Enhancing Intrinsic Features for Debiasing via Investigating Class-Discerning Common Attributes in Bias-Contrastive Pair

Jeonghoon Park*, Chaeyeon Chung*, Jaegul Choo
Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea
{jeonghoon_park, cy_chung, jchoo}@kaist.ac.kr

Abstract

In the image classification task, deep neural networks frequently rely on bias attributes that are spuriously correlated with a target class in the presence of dataset bias, resulting in degraded performance when applied to data without bias attributes. The task of debiasing aims to compel classifiers to learn intrinsic attributes that inherently define a target class rather than focusing on bias attributes. While recent approaches mainly focus on emphasizing the learning of data samples without bias attributes (i.e., bias-conflicting samples) compared to samples with bias attributes (i.e., bias-aligned samples), they fall short of directly guiding models where to focus for learning intrinsic features. To address this limitation, this paper proposes a method that provides the model with explicit spatial guidance that indicates the region of intrinsic features. We first identify the intrinsic features by investigating the class-discerning common features between a bias-aligned (BA) sample and a bias-conflicting (BC) sample (i.e., bias-contrastive pair). Next, we enhance the intrinsic features in the BA sample that are relatively under-exploited for prediction compared to the BC sample. To construct the bias-contrastive pair without using bias information, we introduce a bias-negative score that distinguishes BC samples from BA samples employing a biased model. The experiments demonstrate that our method achieves state-of-the-art performance on synthetic and real-world datasets with various levels of bias severity.

1. Introduction

Deep neural networks in image classification [5, 19, 20, 24] are known to be vulnerable to the dataset bias [22], which refers to a spurious correlation between the target classes and the peripheral attributes. Basically, image classification aims to learn intrinsic attributes — the visual features that inherently define a target class — that generally appear across the samples in the class. However, when the dataset bias exists in the training data, the models tend to use the frequently appearing peripheral attribute (*i.e.*, bias attribute) to predict the class unintentionally. For instance, if airplanes in the training images are mostly in the sky, a model can heavily rely on the sky to predict an image as an airplane class due to its high correlation with the airplane class. This indicates that the model is biased towards the bias attribute (*e.g.*, sky) rather than focusing on intrinsic features (*e.g.*, the shape of wings or the body) when making decisions. As a result, even though the biased model achieves high accuracy on the samples including bias attributes (*e.g.*, airplanes in the sky), termed as bias-aligned (BA) samples, it may fail to accurately predict samples devoid of such bias attributes (*e.g.*, airplanes on the runway), referred to as bias-conflicting (BC) samples.

In this regard, debiasing aims to encourage the model to focus on intrinsic attributes rather than bias attributes when dataset bias exists. One straightforward approach is utilizing prior knowledge regarding bias (*e.g.*, labels for bias attribute) to inform the model which attributes to focus on or not to focus on [2, 8, 21, 23]. However, acquiring such bias information is often infeasible in real-world scenarios. Therefore, recent studies [7, 11–14] have proposed debiasing methods that do not require bias information. They identify and emphasize BC samples during the training using an additional biased classifier that mainly learns the bias attributes. However, such a training strategy fails to directly indicate where the model should focus to learn the intrinsic features.

To address this issue, we present a debiasing approach that explicitly informs the model of the region of the intrinsic features during the training while not using bias labels. While the intrinsic features in the unbiased dataset can simply be identified in generally appearing features in the training samples, generally appearing features in the biased dataset inevitably include bias features. Therefore, we identify the intrinsic features in the biased dataset by investigating the common features between a BA and a BC sample (i.e., a bias-contrastive pair). Here, the common features also need to be class-discerning since the common features

We note that Juyoung Lee (michael.ljy@kakaobrain.com) at Kakao Brain, who significantly contributed to this work, should have been the third author, but unfortunately, he was omitted due to the failure of adding him during the paper registration phase. * indicates equal contribution.

might include irrelevant environmental features. For example, in the above scenario, the common feature between an airplane in the sky (BA sample) and an airplane on the runway (BC sample) might include the features of wings, the body, and trees. In this case, the intrinsic features are the shape of the wings and the body that can distinguish the airplane class from the others.

Specifically, we introduce an intrinsic feature enhancement (IE) weight that identifies the spatial regions of intrinsic features commonly appearing in a bias-contrastive pair. We leverage an auxiliary sample in addition to the original input to construct the bias-contrastive pair. Since the majority of the original input from training samples are BA samples, we mainly adopt the BC samples as the auxiliary sample. To achieve this without bias information, we present a bias-negative (BN) score that identifies BC samples by employing a classification loss of a biased model. Our IE weight investigates common features in the bias-contrastive pair and identifies the class-discerning features among the common features. Within the identified intrinsic features, we enhance the features that are relatively under-exploited in the BA samples compared to the BC samples. In this way, we can explicitly provide our model with spatial guidance for intrinsic attributes while not using bias labels.

We verify the effectiveness of our method on both synthetic and real-world datasets with various levels of bias severity. Furthermore, the in-depth analysis demonstrates that our method successfully guides the model to make predictions based on the intrinsic features.

2. Related work

Debiasing with bias information. Previous approaches [2, 4, 8, 17, 21, 23] utilize bias labels or predefined bias types to encourage the model to learn intrinsic attributes for debiasing. Kim *et al*. [8], Tartaglione *et al*. [21], and Sagawa *et al*. [17] employ bias labels to encourage the model not to learn specific bias features. Wang *et al*. [23] and Bahng *et al*. [2] predefine the bias type (e.g., color, texture, etc.) and utilize such prior knowledge to supervise models to be robust against such predefined bias type. However, obtaining bias information requires additional cost, which is often infeasible in the real world.

Debiasing without bias information. Recent studies [1, 3, 6, 7, 10–15, 26] propose debiasing strategies that do not require bias information. Nam *et al*. [14] present an approach that encourages the model to concentrate on BC samples during the training process considering that the bias attributes are easier to learn than intrinsic attributes. Instead of using bias information, they additionally train a biased model that mainly learns bias attributes and regard the samples that are not easily trained by the biased model as BC samples. Lee *et al*. [12] reveal that BC samples serve as noisy samples when training the additional biased model and propose a method to eliminate such BC samples using multiple biased models. Liu *et al*. [13] regard the samples misclassified by the model trained with empirical risk minimization as BC samples and emphasize them during training of a debiased model. Also, MaskTune [1] expects the model to learn intrinsic features by fine-tuning the model with the data whose already-explored area is masked out using Grad-CAM [18].

Another stream of approaches [7, 9, 11] synthesize samples having similar characteristics with BC samples and employ them to train a debiased model. Kim *et al*. [9] synthesize images without bias attributes leveraging an image-to-image translation model [16]. Lee *et al*. [11] and Hwang *et al*. [7] augment BC samples in the feature space by employing the disentangled representations and mixup [25], respectively. A recent pair-wise debiasing method $\mathcal{X}^2$-model [26] encourages the model to retain intra-class compactness using samples generated via feature-level interpolation between BC and BA samples. However, such approaches lack explicit supervision about which features to focus on to learn intrinsic features. To address this issue, we present a debiasing method that provides spatial guidance to encourage a model to learn intrinsic features during the training while not using bias labels. We design our model architecture using bias-contrastive pairs referring to the previous studies [7, 26].

3. Methodology

3.1. Overview

As shown in Fig. 1, our framework consists of a biased model f_b that focuses on bias attributes and a debiased model f_d that learns debiased representations. We use BiasEnsemble (BE) [12] as a backbone, where f_b is trained with bias-amplified dataset $\mathcal{D}^{\text{A}}$ which mainly consists of BA samples, while f_d concentrates on the samples that f_b fails to learn. Our method provides f_d with spatial guidance for intrinsic features using a bias-contrastive pair: an input $\mathbf{x}$ and an auxiliary input $\mathbf{x}^{\text{BN}}$. We denote the auxiliary input $\mathbf{x}^{\text{BN}}$ as a bias-negative (BN) sample because we primarily adopt samples devoid of bias attributes. We sample an image $\mathbf{x}$ from the original training data $\mathcal{D}$, and $\mathbf{x}^{\text{BN}}$ from a BN dataset $\mathcal{D}^{\text{BN}}$ which mainly consists of BC samples. $\mathcal{D}^{\text{BN}}$ is updated every iteration to mainly include BC samples using the BN score S that employs f_b to identify BC samples. The BN score is also updated every iteration. Given the intermediate features $\mathbf{z}$ and $\mathbf{z}^{\text{BN}}$, we first extract the common features between the bias-contrastive pair ($c(\mathbf{z})$ in Fig. 1). Also, we identify the class-discerning features that are relatively under-exploited in $\mathbf{z}$ compared to $\mathbf{z}^{\text{BN}}$ ($r(\mathbf{z})$ in Fig. 1). Next, we calculate the IE weight that indicates relatively under-exploited intrinsic features in $\mathbf{z}$ based on $c(\mathbf{z})$ and $r(\mathbf{z})$ (IE($\mathbf{z}$) in Fig. 1). Finally, we obtain the guidance $g(\mathbf{z})$

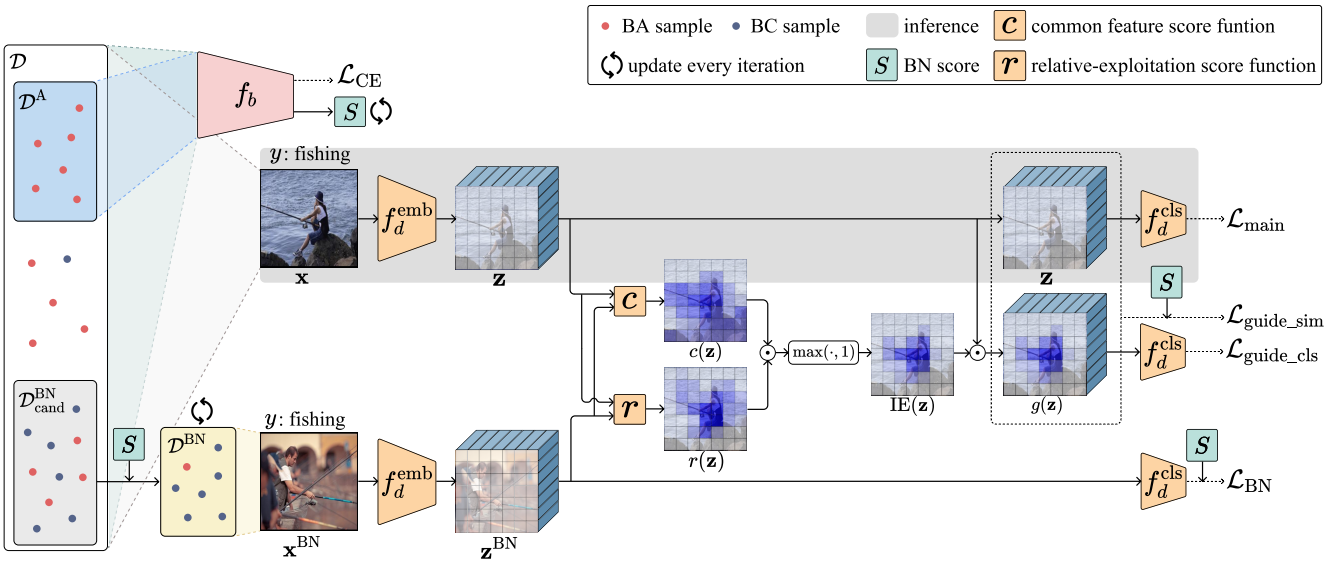


Figure 1. Overview of our method. We provide explicit spatial guidance $g(\mathbf{z})$ for a debiased model f_d, which is described with f_d^{emb} and f_d^{cls}, to learn intrinsic features. To achieve this, we leverage a bias-contrastive pair, $\mathbf{x}$ and $\mathbf{x}^{\text{BN}}$ from the same target class y. $g(\mathbf{z})$ highlights intrinsic features that are relatively under-exploited in $\mathbf{z}$ compared to $\mathbf{z}^{\text{BN}}$, calculated by common feature score c and relative-exploitation score r. Here, we mainly adopt BC samples from $\mathcal{D}^{\text{BN}}_{\text{cand}}$ to construct $\mathcal{D}^{\text{BN}}$, where we sample $\mathbf{x}^{\text{BN}}$. $\mathcal{D}^{\text{BN}}$ is updated every iteration using the BN score S, which is also updated every iteration. At the inference, we only use f_d in the gray-colored area.

that emphasizes the region of intrinsic feature in $\mathbf{z}$ during the training. At the inference, we utilize f_d without $\mathbf{x}^{\text{BN}}$, as in a gray-colored area of Fig. 1.

3.2. Constructing bias-negative dataset

We construct a BN dataset $\mathcal{D}^{\text{BN}}$, where we sample $\mathbf{x}^{\text{BN}}$ during the training. As the majority of the training dataset is BA samples, we aim to mainly adopt BC samples as $\mathbf{x}^{\text{BN}}$ to construct bias-contrastive pairs. To achieve this, we first construct $\mathcal{D}^{\text{BN}}_{\text{cand}}$, a candidate dataset for $\mathcal{D}^{\text{BN}}$, that contains roughly identified BC samples. During the training, we dynamically update $\mathcal{D}^{\text{BN}}$ every iteration to mainly adopt BC samples from $\mathcal{D}^{\text{BN}}_{\text{cand}}$ using our newly proposed BN score.

Constructing candidate dataset $\mathcal{D}^{\text{BN}}_{\text{cand}}$. To roughly identify BC samples in $\mathcal{D}$, we filter out easily learned BA samples from $\mathcal{D}$ using multiple biased models, following BE [12]. Since the bias features are easier to learn than the intrinsic features [14], each biased model is trained only for a few iterations so that BC samples can be distinguished from the easily learned BA samples. Inspired by JTT [13], we regard the samples that are incorrectly predicted by the majority of the biased models as BC samples. Finally, we construct $\mathcal{D}^{\text{BN}}_{\text{cand}}$ with the roughly identified BC samples.

Adopting BC samples with BN score. We introduce a BN score to update $\mathcal{D}^{\text{BN}}$ to primarily exploit BC samples as $\mathbf{x}^{\text{BN}}$ from $\mathcal{D}^{\text{BN}}_{\text{cand}}$ during training f_d. Considering the unavailability of bias labels, the BN score employs f_b to further exclude BA samples from $\mathcal{D}^{\text{BN}}_{\text{cand}}$. As training proceeds, f_b is overfitted to the bias attributes in $\mathcal{D}^{\text{A}}$ and outputs a high probability on the ground-truth label for the samples that have similar bias features with samples in $\mathcal{D}^{\text{A}}$. This indicates that samples whose f_b loss decreases as training proceeds are likely to have bias attributes learned from $\mathcal{D}^{\text{A}}$. Such samples disturb the extraction of intrinsic features when selected as $\mathbf{x}^{\text{BN}}$. To validate this, we investigate the samples in $\mathcal{D}^{\text{BN}}_{\text{cand}}$ whose f_b loss at the later stage of training (50K-th iteration) decreases compared to the early stage of training (1K-th iteration). The result shows that 95.63% of them are BA samples. We use the BFFHQ dataset [9] with a bias severity of 1% for the analysis. Further details of the dataset are described in Sec. 4.1.

In this regard, we design a BN score to exclude the samples with decreasing f_b loss from the $\mathcal{D}^{\text{BN}}_{\text{cand}}$ to construct $\mathcal{D}^{\text{BN}}$ by tracking the f_b loss during training f_d. First, the f_b loss of $\mathbf{x}$ at the t-th iteration is calculated as follows:

$$l_t(\mathbf{x}) = \alpha_l \cdot \mathcal{L}_{\text{CE}}(f_b(\mathbf{x}), y) + (1 - \alpha_l) \cdot l_{t-1}(\mathbf{x}), \quad (1)$$

where $\mathcal{L}_{\text{CE}}(f_b(\mathbf{x}), y)$ indicates the cross-entropy (CE) loss of $\mathbf{x}$ on its ground-truth label y and α_l is a hyperparameter for the exponential moving average (EMA). We employ EMA to enable a stable tracking of the classification losses. Note that l_t is updated only for the samples in a mini-batch at the t-th iteration.

The BN score tracks $l_t(\mathbf{x})$ compared to the loss recorded at the early stage of training. The BN score at the t-th iteration is formulated as follows:

$$s_t(\mathbf{x}) = \alpha_s \cdot (l_t(\mathbf{x}) - l_{\text{ref}}(\mathbf{x})) + (1 - \alpha_s) \cdot s_{t-1}(\mathbf{x}), \quad (2)$$

where α_s is a hyperparameter for the EMA and $l_{\text{ref}}(\mathbf{x})$ denotes the reference loss of $\mathbf{x}$ that is first recorded after a few iterations of training. We exploit EMA to stabilize the tracking. Note that we update s_t only for the samples in a mini-batch at the t-th iteration. The negative value of $s_t(\mathbf{x})$ indicates that the loss of $\mathbf{x}$ decreased compared to the early stage of training, which means that the sample is likely to contain bias attributes.

Updating $\mathcal{D}^{\text{BN}}$ with BN score. At every iteration, we update $\mathcal{D}^{\text{BN}}$ to exclude the newly detected BA samples whose BN score $s_t(\mathbf{x})$ is smaller than zero as follows:

$$\mathcal{D}_t^{\text{BN}} = \{\mathbf{x} \mid s_t(\mathbf{x}) > 0, \mathbf{x} \sim \mathcal{D}_{\text{cand}}^{\text{BN}}\}, \tag{3}$$

where $\mathcal{D}_t^{\text{BN}}$ indicates $\mathcal{D}^{\text{BN}}$ at the t-th iteration. We employ $\mathbf{x}^{\text{BN}} \sim \mathcal{D}_t^{\text{BN}}$ as an auxiliary input at the t-th iteration. In this way, we can construct a bias-contrastive pair that encourages the intrinsic attributes to be extracted as their common features. We abbreviate $\mathcal{D}_t^{\text{BN}}$ as $\mathcal{D}^{\text{BN}}$ for brevity in the rest of the paper.

3.3. Intrinsic feature enhancement

To emphasize the intrinsic features in f_d, we introduce the intrinsic feature enhancement (IE) weight that imposes a high value on the intrinsic features. The IE weight identifies the region of intrinsic features from bias-contrastive pairs by investigating 1) their common features with common feature score c and 2) class-discerning features that are relatively under-exploited in the input with relative-exploitation score r. For the explanation, we split f_d into two parts. $f_d^{\text{emb}} : \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{h\times w\times c}$ maps an input to the intermediate feature, and $f_d^{\text{cls}} : \mathbb{R}^{h\times w\times c} \rightarrow \mathbb{R}^C$ is composed of the average pooling and the linear classifier and outputs the classification logits, where $f_d(\mathbf{x}) = f_d^{\text{cls}}\left(f_d^{\text{emb}}(\mathbf{x})\right)$.

First, given the input $\mathbf{x}$, common feature score c identifies the features that are similar to the features in $\mathbf{x}^{\text{BN}}$ that has the same class label as $\mathbf{x}$ while not having bias attributes. Specifically, we extract the intermediate features $\mathbf{z} = f_d^{\text{emb}}(\mathbf{x})$ and $\mathbf{z}^{\text{BN}} = f_d^{\text{emb}}(\mathbf{x}^{\text{BN}})$, respectively. Next, we obtain the common feature score of $\mathbf{z}$ (*i.e.*, $c(\mathbf{z}) \in \mathbb{R}^{h\times w}$). Given the n-th feature of $\mathbf{z}$ (*i.e.*, $\mathbf{z}_n \in \mathbb{R}^c$), let i^*-th feature of $\mathbf{z}^{\text{BN}}$ (*i.e.*, $\mathbf{z}_{i^*}^{\text{BN}} \in \mathbb{R}^c$) be the most similar feature to $\mathbf{z}_n$, where $i^* = \arg\max_i \left(\mathbf{z}_i^{\text{BN}} \cdot \mathbf{z}_n\right)$. Then, the n-th element of $c(\mathbf{z})$ denotes the similarity score between $\mathbf{z}_n$ and $\mathbf{z}_{i^*}^{\text{BN}}$, which is formulated as follows:

$$c(\mathbf{z})_n = \frac{\mathbf{z}_{i^*}^{\text{BN}} \cdot \mathbf{z}_n}{\max_{i,j}(\mathbf{z}_i^{\text{BN}} \cdot \mathbf{z}_j)}, \tag{4}$$

where $\cdot$ indicates a dot product operation. We adopt the dot product for the similarity metric to consider both the scale and the direction of the features. The max normalization is employed to limit the score to less than one. We consider the features with a high common feature score c in $\mathbf{z}$ as features that have a high likelihood of being intrinsic features.

Next, the relative-exploitation score r identifies class-discerning features that are relatively under-exploited in $\mathbf{x}$ compared to $\mathbf{x}^{\text{BN}}$. Since most of the $\mathbf{x}^{\text{BN}}$ does not contain bias attributes, we identify class-discerning intrinsic features by investigating the features that are mainly used to predict $\mathbf{x}^{\text{BN}}$ as its target label. At the same time, we identify the features that are under-exploited in the $\mathbf{x}$ compared to the $\mathbf{x}^{\text{BN}}$. To achieve this, we use a visual explanation map of Grad-CAM [18] that imposes a higher value on the features that have more contribution to predicting a specific label. We calculate the explanation map $\text{E}(\mathbf{z})$ and $\text{E}(\mathbf{z}^{\text{BN}})$ with respect to their ground-truth labels. We apply max normalization to the explanation maps to compare the relative importance of the features in prediction. We compare the n-th value of $\text{E}(\mathbf{z})$ (*i.e.*, $\text{E}(\mathbf{z})_n$) with the i^*-th value of $\text{E}(\mathbf{z}^{\text{BN}})$, where i^* is the index of the feature in $\mathbf{z}^{\text{BN}}$ that is the most similar with $\mathbf{z}_n$. Accordingly, the n-th element of $r(\mathbf{z}) \in \mathbb{R}^{h\times w}$ is calculated as:

$$r(\mathbf{z})_n = \left(\frac{2\text{E}(\mathbf{z}^{\text{BN}})_{i^*}}{\text{E}(\mathbf{z}^{\text{BN}})_{i^*} + \text{E}(\mathbf{z})_n}\right)^{\tau}, \tag{5}$$

where τ is the amplification factor. The score becomes larger than one when the $\mathbf{z}_n$ is relatively under-exploited than $\mathbf{z}_{i^*}^{\text{BN}}$ for prediction. When $\mathbf{z}_{i^*}^{\text{BN}}$ is not used for discerning the class, $\text{E}(\mathbf{z}^{\text{BN}})_{i^*}$ becomes close to zero and the score converges to zero.

Finally, n-th element of the $\text{IE}(\mathbf{z})$ is defined as:

$$\text{IE}(\mathbf{z})_n = \max(c(\mathbf{z})_n \odot r(\mathbf{z})_n, 1), \tag{6}$$

where $\odot$ indicates the element-wise multiplication. The IE weight has a large value on the features of $\mathbf{x}$ that commonly appear in $\mathbf{x}^{\text{BN}}$ but has not exploited enough for the prediction of $\mathbf{x}$. We clip the values to be larger than one to enhance only the relatively under-exploited features in $\mathbf{z}$ while preserving the other features.

Using the IE weight, we obtain the guidance $g(\mathbf{z})$ that emphasizes the intrinsic features in $\mathbf{z}$ as follows:

$$g(\mathbf{z}) = \mathbf{z} \odot \text{IE}(\mathbf{z}). \tag{7}$$

We broadcast $\text{IE}(\mathbf{z})$ to match its shape with $\mathbf{z}$ before multiplication. During the training, this spatial guidance informs the model of where to focus to learn intrinsic features from training samples.

3.4. Training with intrinsic feature guidance

We basically train f_d with the CE loss as follows:

$$\mathcal{L}_{\text{main}} = w(\mathbf{x})\mathcal{L}_{\text{CE}}(f_d(\mathbf{x}), y) \tag{8}$$

where $w(\mathbf{x})$ is the sample reweighting value of $\mathbf{x}$ [14]. $w(\mathbf{x})$ emphasizes the samples that f_b fails to learn, which are

Algorithm 1 Debiasing with the intrinsic feature guidance

Input: pretrained biased models, training dataset $\mathcal{D}$, biased model f_b, debiased model f_d, reference iteration T_1 for BN score, starting iteration $T_2 (\geq T_1)$ to apply intrinsic feature guidance
Output: trained debiased model f_d

Construct $\mathcal{D}^{\text{A}}, \mathcal{D}^{\text{BN}}_{\text{cand}}$ from $\mathcal{D}$ using pretrained biased models
for every iteration t **do**
  Sample $\mathbf{x} \sim \mathcal{D}$
  if $t \geq T_1$ **and** $l_{\text{ref}}(\mathbf{x})$ is not initialized **then**
    $l_{\text{ref}}(\mathbf{x}) \leftarrow l(\mathbf{x})$
  end if
  if $\mathbf{x} \in \mathcal{D}^{\text{A}}$ **then**
    Train $f_b(\mathbf{x})$ with $\mathcal{L}_{\text{CE}}$
  end if
  if $t < T_2$ **then** ▷ Train w/o guidance
    Train $f_d(\mathbf{x})$ with $\mathcal{L}_{\text{main}}$
  else if $t \geq T_2$ **then** ▷ Train w/ guidance
    Update $\mathcal{D}^{\text{BN}}$
    Sample $\mathbf{x}^{\text{BN}} \sim \mathcal{D}^{\text{BN}}$
    Train $f_d(\mathbf{x}, \mathbf{x}^{\text{BN}})$ with $\mathcal{L}_{\text{total}}$
  end if
end for

mostly BC samples. The detailed description of $w(\mathbf{x})$ is included in the Supplementary.

In addition, we guide the model to focus on the region of intrinsic features through a guidance loss and a BN loss. We observe that the BN score has a higher value on the BC samples compared to the BA samples as training f_b proceeds (See Sec. 4.3). In this respect, we employ the BN score of $\mathbf{x}^{\text{BN}}$ (*i.e.*, $s(\mathbf{x}^{\text{BN}})$) to upweight the loss when BC samples are adopted as $\mathbf{x}^{\text{BN}}$. Here, we clip the value of loss weight $s(\mathbf{x}^{\text{BN}})$ to be larger than zero.

Guidance loss. To guide the model to exploit the intrinsic features from $\mathbf{x}$, we minimize the L1 distance between $g(\mathbf{z})$ and $\mathbf{z}$ as follows:

$$\mathcal{L}_{\text{guide\_sim}} = s(\mathbf{x}^{\text{BN}}) \| \text{GAP}(\mathbf{z}) - \text{GAP}\left(g(\mathbf{z})\right) \|_1, \quad (9)$$

where GAP represents the global average pooling. $s(\mathbf{x}^{\text{BN}})$ is multiplied as a loss weight to impose a high weight on the loss when BC samples are selected as $\mathbf{x}^{\text{BN}}$.

Also, we apply the CE loss to the guidance $g(\mathbf{z})$ to encourage it to include the intrinsic features that contribute to the correct prediction as follows:

$$\mathcal{L}_{\text{guide\_cls}} = w(\mathbf{x}) \mathcal{L}_{\text{CE}} \left(f_d^{\text{cls}}(g(\mathbf{z})), y \right), \quad (10)$$

where $w(\mathbf{x})$ is the reweighting value as in $\mathcal{L}_{\text{main}}$.

Finally, our guidance loss $\mathcal{L}_{\text{guide}}$ is calculated as follows:

$$\mathcal{L}_{\text{guide}} = \lambda_{\text{sim}} \mathcal{L}_{\text{guide\_sim}} + \mathcal{L}_{\text{guide\_cls}}, \quad (11)$$

where λ_{sim} is a hyperparameter to control the relative significance between the losses. We set λ_{sim} set as 0.1.

BN loss. We also employ the CE loss on $\mathbf{x}^{\text{BN}}$ to encourage the model to learn class-discerning features. This enables the IE weight to find intrinsic features among the common features. The BN loss is defined as:

$$\mathcal{L}_{\text{BN}} = s(\mathbf{x}^{\text{BN}}) \mathcal{L}_{\text{CE}}(f_d(\mathbf{x}^{\text{BN}}), y). \quad (12)$$

Here, we also exploit $s(\mathbf{x}^{\text{BN}})$ to impose high weight on the loss when $\mathbf{x}^{\text{BN}}$ is a BC sample.

Overall objective function. In summary, the overall objective function is defined as follows:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{main}} \mathcal{L}_{\text{main}} + \mathcal{L}_{\text{guide}} + \mathcal{L}_{\text{BN}}, \quad (13)$$

λ_{main} is the constant value that linearly increases from zero to one during training f_d with the guidance. This prevents the model from focusing on bias features in $\mathbf{x}$ in the early phase. The overall process of our method is provided in Algorithm 1. Here, we set T_1 and T_2 as 1K and 10K, respectively. Note that all the hyperparameters are identically applied across different datasets and bias severities. We provide further details of the training and the implementation in the Supplementary.

4. Experiments

4.1. Experimental settings

Dataset. We utilize Waterbirds [17], biased FFHQ (BFFHQ) [9], and BAR [14] for the experiments. Each dataset contains different types of target class and bias attributes: Waterbirds - {bird type, background}, BFFHQ - {age, gender}, and BAR - {action, background}. The former and the latter in the bracket indicate the target class and the bias attribute, respectively. To be specific, the Waterbirds dataset has two bird classes: waterbirds and landbirds. Most of the waterbirds are in the water background, and most of the landbirds are in the land background. In the training dataset of BFFHQ, most young people are female while most old people are male. The word 'young' indicates an age ranging between 10 and 29, and 'old' indicates an age ranging between 40 and 59. Lastly, the BAR dataset consists of six classes of action (e.g., fishing), where the background (e.g., water surface) is highly correlated with each class. Following the previous studies [11, 12, 14], we validate our model's effectiveness under different levels of bias severity, *i.e.*, a ratio of BC samples to the total training samples: 0.5%, 1%, 2%, and 5%. In the test sets, the spurious correlations found in the training set do not exist. More details are provided in Supplementary.

Evaluation. We report the best accuracy of the test set averaged over five independent trials with different random seeds. The Waterbirds dataset has an extremely skewed test

Method	Waterbirds				BFFHQ				BAR	
	0.5	1.0	2.0	5.0	0.5	1.0	2.0	5.0	1.0	5.0
Vanilla [5]	57.41	58.07	61.04	64.13	55.64	60.96	69.00	82.88	70.55	82.53
HEX [23]	57.88	58.28	61.02	64.32	56.96	62.32	70.72	83.40	70.48	81.20
LNL [8]	58.49	59.68	62.27	66.07	56.88	62.64	69.80	83.08	-	-
EnD [21]	58.47	57.81	61.26	64.11	55.96	60.88	69.72	82.88	-	-
ReBias [2]	55.44	55.93	58.53	62.14	55.76	60.68	69.60	82.64	73.04	83.90
LfF [14]	60.66	61.78	58.92	61.43	65.19	69.24	73.08	79.80	70.16	82.95
DisEnt [11]	59.59	60.05	59.76	64.01	62.08	66.00	69.92	80.68	70.33	83.13
LfF+BE [12]	61.22	62.58	63.00	63.48	67.36	75.08	80.32	85.48	73.36	83.87
DisEnt+BE [12]	51.65	54.10	53.43	54.21	67.56	73.48	79.48	84.84	73.29	84.96
Ours	**63.64**	**65.22**	**65.23**	**66.33**	**71.68**	**77.56**	**83.08**	**87.60**	**75.14**	**85.03**

Table 1. Comparison to the baselines. We measure the classification accuracy on test sets with different bias severities. The best accuracy values are in bold. The hyphen mark '-' means it is not applicable. Results with standard deviations are provided in the Supplementary.

	$\mathcal{D}^{\text{BN}}_{\text{cand}}$ - $\mathcal{D}^{\text{BN}}$		$\mathcal{D}^{\text{BN}}/\mathcal{D}$ (%)	
Dataset	BA	BC	BA	BC
Waterbirds	26.50 $\pm$5.32	0.75 $\pm$0.83	2.75 $\pm$0.31	79.69 $\pm$3.72
BFFHQ	199.80 $\pm$40.14	8.00 $\pm$2.76	0.46 $\pm$0.09	50.00 $\pm$1.04
BAR	30.60 $\pm$3.83	3.20 $\pm$1.60	3.58 $\pm$0.14	47.14 $\pm$5.71

Table 2. Effectiveness of BN score on excluding BA samples. $\mathcal{D}^{\text{BN}}_{\text{cand}}$ - $\mathcal{D}^{\text{BN}}$ presents the number of excluded samples when constructing $\mathcal{D}^{\text{BN}}$ from $\mathcal{D}^{\text{BN}}_{\text{cand}}$. $\mathcal{D}^{\text{BN}}/\mathcal{D}$ indicates that the ratio of samples in $\mathcal{D}^{\text{BN}}$ to the samples in $\mathcal{D}$.

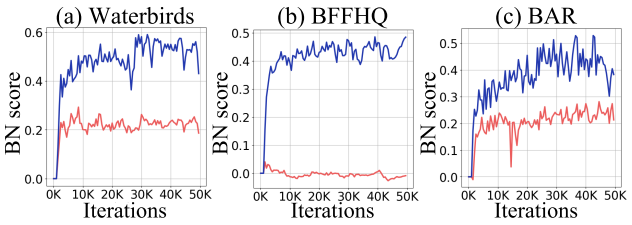


Figure 2. Visualization of BN scores of the samples in $\mathcal{D}^{\text{BN}}_{\text{cand}}$ during the training. The red lines and the blue lines indicate the BN scores of BA and BC samples, respectively.

dataset composed of 4,600 landbirds and 1,194 waterbirds. This can mislead the debiasing performance as the model may achieve high classification accuracy by simply predicting most images as landbirds. We measure the classification accuracy for each class and report their average value to obtain an accurate understanding of the effectiveness of methods, regardless of class frequencies. Also, for the BFFHQ, we report the best accuracy of BC samples in the test set, following the previous works [11, 12]. For the analyses, we utilize the datasets with 1% bias severity.

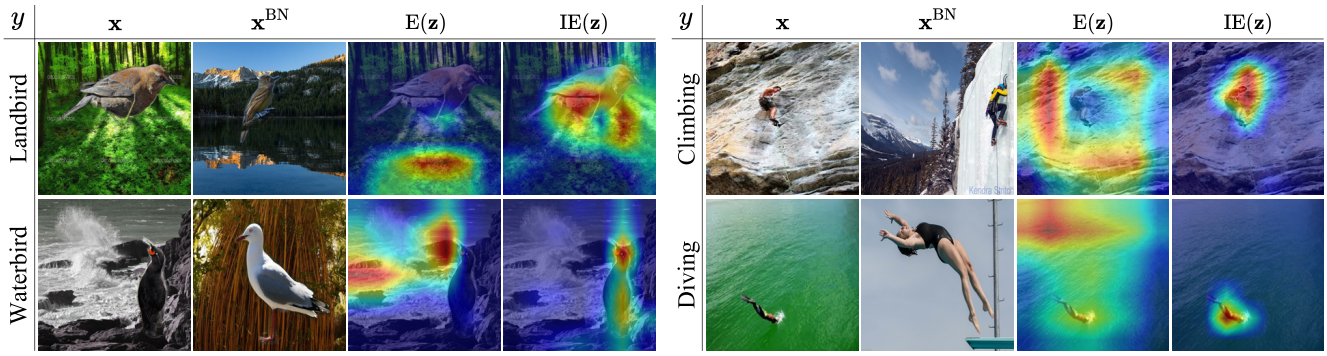
4.2. Comparison to previous works

We compare the classification accuracy on the test sets between the baselines and ours in Table 1. For baselines, we employ a vanilla model trained with the CE loss, the methods using explicit bias label (*i.e.*, LNL [8], EnD [21]), presuming the type of the bias (*i.e.*, HEX [23], ReBias [2]), and assuming the bias information is unknown (*i.e.*, LfF [14], DisEnt [11], LfF+BE [12], DisEnt+BE [12]). Our approach achieves state-of-the-art performance in comparison to the previous methods including those utilizing explicit bias labels. The results exhibit that our method improves performance robustly across various levels of bias severity, even under the constraints of extreme bias severity (*e.g.*, 0.5 %). This shows that providing explicit spatial guidance for intrinsic features effectively encourages the model to learn debiased representations, leading to performance improvement.

4.3. Analysis of BN score

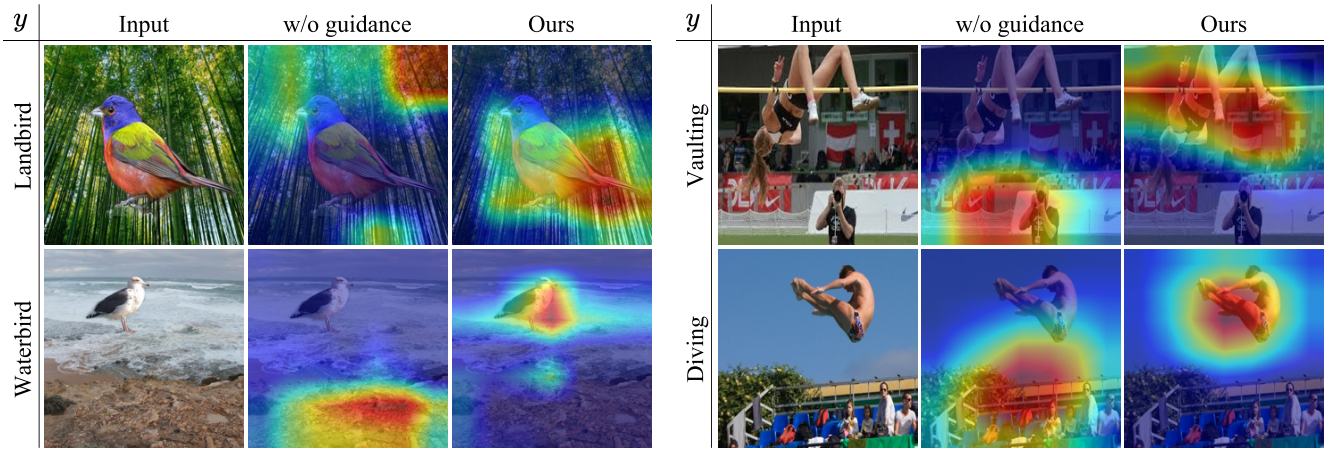
We analyze our BN score that identifies and emphasizes BC samples in $\mathcal{D}^{\text{BN}}_{\text{cand}}$ during training f_d. In this section, we assess the effectiveness of the BN score on excluding BA samples from $\mathcal{D}^{\text{BN}}_{\text{cand}}$. Also, we evaluate the efficacy of the BN score as a loss weight by investigating the BN scores of the samples in $\mathcal{D}^{\text{BN}}_{\text{cand}}$.

Effectiveness of BN score on excluding BA samples. Table 2 presents how the BN score effectively filters out BA samples from $\mathcal{D}^{\text{BN}}_{\text{cand}}$ while preserving BC samples when constructing $\mathcal{D}^{\text{BN}}$. The first two columns present the number of the BA and BC samples excluded from $\mathcal{D}^{\text{BN}}_{\text{cand}}$ to construct $\mathcal{D}^{\text{BN}}$, respectively. Also, the last two columns represent the ratio of the number of the BA and BC samples in $\mathcal{D}^{\text{BN}}$ to that in $\mathcal{D}$, respectively. For the analysis, we use $\mathcal{D}^{\text{BN}}$ at the 50K-th iteration and report the mean value of the five independent trials. Here, we expect $\mathcal{D}^{\text{BN}}$ to contain a maximal number of BC samples while including a minimal number of BA samples. As shown in the first two columns



(a) Waterbirds (b) BAR

Figure 3. Visualization of the spatial guidance using (a) Waterbirds and (b) BAR dataset. Given bias-contrastive pairs, $\mathbf{x}$ and $\mathbf{x}^{\text{BN}}$, $\text{E}(\mathbf{z})$ indicates the regions originally focused on by f_d and $\text{IE}(\mathbf{z})$ shows the regions highlighted by our IE weight.



(a) Waterbirds (b) BAR

Figure 4. Comparison of the region focused by a debiased model trained with and without our method. We compare Grad-CAM results on the test set of (a) Waterbirds and (b) BAR.

of Table 2, our BN score excludes a large number of BA samples while minimizing the loss of BC samples. As a result, $\mathcal{D}^{\text{BN}}$ preserves around 50-80% of BC samples, while containing a minimal number of BA samples compared to $\mathcal{D}$, as presented in the last two columns.

Efficacy of BN score as loss weight. We utilize the BN score to upweight the training loss when BC samples are chosen as $\mathbf{x}^{\text{BN}}$. To verify its effectiveness as a loss weight, we compare the BN scores of BC samples and BA samples in $\mathcal{D}^{\text{BN}}_{\text{cand}}$ during the training for the Waterbirds, BFFHQ, and BAR dataset. In Fig. 2, we present the average BN scores of BA (red line) and BC samples (blue line) in $\mathcal{D}^{\text{BN}}_{\text{cand}}$ at every 500 iterations. Since BN scores are recorded after the 1K-th iteration, the BN scores until the 1K-th iteration are reported as zero. The BN scores of BC samples in the BFFHQ dataset mostly range from 0.4 to 0.5 while the scores of BA samples are close to 0. This indicates that the BN score as a loss weight in the BFFHQ imposes a much larger value on the BC samples, while approximately zero values on the BA samples. Also, the BN scores of BC samples in the Waterbirds and the BAR dataset become twice larger than the scores of BA samples. The result shows that the BN score effectively emphasizes BC samples compared to the BA samples in $\mathcal{D}^{\text{BN}}_{\text{cand}}$ during the training. Further analysis of the BN score is included in the Supplementary.

4.4. Analysis of intrinsic feature guidance

We conduct a qualitative analysis of the regions emphasized by our intrinsic feature guidance during the training and the features learned by f_d after training. We use the Waterbirds and BAR datasets with 1% of bias severity for the analysis.

Visualization of the guidance during training. In Fig. 3, we visualize the features emphasized by our IE weight $\text{IE}(\mathbf{z})$. For comparison, we also visualize $\text{E}(\mathbf{z})$, the features focused by the model before applying $\text{IE}(\mathbf{z})$ for guidance. We select a BA sample as $\mathbf{x}$ and a BC sample as $\mathbf{x}^{\text{BN}}$ from the training data for the analysis. For the Waterbirds dataset in Fig. 3(a), $\text{IE}(\mathbf{z})$ highlights the wings or the beak

of the bird compared to $\mathrm{E}(\mathbf{z})$, where the forest or the water (*i.e.*, bias attributes) is highlighted. Also, in Fig. 3(b), $\mathrm{E}(\mathbf{z})$ focuses more on the bias attributes such as rocks or the water than the intrinsic attributes. In contrast, $\mathrm{IE}(\mathbf{z})$ emphasizes the action of the human that is less exploited compared to the bias features in $\mathrm{E}(\mathbf{z})$. The results demonstrate that our guidance successfully identifies and enhances under-exploited intrinsic features during the training.

Effect of intrinsic feature guidance on debiasing. We qualitatively evaluate the effectiveness of the intrinsic feature guidance by investigating the visual explanation maps of the test samples. We compare the Grad-CAM [18] results of the model trained with and without our method in Fig 4. The Grad-CAM results highlight the features that the model employs to predict the input as its ground-truth label. In Fig. 4 (a), while the model trained without guidance focuses on the forest or the sea, ours focuses on the tail or a curved shape of the bird's body. Additionally, Fig. 4 (b) shows that ours focuses on the motion of the human rather than the backgrounds that are concentrated on by the model trained without our guidance. The results verify that our method successfully encourages the model to learn intrinsic features from the training dataset, improving the robustness of the model against dataset bias.

4.5. Ablation study

As shown in Table 3, we perform ablation studies to verify the effectiveness of the individual components in our method. The results of ours are reported in the last row.

Importance of $\mathbf{x}^{\mathrm{BN}}$ selection and BN score as loss weight. We demonstrate the efficacy of adopting BC samples as $\mathbf{x}^{\mathrm{BN}}$. We train the model by sampling $\mathbf{x}^{\mathrm{BN}}$ from three different datasets: $\mathcal{D}$, $\mathcal{D}^{\mathrm{BN}}_{\mathrm{cand}}$, and $\mathcal{D}^{\mathrm{BN}}$. In the first row of Table 3, we randomly sample $\mathbf{x}^{\mathrm{BN}}$ from the training dataset $\mathcal{D}$ without using the BN score. In the second row, we train the model with $\mathbf{x}^{\mathrm{BN}}$ sampled from $\mathcal{D}^{\mathrm{BN}}_{\mathrm{cand}}$, where BA samples are roughly filtered out using the early-stopped biased models. The model in the third row is trained with $\mathbf{x}^{\mathrm{BN}}$ from $\mathcal{D}^{\mathrm{BN}}$ that mainly includes BC samples using our BN score. The results show a gradual improvement in debiasing performance as more BC samples are selected as $\mathbf{x}^{\mathrm{BN}}$. This is because auxiliary samples without bias attributes prevent the common features from including the bias features, composing a bias-contrastive pair with the input. Therefore, our guidance effectively enhances the intrinsic features during the training. Finally, the last row in Table 3 presents that employing the BN score $s(\mathbf{x}^{\mathrm{BN}})$ to reweight the losses further enhances performance by emphasizing the usage of BC samples as $\mathbf{x}^{\mathrm{BN}}$.

Training objectives. We examine the impact of each training objective, $\mathcal{L}_{\mathrm{guide}}$ and $\mathcal{L}_{\mathrm{BN}}$, in our method. We report the performance of the model trained without $\mathcal{L}_{\mathrm{guide}}$ (the fourth row) and without $\mathcal{L}_{\mathrm{BN}}$ (the fifth row) in Table 3.

$\mathcal{L}_{\mathrm{guide}}$	$\mathcal{L}_{\mathrm{BN}}$	$\mathbf{x}^{\mathrm{BN}}$	$s(\mathbf{x}^{\mathrm{BN}})$ as loss weight	Waterbirds	BFFHQ	BAR
✓	✓	$\mathcal{D}$	✗	62.79 ±1.21	71.04 ±2.55	73.36 ±1.40
✓	✓	$\mathcal{D}^{\mathrm{BN}}_{\mathrm{cand}}$	✗	64.65 ±1.23	75.64 ±1.87	74.27 ±0.66
✓	✓	$\mathcal{D}^{\mathrm{BN}}$	✗	65.10 ±0.87	77.08 ±2.05	74.62 ±1.07
✗	✓	$\mathcal{D}^{\mathrm{BN}}$	✓	63.81 ±1.24	76.92 ±1.03	74.03 ±1.13
✓	✗	$\mathcal{D}^{\mathrm{BN}}$	✓	62.10 ±3.35	74.84 ±2.00	74.87 ±1.51
✓	✓	$\mathcal{D}^{\mathrm{BN}}$	✓	**65.22** ±0.95	**77.56** ±1.24	**75.14** ±0.82

Table 3. Ablation study on the proposed training objectives, the dataset that $\mathbf{x}^{\mathrm{BN}}$ is sampled from, and the BN score of $\mathbf{x}^{\mathrm{BN}}$ as a loss weight. The check mark (✓) denotes the inclusion of the corresponding method, while the cross mark (✗) indicates the exclusion of the component in the experiment.

The model trained without $\mathcal{L}_{\mathrm{guide}}$ exhibits degraded performance, facing difficulties in identifying where to focus to learn intrinsic features. Similarly, training the model without $\mathcal{L}_{\mathrm{BN}}$ also results in a performance decrease. The results verify that $\mathcal{L}_{\mathrm{BN}}$ successfully supports the IE weight to identify intrinsic features among the common features by learning class-discerning features from $\mathbf{x}^{\mathrm{BN}}$. The model that incorporates both $\mathcal{L}_{\mathrm{guide}}$ and $\mathcal{L}_{\mathrm{BN}}$ demonstrates the best performance (the last row of Table 3).

5. Conclusion

In this paper, we propose a debiasing method that explicitly provides the model with spatial guidance for intrinsic features. Leveraging an auxiliary sample, we first identify intrinsic features by investigating the class-discerning features commonly appearing in a bias-contrastive pair. Our IE weight enhances the intrinsic features that have not been focused on yet in the input by a debiased model. To construct the bias-contrastive pair without bias labels, we introduce a bias-negative (BN) score that tracks the classification loss of a biased model to distinguish BC samples from BA samples during the training. The effectiveness of our method is demonstrated through experiments on synthetic and real-world datasets with varying levels of bias severity. We believe this work sheds light on the significance of providing explicit guidance on the intrinsic attributes for debiasing.

Acknowledgments

This work was supported by the Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program(KAIST), the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2022R1A2B5B02001913 & No. 2022R1A5A7083908) and partly by Kakao Brain Corporation.

References

[1] Saeid Asgari, Aliasghar Khani, Fereshte Khani, Ali Gholami, Linh Tran, Ali Mahdavi Amiri, and Ghassan Hamarneh. Masktune: Mitigating spurious correlations by forcing to explore. In *Proc. the Advances in Neural Information Processing Systems (NeurIPS)*, pages 23284–23296, 2022. 2

[2] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *Proc. the International Conference on Machine Learning (ICML)*, 2020. 1, 2, 6

[3] Luke Darlow, Stanislaw Jastrzebski, and Amos Storkey. Latent adversarial debiasing: Mitigating collider bias in deep neural networks. *arXiv preprint arXiv:2011.11486*, 2020. 2

[4] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *Proc. the International Conference on Learning Representations (ICLR)*, 2019. 2

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016. 1, 6

[6] Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2020. 2

[7] Inwoo Hwang, Sangjun Lee, Yunhyeok Kwak, Seong Joon Oh, Damien Teney, Jin-Hwa Kim, and Byoung-Tak Zhang. Selecmix: Debiased learning by contradicting-pair sampling. In *Proc. the Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 2

[8] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2019. 1, 2, 6

[9] Eungyeup Kim, Jihyeon Lee, and Jaegul Choo. Biaswap: Removing dataset bias with bias-tailored swapping augmentation. In *Proc. of the IEEE international conference on computer vision (ICCV)*, pages 14992–15001, 2021. 2, 3, 5

[10] Bum Chul Kwon, Jungsoo Lee, Chaeyeon Chung, Nyoungwoo Lee, Ho-Jin Choi, and Jaegul Choo. DASH: Visual Analytics for Debiasing Image Classification via User-Driven Synthetic Data Augmentation. *Eurographics Conference on Visualization (EuroVis)-Short Papers. The Eurographics Association*, 2022. 2

[11] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. Learning debiased representation via disentangled feature augmentation. In *Proc. the Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1, 2, 5, 6

[12] Jungsoo Lee, Jeonghoon Park, Daeyoung Kim, Juyoung Lee, Edward Choi, and Jaegul Choo. Revisiting the importance of amplifying bias for debiasing. In *Proc. the AAAI Conference on Artificial Intelligence (AAAI)*, pages 14974–14981, 2023. 2, 3, 5, 6

[13] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *Proc. the International Conference on Machine Learning (ICML)*, pages 6781–6792, 2021. 2, 3

[14] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: Training debiased classifier from biased classifier. In *Proc. the Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2, 3, 4, 5, 6

[15] Geon Yeong Park, Sangmin Lee, Sang Wan Lee, and Jong Chul Ye. Training debiased subnetworks with contrastive weight pruning. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 7929–7938, 2023. 2

[16] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A. Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In *Proc. the Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[17] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *Proc. the International Conference on Learning Representations (ICLR)*, 2020. 2, 5

[18] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proc. of the IEEE international conference on computer vision (ICCV)*, 2017. 2, 4, 8

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *Proc. the International Conference on Learning Representations (ICLR)*, 2015. 1

[20] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1–9, 2015. 1

[21] Enzo Tartaglione, Carlo Alberto Barbano, and Marco Grangetto. End: Entangling and disentangling deep representations for bias correction. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 13508–13517, 2021. 1, 2, 6

[22] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1521–1528, 2011. 1

[23] Haohan Wang, Zexue He, Zachary L. Lipton, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In *Proc. the International Conference on Learning Representations (ICLR)*, 2019. 1, 2, 6

[24] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *Proc. of the British Machine Vision Conference (BMVC)*, 2016. 1

[25] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proc. the International Conference on Learning Representations (ICLR)*, 2018. 2

[26] Yi-Kai Zhang, Qi-Wei Wang, De-Chuan Zhan, and Han-Jia Ye. Learning debiased representations via conditional attribute interpolation. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 7599–7608, 2023. 2