. To overcome this challenge, we propose to approximate $\frac{1}{p(u|b)}$ using a biased classifier trained with "bias amplification" losses. Extensive experiments on various biased datasets demonstrate the superiority of our method over existing debiasing techniques in most settings, validating our theoretical analysis.

1 Introduction

In recent years, Deep Neural Networks (DNNs) have achieved remarkable performance in Computer Vision and Natural Language Processing tasks. This success can be attributed to their capability of capturing various patterns in the training data that are indicative of the target class. However, when the training data exhibits strong correlation between a non-class attribute and the target class (often referred to as *"dataset bias"* [3, 4, 18, 23]), DNNs may overly rely on the non-class attribute instead of the actual class attribute, especially if the non-class attribute is easier to learn [23]. This leads to biased models that struggle to generalize to new scenarios where the training bias is absent. For instance, consider a dataset of human face images where men typically have black hair and women usually have blond hair. If we train a DNN on this dataset for gender classification, the model might take a "shortcut" and use hair color (a non-class attribute) as a primary predictor. As a result, when the model encounters a man with blond hair during testing, it erroneously predicts the individual as a woman.

To tackle the dataset bias problem, earlier approaches rely on the availability of bias labels [14]. They employ supervised learning to train a bias prediction model to capture the bias in the training data, and concurrently learn debiased features that share the smallest mutual information with the captured bias. The debiased features are then utilized for predicting the target class. On the other hand, alternative approaches relax the assumption of bias label availability and focus on specific types of bias [4, 32]. They introduce specialized network architectures to capture these specific types of bias. For example, Bahng et al. [4] leverage convolutional networks with small receptive fields to capture textural bias in images. However, acquiring human annotations for bias can be laborious, expensive, and requires expertise in bias identification, making it challenging in practical scenarios. Furthermore, bias labeling may not encompass all forms of bias present in the training data, particularly those that are continuous. As a result, recent approaches have shifted their attention to settings where no prior knowledge about bias is available [3, 13, 15, 16, 20, 23]. Many of these methods exploit knowledge from a "biased" model trained by minimizing a "bias amplification" loss [34] to effectively mitigate bias [3, 20, 23]. They have achieved significant improvement in bias mitigation, even surpassing approaches that assume bias labels. However, the heuristic nature of their bias correction formulas makes it difficult to clearly understand why these methods perform well in practice.

In this paper, we revisit the dataset bias problem from a statistical perspective, and present a mathematical representation of this bias, expressed as either $p(u|b) \neq p(u)$ or $p(b|u) \neq p(b)$ where u, b refer to the *class attribute* and *non-class (bias) attribute*, respectively. Our representation characterizes the common understanding of dataset bias as *"high correlation between the bias attribute and class attribute"* [4, 20]. In addition, we demonstrate that dataset bias arises naturally within the standard maximum log-likelihood objective as part of the sampling distribution, alongside the "imbalance bias". Building on this insight, we propose two approaches to mitigate dataset bias: weighting the loss of each sample n by $\frac{1}{p(u_n|b_n)}$, or sampling the sample with a weight proportional to $\frac{1}{p(u_n|b_n)}$. Through empirical analysis, we highlight the distinct behaviors of these methods, despite their statistical equivalence. Furthermore, we offer an intriguing perspective on dataset bias as a "confounding bias" in causal reasoning, and theoretically show that our method actually learns the causal relationship between the target class y and the class at-

tribute u via minimizing an upper bound of the expected *negative interventional log-likelihood* $\mathbb{E}_n\left[-\log p_\theta(y_n|\text{do}(u_n))\right]$. However, accurately computing $\frac{1}{p(u_n|b_n)}$ or $p(u_n|b_n)$ poses a significant challenge when b_n is unknown and intertwined with u_n in the input x_n. To address this issue, we propose an alternative approach that approximates $p(y_n|b_n)$, a proxy for $p(u_n|b_n)$, using a biased classifier trained with "bias amplification" losses [23]. Our intuition is that if the biased classifier $p_\psi(y_n|x_n)$ is properly trained to use only the bias attribute b_n in the input x_n for predicting y_n, then $p_\psi(y_n|x_n)$ can serve as a reasonable approximation of $p(y_n|b_n)$.

We conduct comprehensive experiments on four popular biased datasets that encompass various forms of bias: Colored MNIST, Corrupted CIFAR10, Biased CelebA, and BAR [23]. Experimental results show that our method achieves superior bias mitigation results compared to many existing baselines. This validates the soundness of our theoretical analysis and demonstrates the effectiveness of our method in mitigating bias, especially when no bias label is available. Additionally, our ablation studies reveal alignments between the optimal configurations of our method and the values indicated by our theoretical analysis on simple datasets like Colored MNIST.

2 Related Work

Methods for mitigating dataset bias can be broadly categorized into two groups: i) those that utilize the bias label or prior knowledge about bias, and ii) those that do not. The two groups are discussed in detail below.

Debiasing given the bias label or certain types of bias When the bias label is available, a straightforward approach is to train a bias prediction network or a *"biased"* network in a supervised manner. The bias knowledge acquired from the biased network can then be utilized as a form of regularization to train another *"debiased"* network. One commonly used regularization strategy involves minimizing the mutual information between the biased and debiased networks through adversarial training [4, 14, 36]. This compels the debiased network to learn features independent of the bias information, which are considered unbiased. In the model proposed by [14], a "biased" head is positioned on top of a "debiased" backbone with a gradient reversal layer [7] in between to facilitate adversarial learning. The backbone is trained to trick the biased head into predicting incorrect bias labels while the biased head attempts to make correct bias predictions. Other works, such as [4, 32], do not make use of the bias label; rather, they assume that image texture is the main source of bias. This comes from the observation that outputs of deep neural networks depend heavily on superficial statistics of the input image such as texture [8]. The framework in [32] consists of two branches: a conventional CNN for encoding visual features from the input image, and a set of learnable gray-level co-occurrence matrices (GLCMs) for extracting the textural bias information. Besides adversarial regularization, the authors of [32] introduce another regularization technique known as HEX, which projects the CNN features into a hidden space so that the projected vectors contain minimal information about the texture bias captured by the GLCM branch. [4], on the other hand, use a CNN with small receptive fields as the biased network, and the Hilbert-Schmidt Independence Criterion (HSIC) as a measure of mutual information between the biased and debiased classifiers' features. [12] draw a probabilistic connection between data generated by $p(y)p(b)$ and by $p(y|b)p(b)$, and utilize the assumption that $p(x|y,b)$ remains unchanged regardless of the change in $p(y,b)$ to derive an effective bias correction method called BiasBal. They also propose BiasCon - a debiasing method based on contrastive learning. In the case the bias label is not provided, they assume the bias is texture and make use of the biased network proposed in [4]. EnD [31] employs a regularization loss for debiasing that comprises two terms: a "disentangling" term that promotes the decorrelation of samples with similar bias labels, and an "entangling" term which forces samples belonging to the same class but having different bias labels to be correlated. EnD exhibits certain similarities to BiasCon [12], as the "entangling" and "disentangling" terms can be viewed as the positive and negative components of a contrastive loss, respectively.

In visual question answering (VQA), bias can arise from the co-occurrence of words in the question and answer, causing the model to overlook visual cues when making predictions [1, 2]. To overcome this bias, common approaches involve training a biased network that takes only questions as input to predict answers. The prediction from this biased network is then used to modulate the prediction of a debiased network trained on both questions and images [5, 6, 27].

Debiasing without prior knowledge about bias Due to the challenges associated with identifying and annotating bias in real-world scenarios, recent attention has shifted towards methods that do not rely on bias labels or make assumptions about specific types of bias. LfF [23] is a pioneering method in this regard. It utilizes the GCE loss [34], which is capable of amplifying the bias in the input, to train the biased classifier, thereby eliminating the need for bias labels. This strategy has been inherited and extended in numerous subsequent works [3, 13, 15, 20, 21]. [20] emphasize the importance of diversity in bias mitigation, and propose a method that augments the training data by swapping the bias features of two samples, as extracted by the biased classifier. BiaSwap [15], on the other hand, leverages SwapAE [25] and CAM [35] to generate "bias-swapped" images. SelecMix [13] applies mixup on "contradicting" pairs of samples (i.e., those having the same label but far away in the latent space, or different labels but close), and uses the mixed-up samples for training the debiased classifier. PGD [3] uses the biased network's gradient to compute the resampling weight. [21] aim to improve the biased classifier by training it using bias-aligned samples only. LWBC [16] trains a "biased committee" - a group of multiple biased classifiers - using the cross-entropy loss and knowledge distilled from the main classifier trained in parallel. Outputs from the biased classifiers are used to compute the sample weights for training the main classifier. [30] conduct an extensive empirical study about some existing bias mitigation methods, and discover that many of them are sensitive to hyperparameter tuning. Based on their findings, they suggest to adopt more rigorous assessments.

3 A Statistical View of Dataset Bias

We consider the standard supervised learning problem which involves learning a classifier $p_\theta(y|x)$, parameterized by θ, that maps an input sample x to the class probability vector. Let $\mathcal{D} := \{(x_n, y_n)\}_{n=1}^N$ denote the training dataset consisting of N samples. The typical learning strategy minimizes the expected negative log-likelihood (NLL) of y conditional on x, computed as follows:

$$\mathcal{L}_\theta^{\text{NLL}} := \mathbb{E}_{p_\mathcal{D}(x_n,y_n)}\left[-\log p_\theta(y_n|x_n)\right] \tag{1}$$

In the above equation, we intentionally include the subscript n to emphasize that x_n and y_n correspond to a particular sample n rather than being arbitrary. Without loss of generality, we assume that each input x consists of two types of attributes: the *class* attribute (denoted by u) and the *non-class* attribute (denoted by b), i.e., $x = (u, b)$. For

example, in the ColoredMNIST dataset [4, 23], u represents the digit shape and b represents the background color. Eq. 1 can be written as:

$$\mathcal{L}_\theta^{\text{NLL}} = \mathbb{E}_{p_\mathcal{D}(x_n)p_\mathcal{D}(y_n|x_n)}\left[-\log p_\theta(y_n|x_n)\right] \quad (2)$$
$$= \mathbb{E}_{p_\mathcal{D}(x_n)}\left[\mathcal{L}_\theta^{\text{xent}}(x_n, y_n)\right] \quad (3)$$
$$= \mathbb{E}_{p(u_n)p(b_n|u_n)}\left[\mathcal{L}_\theta^{\text{xent}}(x_n, y_n)\right] \quad (4)$$
$$= \mathbb{E}_{p(b_n)p(u_n|b_n)}\left[\mathcal{L}_\theta^{\text{xent}}(x_n, y_n)\right] \quad (5)$$

where $\mathcal{L}_\theta^{\text{xent}}$ denotes the cross-entropy loss. Intuitively, if u is distinctive among classes, u can be generally treated as a categorical random variable. In this case, we will have $p(u_n) \approx p(y_n)$ and $p(b_n|u_n) \approx p(b_n|y_n)$. From Eq. 4, it is clear that there are two main sources of bias in the training data. One comes from the non-uniform distribution of the class attribute (i.e., $p(u)$ is not uniform among classes), and the other comes from the strong correlation between the non-class attribute b and the class attribute u (i.e., $p(b|u)$ is very different from $p(b)$). A highly correlated non-class attribute will cause the model to depend more on b and less on u to predict y. The former is commonly known as the *"class-imbalance bias*", while the latter is often referred to as the *"dataset bias"* [3, 4, 18, 23]. In this paper, we focus exclusively on addressing the dataset bias due to its difficulty, especially when no prior knowledge about the bias is available. Besides, we decide to rename the dataset bias as *"feature-correlation bias"* since we believe this name better characterizes the property of the bias. We also refer to b as a *bias attribute* because it is the primary factor contributing to the bias. In the next section, we will discuss in detail our methods for mitigating the feature-correlation bias.

4 Mitigating Dataset Bias from Statistical and Causal Perspectives

4.1 Bias mitigation based on $p(u|b)$

Eq. 5 suggests that we can mitigate the dataset bias (or feature-correlation bias) by either weighting the individual loss $\mathcal{L}_\theta^{\text{xent}}(x_n, y_n)$ by $\frac{1}{p(u_n|b_n)}$ during training or sampling each data point (x_n, y_n) with the weight proportional to $\frac{1}{p(u_n|b_n)}$. We refer to the two techniques as *loss weighting* (LW) and *weighted sampling* (WS), respectively. The two techniques are statistically equivalent, and transform the objective in Eq. 5 into $\mathbb{E}_{p(b_n)}\left[\mathcal{L}_\theta^{\text{xent}}(x_n, y_n)\right]$, which no longer contains $p(u_n|b_n)$.

We can approximate $p(u_n|b_n)$ using its proxy $p(y_n|b_n)$. However, explicitly modeling $p(y|b)$ pose challenges due to the typical unknown nature of b. Meanwhile, modeling $p(y|x)$ (or $p(y|u, b)$) is straightforward. Therefore, we propose to model $p(y|b)$ indirectly through $p(y|x)$ by training a parameterized model $p_\psi(y|x)$ in a manner that amplifies the influence of the bias attribute b (in x) on y. We refer to $p_\psi(y|x)$ as the *biased classifier* and train it for T_{bias} epochs using a bias amplification loss. Specifically, we choose the generalized cross-entropy (GCE) loss $\mathcal{L}_\psi^{\text{GCE}}(x_n, y_n) = \frac{1-p_\psi(y_n|x_n)^\tau}{\tau}$ [34] where $\tau \in (0, 1]$ is a hyperparameter controlling the degree of amplification. Once $p_\psi(y|x)$ has been trained, we can compute the weight for sample n as follows:

$$w_n = \min\left(\frac{1}{p_\psi(y_n|x_n)}, \gamma\right) \quad (6)$$

where $\gamma > 0$ is a clamp hyperparameter that prevents w_n from becoming infinite when $p_\psi(y_n|x_n)$ is close to 0. Since $p_\psi(y_n|x_n) \in [0, 1]$, $w_n \in [1, \gamma]$. w_n can be considered as an approximation of $\frac{1}{p(u_n|b_n)}$. For the debiasing purpose, we can train $p_\theta(y|x)$ via either loss weighting (LW) or weighted sampling (WS) with the weight w_n. In the case of LW, the debiasing loss becomes:

$$\mathcal{L}_\theta^{\text{LW}} = \mathbb{E}_{p(b_n)p(u_n|b_n)}\left[\frac{\mathcal{L}_\theta^{\text{xent}}(x_n, y_n)}{p(u_n|b_n)}\right] \quad (7)$$
$$\approx \mathbb{E}_{p_\mathcal{D}(x_n)}\left[w_n \cdot \mathcal{L}_\theta^{\text{xent}}(x_n, y_n)\right] \quad (8)$$

Although LW and WS are statistically equivalent, they perform differently in practice. LW preserves the diversity of training data but introduces different scales to the loss. To make training with LW stable, we rescale w_n so that its maximum value is not γ (which could be thousands) but a small constant value, which is 10 in this work. This means w_n lies in the range $\left[\frac{10}{\gamma}, 10\right]$. WS, by contrast, maintains a constant scale of the loss but fails to ensure the diversity of training data due to over/under-sampling. During our experiment, we observed that LW often yields better performance than WS, which highlights the importance of data diversity.

However, if we simply fix the sample weight to be w_n throughout training, a classifier trained via LW will take long time to achieve good results, and the results will be not optimal in some cases. It is because bias-aligned (BA) samples, which dominates the training data, have very small weights. The classifier will spend most of the training time performing very small updates on these BA samples, and thus, struggles to capture useful information in the training data. To deal with this problem, we propose a simple yet effective annealing strategy for LW. We initially set the weights of all training samples to the same value β and linearly transform β to w_n for T_{anneal} steps. Mathematically, the weight for sample n at step t is $w_n(t) = \begin{cases} \beta + \frac{t(w_n-\beta)}{T_{\text{anneal}}} & \text{if } 0 < t < T_{\text{anneal}} \\ w_n & \text{otherwise} \end{cases}$ and the loss for annealed loss weighting (ALW) is given below:

$$\mathcal{L}_\theta^{\text{ALW}}(t) = \mathbb{E}_{p_\mathcal{D}(x_n)}\left[w_n(t) \cdot \mathcal{L}_\theta^{\text{xent}}(x_n, y_n)\right] \quad (9)$$

4.2 Interpretation of $\mathcal{L}_\theta^{LW}$ from a causal perspective

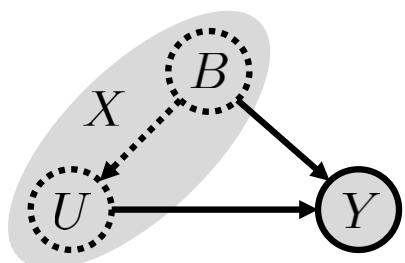


Figure 1. A causal graphical model representing the feature-correlation bias problem. X, Y are the input image and class label, respectively. Both are *observed* (marked with shaded background) during training. U, B are the *hidden* class and non-class attributes of X, respectively.

Interestingly, we can interpret the debiasing loss $\mathcal{L}_\theta^{\text{LW}}$ in the language of causal reasoning by utilizing the Potential Outcomes framework [28] illustrated in Fig. 1, where the class label Y, class attribute U, and non-class attribute B play the roles of the outcome, treatment, and confounder, respectively. In our setting, both U and B are hidden but can be accessed through the observed input X. When the unconfoundedness and positivity assumptions (i.e. the backdoor assumptions) [33] are met, we can estimate the causal quantity $p(y|\text{do}(u))$

Dataset	BC (%)	Vanilla	LNL	BiasBal	WS	LW
Colored MNIST	0.5	80.18±1.38	80.05±1.02	**97.44±0.12**	96.15±0.33	96.16±0.35
	1.0	87.48±1.75	87.52±1.53	**97.89±0.20**	97.58±0.09	97.11±0.12
	5.0	97.04±0.21	98.77±0.18	98.92±0.13	**98.92±0.07**	98.55±0.07
Corrupted CIFAR10	0.5	28.00±1.15	28.13±0.98	**45.52±1.37**	31.02±0.72	36.50±0.51
	1.0	34.56±0.87	34.43±1.03	**53.18±1.87**	40.06±1.24	46.41±1.46
	5.0	59.33±1.26	59.52±0.85	**72.60±0.69**	69.91±0.54	70.29±0.75
Biased CelebA	0.5	77.43±0.42	76.64±0.73	**87.74±0.19**	78.48±0.62	86.49±0.33
	1.0	80.58±0.41	79.72±0.54	**87.82±0.36**	82.41±0.33	85.32±0.40
	5.0	86.35±0.33	86.41±0.35	**88.78±0.22**	86.90±0.23	87.83±0.21

Table 1. Test accuracies of debiasing methods when the bias label is *available*. The best and second best results are highlighted in bold and gray, respectively.

from observational data via backdoor adjustment [26] as follows:

$$p(y|\text{do}(u)) = \sum_b \frac{p(y,u,b)}{p(u|b)} \tag{10}$$

$$= \mathbb{E}_{p(u,b)}\left[\frac{p(y|u,b)}{p(u|b)}\right] \tag{11}$$

$$= \mathbb{E}_{p(b)}\left[p(y|u,b)\right] \tag{12}$$

Eq. 11 is typically known as Inverse Probability Weighting (IPW), where $p(u|b)$ is called the propensity score [11, 10]. In Eq. 11, the class prediction $p(y|u,b)$ is weighted by the inverse propensity score $\frac{1}{p(u|b)}$, exhibiting a degree of resemblance to our loss $\mathcal{L}_\theta^{\text{LW}}$ in Eq. 7. It suggests that we can interpret $\mathcal{L}_\theta^{\text{LW}}$ from a causal standpoint. In fact, $\mathcal{L}_\theta^{\text{LW}}$ acts as an upper bound of the expected *negative interventional log-likelihood* (NILL) $\mathcal{L}_\theta^{\text{NILL}} = \mathbb{E}_n\left[-\log p_\theta(y_n|\text{do}(u_n))\right]$ [24]. The relationship between these two losses is provided below:

$$\mathcal{L}_\theta^{\text{NILL}} = \mathbb{E}_n\left[-\log p_\theta(y_n|\text{do}(u_n))\right] \tag{13}$$

$$\leq \mathbb{E}_{p(u_n,b_n,y_n)}\left[\frac{-\log p_\theta(y_n|u_n,b_n)}{p(u_n|b_n)}\right] \tag{14}$$

$$= \mathbb{E}_{p(u_n,b_n)}\left[\frac{\mathcal{L}_\theta^{\text{xent}}(x_n,y_n)}{p(u_n|b_n)}\right] = \mathcal{L}_\theta^{\text{LW}} \tag{15}$$

where the inequality in Eq. 14 is derived from:

$$-\log p_\theta(y|\text{do}(u)) = -\log \mathbb{E}_{p(b)}\left[p_\theta(y|u,b)\right] \tag{16}$$

$$\leq -\mathbb{E}_{p(b)}\left[\log p_\theta(y|u,b)\right] \tag{17}$$

$$= \mathbb{E}_{p(u,b)}\left[\frac{-\log p_\theta(y|u,b)}{p(u|b)}\right] \tag{18}$$

Eq. 17 is the Jensen inequality with equality attained when $p_\theta(y|u,b) = p_\theta(y|u,b')\ \forall\ b,b'$, i.e. y is independent of b given u. This condition matches our target of learning an unbiased classifier $p_\theta(y|u,b)$. In Eq. 18, $p(u|b)$ is introduced to allow (u,b) to be sampled jointly from observational data. Eq. 14 can be viewed as a Monte Carlo estimation of Eq. 18 (or Eq. 17) using a single sample of b, i.e. b_n. From Eqs. 13 - 18, we see that minimizing $\mathcal{L}_\theta^{\text{LW}}$ also minimizes $\mathcal{L}_\theta^{\text{NILL}}$ and encourages $p_\theta(y|u,b)$ to be close to $p_\theta(y|u)$. Minimizing $\mathcal{L}_\theta^{\text{NILL}}$ causes the model to focus more (less) on uncommon (common) samples n which has small (big) $p(u_n|b_n)$.

4.3 Bias mitigation based on $p(b|u)$

Eq. 4 suggests an alternative approach to mitigating the feature-correlation bias, which involves weighting each individual sample n by $\frac{1}{p(b_n|u_n)}$ rather than $\frac{1}{p(u_n|b_n)}$. However, accurately estimating $p(b|u)$ poses challenges in practical implementations. In Appdx. 4.1, we present an idea about using conditional generative models to approximate $p(b|u)$ and discuss its limitations.

5 Experiments

5.1 Experimental Setup

5.1.1 Datasets

We evaluate our proposed methods on 4 popular datasets for bias correction, namely Colored MNIST [4, 23], Corrupted CIFAR10 [23], Biased CelebA [23, 29], and BAR [23]. Details about these datasets are provided below and in Appdx. B.1.

In Colored MNIST, the target attribute is the digit, while the bias attribute is the background color. In Corrupted CIFAR10, the target attribute is the object, and the bias attribute is the corruption noise. We created Colored MNIST and Corrupted CIFAR10 from the standard MNIST [19] and CIFAR10 [17] datasets respectively using the official code provided by the authors of [23] with some slight modifications. Specifically, we used distinctive background colors for Colored MNIST, and set the severity of the corruption noise to 2 for Corrupted CIFAR10 to retain enough semantic information for the main task. Following [23], we created 3 versions of Colored MNIST and Corrupted CIFAR10 with 3 different bias-conflicting ratios (BC ratios) which are 0.5%, 1%, and 5%.

In Biased CelebA, the hair color (*blond(e)* or *not blond(e)*) serves as the target attribute, while the gender (*male* or *female*) is considered the bias attribute. Individuals with blond(e) hair exhibit a bias toward being female, whereas those without blond(e) hair are biased toward being male. We created Biased CelebA ourselves by selecting a random subset of the original training samples from the CelebA dataset [22] to ensure a certain BC ratio is achieved. We consider 3 BC ratios of 0.5%, 1%, and 5%. Each BC ratio is associated with a specific number of BC samples per target class, which is 100, 200, and 500, respectively. As a result, the training set for Biased CelebA comprises a total of 39998, 39998, and 19998 samples for the BC ratios of 0.5%, 1%, and 5%, respectively.

BAR is a dataset for action recognition which consists of 6 action classes, namely *climbing*, *diving*, *fishing*, *racing*, *throwing*, and *vaulting*. The bias in this dataset is the place where the action is performed. For example, climbing is usually performed on rocky mountains, or diving is typically practiced under water. This dataset does not have bias labels. We use the default train/valid/test splits provided by the authors [23].

5.1.2 Baselines

We conduct a comprehensive comparison of our method with popular and up-to-date baselines for bias correction [3, 4, 12, 13, 20, 23]. The selected baselines encompass a diverse range of approaches including information-theoretic-based methods [4], loss weighting techniques [23], weighted sampling strategies [3], mix-up approaches [13], and BC samples synthesis methods [20]. Some of them [23, 3]

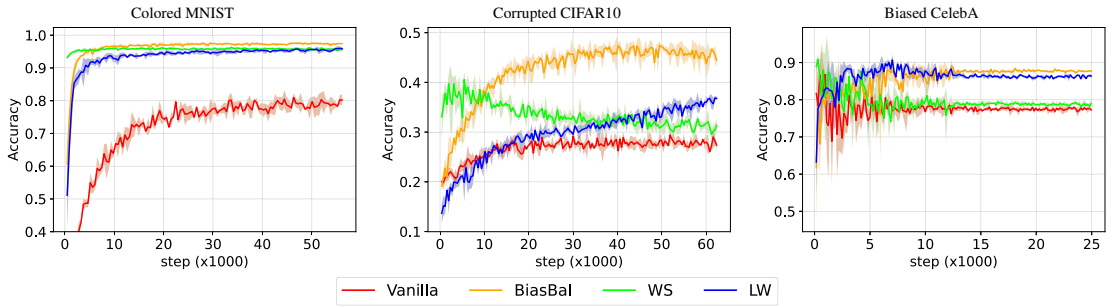


Figure 2. Test accuracy curves of the vanilla classifier, BiasBal, and our proposed WS and LW on Colored MNIST, Corrupted CIFAR10, and Biased CelebA with the BC ratio of 0.5%. The bias label is assumed to be given, and $p(y|b)$ used in BiasBal, WS, and LW is estimated directly from training data.

Dataset	BC (%)	Vanilla	ReBias	LfF	DFA	SelecMix	PGD	LW (Ours)
Colored MNIST	0.5	80.18±1.38	74.85±1.97	93.38±0.52	91.85±0.92	83.41±1.26	**96.15±0.28**	95.57±0.41
	1.0	87.48±1.75	84.23±1.56	94.09±0.78	94.32±0.89	91.59±0.99	**97.93±0.19**	97.18±0.34
	5.0	97.04±0.21	95.76±0.50	97.40±0.25	96.74±0.43	97.37±0.15	**98.74±0.12**	98.61±0.09
Corrupted CIFAR10	0.5	28.00±1.15	-	41.95±1.56	40.54±1.98	31.67±0.90	44.89±1.36	**45.76±1.49**
	1.0	34.56±0.87	-	**53.36±1.87**	50.27±0.94	36.28±1.22	47.38±1.01	51.64±1.12
	5.0	59.33±1.26	-	70.04±1.05	67.05±1.82	63.15±1.17	63.60±0.58	**70.45±1.24**

Table 2. Test accuracies of different debiasing methods on Colored MNIST and Corrupted CIFAR10 when the bias label is unavailable. The best and second best results are highlighted in bold and gray, respectively.

Dataset	BC (%)	Vanilla	LfF	PGD	LW (Ours)
Biased CelebA	0.5	77.43±0.42	77.81±1.01	78.07±2.18	**87.54±0.32**
	1.0	80.58±0.41	85.54±1.27	79.26±0.88	**86.38±0.37**
	5.0	86.35±0.33	80.22±1.58	83.47±0.95	**87.43±0.34**
BAR	-	68.45±0.32	62.09±0.21	70.49±0.65	**71.24±0.53**

Table 3. Test accuracies of different debiasing methods on Biased CelebA and BAR when the bias label is unavailable. The best and second best results are highlighted in bold and gray, respectively.

are closely related to our methods, and will receive in-depth analysis. To establish a fair playing ground, we employ *identical* classifier architectures, data augmentations, optimizers, and learning rate schedules for both our methods and the baselines. We also search for the learning rates that lead to the best performances of the baselines. For other hyperparameters of the baselines, we primarily adhere to the default settings outlined in the original papers. Details about these settings are provided below and in Appdx. B.

5.1.3 Implementation details

We implement the classifier using a simple convolutional neural network (CNN) for Colored MNIST, a small ResNet18 for Corrupted CIFAR10, and the standard ResNet18 [9] for Biased CelebA and BAR. The CNN used for Colored MNIST is adapted from the code provided in [3]. The standard ResNet18 architecture is sourced from the torchvision library. Given that the input size for Biased CelebA is 128×128, we simply replace the first convolution layer of the standard ResNet18, which originally has a stride of 2, with another convolution layer having a stride of 1.

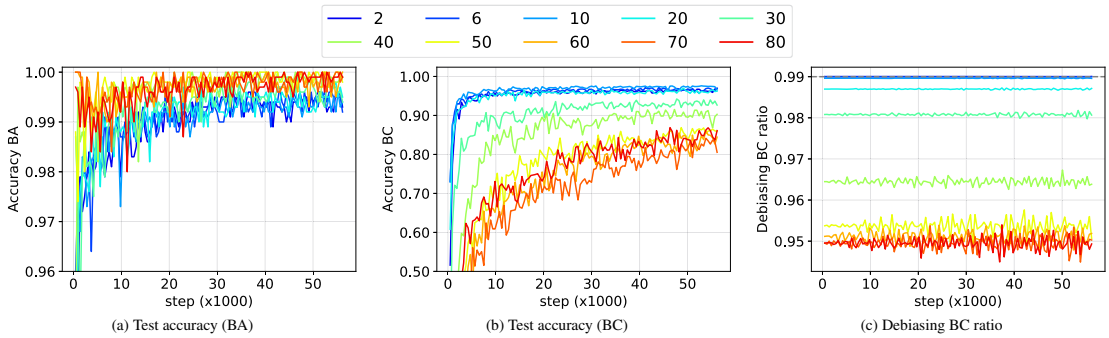
Following [12, 13, 20, 23], we augment the input image with random horizontal flip, random crop, and random resized crop, depending on the dataset (details in Appdx. B.3). Unlike [3], we choose *not* to employ color jitter as a data augmentation technique. This deliberate decision is based on the understanding that such augmentation has the potential to eliminate specific types of bias present in the input image, thereby bolstering the classifier's robustness without necessitating any additional bias mitigation techniques. Consequently, it is challenging to ascertain whether the observed performance improvements of a bias mitigation method genuinely stem from its inherent capabilities or simply result from the applied augmentation, especially on datasets having color bias like Colored MNIST.

We provide details for the optimizer, training epochs, learning rate, etc. corresponding to each dataset in Appdx. B.2.

5.2 Results when the bias label is available

In this section, we aim to empirically validate our theoretical analysis on feature-correlation bias in Sections 3 and 4 by examining the effectiveness of our proposed weighted sampling (WS) and loss weighting (LW) techniques in mitigating bias in datasets with known discrete bias labels. We consider Learning Not to Learn (LNL) [14] and Bias Balancing (BiasBal) [12] as baselines since these approaches explicitly utilize the bias label. From Table 1, it is evident that classifiers trained with either WS or LW achieve significantly higher test accuracies than the vanilla classifier. This observation suggests that WS and LW effectively reduce bias in training data, validating our theoretical analysis. However, WS and LW exhibit distinct training dynamics, resulting in varying levels of effectiveness in bias mitigation. As depicted in Fig. 2, on Corrupted CIFAR10, WS quickly achieves good performance in the early stages of training but gradually degrades over time. We hypothesize that this behavior is attributed to the lack of data diversity resulting from over/under-sampling of BC/BA samples in WS. On the other hand, LW takes more time than WS to achieve similar performance due to its small updates on most training samples (i.e., BA samples). Nonetheless, LW demonstrates a steady improvement in performance.

Besides, WS and LW outperform LNL, a method that aims to learn features independent of bias information. Interestingly, during our experiments, we observed minimal performance difference between LNL and the vanilla classifier, as shown in Table 1. This suggests that the features learned by LNL still contain bias information. We hypothesize that when class and bias labels are highly correlated, it



(a) Test accuracy (BA) (b) Test accuracy (BC) (c) Debiasing BC ratio

Figure 3. Learning curves of LW w.r.t. different training epochs of the biased classifier (T_{bias}), ranging from 2 to 80. The biased dataset is Colored MNIST with the BC ratio of 1%. The maximum sample weight is set to 100. Since BC samples account for 90% of the total test samples, the test accuracies for all samples are very similar to those for BC sample in (b).

is challenging from a statistical perspective to learn features that are truly independent of bias while accurately predicting the class.

However, our methods perform worse and are less robust than BiasBal, especially on Corrupted CIFAR10. Fig. 2 illustrates that BiasBal converges more rapidly than our methods with higher test accuracies and smaller test crossentropy losses in most settings. The exceptional performance of BiasBal motivates us to delve deeper into the theoretical mechanisms and design differences between BiasBal and our methods that account for BiasBal's superior performance.

We discovered that like LW, BiasBal mitigates bias via reweighting the model trained on biased data. It also arrives at the same reweighting term of $\frac{1}{p(y|b)}$ as in LW despite employing a different reasoning technique that leverages Bayes' rule and two assumptions: i) $p(y|x) = p(y|x,b)$ and ii) $p(x|y,b)$ are the same across different joint distributions $p(y,b)$. The bias correction formula of BiasBal (Eq. A.8 in the supplementary material of [12]) is given as follows:

$$p_{train}(y|x) = p_u(y|x)p_{train}(y|b)\frac{K}{C} \tag{19}$$

where p_{train} and p_u denotes the distribution over the biased training data and the ideal unbiased distribution, respectively; C is the number of classes, and $K = \frac{p_u(x|b)}{p_{train}(x|b)}$ is a constant w.r.t. y. Eq. 19 implies that $p_u(y|x) \propto p_{train}(y|x)\frac{1}{p_{train}(y|b)}$, allowing us to reweight the biased distribution $p_{train}(y|x)$ by $\frac{1}{p_{train}(y|b)}$ to achieve a theoretically unbiased distribution.

The key distinction between BiasBal and WS/LW lies in their respective debiasing mechanisms. While WS/LW *corrects the learning process* of $p_\theta(y|x)$ (via resampling/reweighting the training data distribution) to achieve an unbiased target $p_\theta(y|x)$ *indirectly*, BiasBal *directly corrects the target* $p_\theta(y|x)$ and learns with the (unnormalized) bias-adjusted target $\hat{p}_\theta(y|x) = p_\theta(y|x)p_{train}(y|b)$. This debiasing mechanism of BiasBal is generally more robust than that of WS/LW, as it neither compromises the diversity of training data nor introduces training instability. Moreover, while WS and LW focus on correcting bias in $p(y_n|x_n)$ for the class y_n associated with a particular input x_n, BiasBal considers bias correction of $p(y|x_n)$ for every class y. These advancements in the debiasing mechanism of BiasBal likely contribute to its superior performance.

It is worth noting that the original paper [12] on BiasBal exclusively applies the method when the bias label is discrete and known, as $p_{train}(y|b)$ can be estimated nonparametrically from the training data. However, with our proposed method for approximating $p_{train}(y|b)$ using the biased classifier, as discussed in Section 4.1, we can readily extend BiasBal to scenarios where *the bias label is either unavailable or continuous*. In Appdx. D.4, we provide a detailed analysis of this extended version of BiasBal, which we refer to as *"Target Bias Adjustment"* (TBA) to better describe its characteristic of adjusting the target distribution.

5.3 Results when the bias label is unavailable

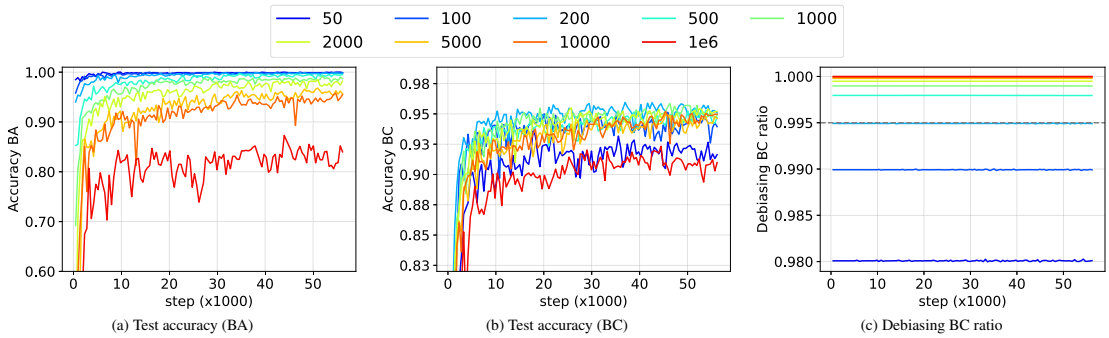
5.3.1 Results on Colored MNIST and Corrupted CIFAR10

As shown in Table 2, our proposed method LW significantly outperforms the vanilla classifier trained with the standard cross-entropy loss, as well as several other debiasing baselines such as ReBias [4], DFA [20], and SelecMix [13]. Furthermore, LW achieves higher test accuracies than LfF in most settings of Colored MNIST and Corrupted CIFAR10. Compared to the current state-of-the-art debiasing method PGD, LW performs slightly worse on Colored MNIST but demonstrates superior performance on Corrupted CIFAR10. Specifically, LW achieves about 1%, 4%, and 7% higher accuracy than PGD on Corrupted CIFAR10 with BC ratios of 0.5%, 1%, and 5%, respectively. Detailed comparisons between our method and LfF and PGD are provided in Appds. D.2 and D.1, respectively. These outcomes substantiate the efficacy of using the GCE loss to train a model of $p(y|x)$ that approximates $p(y|b)$, and endorse the practice of weighting each sample with $\frac{1}{p(y|b)}$ in our method.

5.3.2 Results on Biased CelebA and BAR

In this experiment, we only choose LfF and PGD as our baselines since the two methods have demonstrated superior performances compared to other approaches in our previous experiment and also in [3]. From Table 3, it is clear that LW outperforms LfF and PGD significantly on both Biased CelebA and BAR. Surprisingly, our experiments have revealed that in certain settings, LfF and PGD may perform even worse than the vanilla classifier, particularly on Biased CelebA with BC ratios of 1% and 5%. Additionally, LfF and PGD exhibit sensitivity to hyperparameters on Biased CelebA, as evidenced by the large standard deviations in their results. These findings raise concerns about the effectiveness of the debiasing formulas employed by LfF and PGD, which rely heavily on heuristics.

It is worth noting that on BAR, our implementation of the vanilla classifier achieves a higher test accuracy than what has been reported in [23] and [3]. This implies that the improvement of LW over the vanilla classifier on BAR can be attributed to its debiasing capability rather than the under-performance of the vanilla classifier. As far as



(a) Test accuracy (BA) (b) Test accuracy (BC) (c) Debiasing BC ratio

Figure 4. Learning curves of LW w.r.t. different values of the maximum sample weight (γ), ranging from 50 to 10^6. The biased dataset is Colored MNIST with the BC ratio of 0.5%. The biased classifier is trained for 10 epochs. Since BC samples account for 90% of the total test samples, the test accuracies for all samples are very similar to those for BC sample in (b).

we know, the result of LW on BAR presented in Table 3 currently represents the state-of-the-art performance in this domain.

5.4 Ablation Study

In this section, we closely examine two key hyperparameters that mainly influence the performance of our proposed LW. They are the number of training epochs for the biased classifier (T_{bias}), and the maximum sample weight (γ).

5.4.1 Effect of the number of training epochs for the biased classifier

Since the primary role of the biased classifier in LW is to capture solely the bias attribute b in the input x, it is critical that the biased classifier should not be trained for too long. Otherwise, this classifier may capture the class attribute u, leading to an inaccurate approximation of $p(y|b)$ by $p_\psi(y|x)$. To validate this intuition, we compare the performances of various versions of LW in Fig. 3, each utilizing a biased classifier trained for different numbers of epochs T_{bias}. Apparently, as T_{bias} decreases, LW achieves higher test accuracies on BC samples (Fig. 3b), indicating that $p_\psi(y|x)$ becomes a more reliable approximation of $p(y|b)$. However, if T_{bias} is too small (e.g., T_{bias} < 10), the biased classifier may not have undergone sufficient training to produce accurate class predictions, potentially hurting LW's performance. In fact, the choice of T_{bias} hinges on the learning capability of the biased classifier, which is examined in Appdx. D.3. Further empirical results of T_{bias} can be found in Appdx. C.

It is worth noting that there is a strong correlation between the high test accuracy of LW and the proximity of its *"debiasing BC ratio"* β to the ground-truth BA ratio (Fig. 3c). The debiasing BC ratio is computed by dividing the average loss weight over BC samples by the sum of the average loss weight over BC samples and the average loss weight over BA samples. Mathematically, $\beta = \frac{\mathbb{E}_{n\in\text{BC}}[w_n]}{\mathbb{E}_{n\in\text{BC}}[w_n]+\mathbb{E}_{n\in\text{BA}}[w_n]}$. Interestingly, in this particular setting, LW attains the highest test accuracy when *its debiasing BC ratio coincides with the ground-truth BA ratio*, which is 0.99. This remarkable alignment further corroborates our theoretical analysis, and demonstrates that the hidden true bias term $p(y|b)$ can be effectively approximated via a biased classifier trained with a limited number of epochs.

5.4.2 Effect of the maximum sample weight

The optimal value of the maximum sample weight γ plays a crucial role in ensuring the proper normalization of sample weights w_n and achieving a closer match with the true debiasing terms $\frac{1}{p(y_n|b_n)}$. Generally, the choice of γ depends primarily on the bias ratio present in the training data. To illustrate this, let's consider the Colored MNIST dataset with a BA ratio of 0.995 and a BC ratio of 0.005. In this scenario, we would expect that BC samples are weighted approximately 200 times more than BA samples to achieve full debiasing (since $\frac{0.995}{0.005} \approx 200$). This implies that if BA samples have a weight of 1, BC samples should have a weight of 200. Assuming we have a perfect biased classifier which assigns a class confidence of 1 to BA samples and very low class confidences to BC samples (i.e., $p_\psi(y|x) \approx 1$ for BA samples and small for BC samples), we can compute the weights for BA samples as 1 and for BC samples as very large weights, which are then clamped at γ. By setting γ to 200, we obtain the appropriate weights for BC samples, allowing for full debiasing as discussed earlier. Empirical validation of this reasoning is presented in Fig. 4b, where LW achieves the highest test accuracy on BC samples when γ is equal to 200. Moreover, at this specific value of γ, the debiasing BC ratio of LW aligns with the ground-truth BA ratio of 0.995 (Fig. 4c), indicating that LW likely achieves full debiasing. However, it should be noted that when the bias classifier is not perfect, the optimal value of γ would be different from the ground-truth $\frac{\text{BA ratio}}{\text{BC ratio}}$. Typically, a larger value of γ would be used to assign more weight to BC samples predicted with high class confidences by the imperfect biased classifier. Regarding the test accuracy on BA samples, it monotonically decreases as γ increases due to the widening gap between the weights of BC and BA samples (Fig. 4a). More results of γ on other datasets can be found in Appdx. C.

6 Conclusion

We have proposed a novel method for mitigating dataset bias and analyzed our method from statistical and causal perspectives. While our approach yields promising results, certain limitations persist: i) our method depends on the training epochs of the biased classifier (T_{bias}) and the clamp threshold (γ) to produce good approximations of $p(y|b)$, which are hard to control in practice due to the variability introduced by the unknown bias rate; and ii) our method mitigates bias via balancing the sampling distribution during learning rather than directly adjusting the target. Nevertheless, these limitations are not unique to our method and are shared by other debiasing techniques. For instance, PGD's dependence on T_{bias} is also evident (in the appendix). Additionally, both PGD and LfF employ indirect bias mitigation like ours. In forthcoming work, we aim to develop a novel method to address the aforementioned limitations.

Acknowledgement

This research was partially supported by the Australian Government through the Australian Research Council's Discovery Projects funding scheme (project DP210102798). The views expressed herein are those of the authors and are not necessarily those of the Australian Government or Australian Research Council.

References

[1] A. Agrawal, D. Batra, and D. Parikh. Analyzing the behavior of visual question answering models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1955–1960, 2016.

[2] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi. Dont just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4971–4980, 2018.

[3] S. Ahn, S. Kim, and S.-y. Yun. Mitigating dataset bias by using per-sample gradient. *International Conference on Learning Representations*, 2023.

[4] H. Bahng, S. Chun, S. Yun, J. Choo, and S. J. Oh. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pages 528–539. PMLR, 2020.

[5] R. Cadene, C. Dancette, M. Cord, D. Parikh, et al. Rubi: Reducing unimodal biases for visual question answering. *Advances in neural information processing systems*, 32, 2019.

[6] C. Clark, M. Yatskar, and L. Zettlemoyer. Dont take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082, 2019.

[7] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17:1–35, 2016.

[8] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] K. Hirano and G. W. Imbens. Estimation of causal effects using propensity score weighting: An application to data on right heart catheterization. *Health Services and Outcomes research methodology*, 2:259–278, 2001.

[11] K. Hirano, G. W. Imbens, and G. Ridder. Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, 71(4):1161–1189, 2003.

[12] Y. Hong and E. Yang. Unbiased classification through bias-contrastive and bias-balanced learning. *Advances in Neural Information Processing Systems*, 34:26449–26461, 2021.

[13] I. Hwang, S. Lee, Y. Kwak, S. J. Oh, D. Teney, J.-H. Kim, and B.-T. Zhang. Selecmix: Debiased learning by contradicting-pair sampling. In *Proceedings of the 36th Advances in Neural Information Processing Systems*, 2022.

[14] B. Kim, H. Kim, K. Kim, S. Kim, and J. Kim. Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9012–9020, 2019.

[15] E. Kim, J. Lee, and J. Choo. Biaswap: Removing dataset bias with bias-tailored swapping augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14992–15001, 2021.

[16] N. Kim, S. Hwang, S. Ahn, J. Park, and S. Kwak. Learning debiased classifier with biased committee. In *Proceedings of the 36th Advances in Neural Information Processing Systems*, 2022.

[17] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[18] R. Le Bras, S. Swayamdipta, C. Bhagavatula, R. Zellers, M. Peters, A. Sabharwal, and Y. Choi. Adversarial filters of dataset biases. In *International conference on machine learning*, pages 1078–1088. PMLR, 2020.

[19] Y. LeCun, C. Cortes, C. Burges, et al. Mnist handwritten digit database, 2010.

[20] J. Lee, E. Kim, J. Lee, J. Lee, and J. Choo. Learning debiased representation via disentangled feature augmentation. *Advances in Neural Information Processing Systems*, 34:25123–25133, 2021.

[21] J. Lee, J. Park, D. Kim, J. Lee, E. Choi, and J. Choo. Biasensemble: Revisiting the importance of amplifying bias for debiasing. *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, 2023.

[22] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[23] J. Nam, H. Cha, S. Ahn, J. Lee, and J. Shin. Learning from failure: Debiasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

[24] T. Nguyen, K. Do, D. T. Nguyen, B. Duong, and T. Nguyen. Causal inference via style transfer for out-of-distribution generalisation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1746–1757, 2023.

[25] T. Park, J.-Y. Zhu, O. Wang, J. Lu, E. Shechtman, A. Efros, and R. Zhang. Swapping autoencoder for deep image manipulation. *Advances in Neural Information Processing Systems*, 33:7198–7211, 2020.

[26] J. Pearl et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19(2):3, 2000.

[27] S. Ramakrishnan, A. Agrawal, and S. Lee. Overcoming language priors in visual question answering with adversarial regularization. *Advances in Neural Information Processing Systems*, 31, 2018.

[28] D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

[29] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. 2020.

[30] R. Shrestha, K. Kafle, and C. Kanan. An investigation of critical issues in bias mitigation techniques. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1943–1954, 2022.

[31] E. Tartaglione, C. A. Barbano, and M. Grangetto. End: Entangling and disentangling deep representations for bias correction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13508–13517, 2021.

[32] H. Wang, E. P. Xing, Z. He, and Z. C. Lipton. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations, ICLR 2019*, 2019.

[33] L. Yao, Z. Chu, S. Li, Y. Li, J. Gao, and A. Zhang. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–46, 2021.

[34] Z. Zhang and M. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[35] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[36] W. Zhu, H. Zheng, H. Liao, W. Li, and J. Luo. Learning bias-invariant representation by cross-sample mutual information minimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15002–15012, 2021.